Engineering Systems: Structure, Behavior, and Optimization with Hetero-functional Graph Theory

A Thesis
Submitted to the Faculty
in partial fulfillment of the requirements for the
degree of

Doctor of Philosophy

in

Engineering Sciences

by Wester C.H. Schoonenberg

Thayer School of Engineering
Guarini School of Graduate and Advanced Studies
Dartmouth College
Hanover, New Hampshire

March 2021

Examining Committee:

Chairman _____
Prof. Amro M. Farid

Member _____
Prof. Eugene Santos Jr.

Member _____
Prof. Geoffrey G. Parker

Member _____
Prof. Olivier L. de Weck

_____
F. Jon Kull, Ph.D.
Dean of Guarini School of Graduate and Advanced Studies

Abstract

Our modern life has grown to depend on many and nearly ubiquitous large complex engineering systems. Over the past decades, engineering solutions have evolved from singular technical artifacts into engineering systems that deliver multiple services within a societal and economic context. As these engineering systems perform more services, the delivery of such services becomes increasingly interdependent. Examples include electrified transportation systems, the energy-water nexus, and microgrid-enabled production systems. The interdependence of these systems raises timely questions about resilience and sustainability.

Due to their complexity, these engineering systems require a novel way to be measured, managed, and optimized. This thesis therefore concentrates on the advancement of a modeling framework to enable the analysis of engineering systems. Such a modeling framework is required to accommodate both the extreme heterogeneity of engineering systems as well as their quantitative nature to enable study of engineering system structure, dynamic behavior, and optimal control.

This thesis builds on a Hetero-functional Graph Theory as a modeling framework that accommodates both the heterogeneity and quantitative nature of engineering systems. The work aims to root the mathematical concepts of Hetero-functional Graph Theory in the engineering systems and systems engineering literature, it aims to provide a single consistent overview of the theory, it aims to explore new application domains for dynamic models based on Hetero-functional Graph Theory, and it aims to develop a framework to optimize such dynamic models.

Acknowledgements

I would like to start my thesis by thanking all the people and institutions who have supported and guided me over the past years.

First and foremost, I would like to thank my advisor, Prof. Amro M. Farid. His efforts to guide me have been a critical component in my development during my intercontinental journey in the LIINES (Laboratory for Intelligent and Integrated Networks of Engineering Systems). He took the time to help me build my mathematical and coding skills, while emphasizing a rigorous approach to systems thinking. He also gave me the opportunity to publish a book on hetero-functional graph theory and to represent the LIINES at public fora. Most of all, Prof. Farid is a mentor and a teacher.

I would like to thank my committee members, Prof. Parker and Prof. Santos at Dartmouth's Engineering School, and Prof. De Weck at MIT for their time and guidance.

I would like to thank the Ph.D. in Innovation Program and its director, Prof. Fossum. Your program has broadened my horizons and without your funding I wouldn't have been able to do this work.

My work is part of a larger effort towards more sustainable, resilient, and efficient engineering systems by all members in the LIINES. I would like to thank my colleagues for their collaboration, brainstorm sessions, and coffee chats. I would also like to thank Prof. Inas S. Khayal, the co-author of the hetero-functional graph theory book, for her guidance, input, and collaboration.

In addition to my colleagues, my friends from Utrecht, Delft, Abu Dhabi, and Hanover

have provided their support during stressful periods and celebrated successes after small and large accomplishments. Thank you for your advice, ski trips, and dinner parties.

Naturally, I would like to thank my family. Hendrik, Lidia, I love you. Thank you for accommodating my needs, solving problems as partners, and defining and defying all expectations. Also to my new family, thank you for welcoming me and supporting us, especially with the baby during this pandemic. Vivien, thank you for all our meet-ups during your time in Boston and thank you for sharing the Ph.D. experience. Parents, thank you for the remote support, voices of reason, and for making your way across the pond on a regular basis. And Floris, you are like a brother, thank you for our continued friendship.

Finally, I would like to thank every reader of this thesis. I hope the work below is of value to you. Please reach out if you have any questions.

Preface

This thesis is the culmination of the author's graduate studies at Dartmouth College's Thayer School of Engineering. The thesis aims to tell the story of the advancement of hetero-functional graph theory for the analysis of structure, dynamics, and optimization of engineering systems. This preface aims to provide context to the thesis through a brief introduction to engineering systems, a backstory of hetero-functional graph theory, and an overview of the publications leveraged in each chapter.

Engineering systems are understood to be *"a class of systems characterized by a high degree of technical complexity, social intricacy, and elaborate processes, aimed at fulfilling important functions in society"* [1]. These services range from providing transportation and logistics services to the delivery of electric power (Engineering Systems are more extensively introduced in Chapters 1 and 2). This thesis proposes a hetero-functional graph theory as a mathematical framework for the development of quantitative models to enhance the ability to understand, analyze, and operate engineering systems.

Hetero-functional Graph Theory originates in the work of the author's advisor, Prof. Amro M. Farid, initiated over a decade ago during his Ph.D. work [2]. Prof. Farid's thesis [2] adopts concepts from the Axiomatic Design literature (specifically, the work by Prof. Nam Suh [3]) which has lead to the development of Axiomatic Design for Large Flexible Engineering Systems and the book *"Axiomatic Design in Large Systems: Complex Products, Buildings and Manufacturing Systems"* [4]. That book on Axiomatic Design also used the term *"Hetero-functional Network"* for the first time. Shortly thereafter, the theory

was compiled to provide a complete and consistent overview of hetero-functional graph theory for engineering system structure in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5].

The author's work is part of a larger effort in the LIINES (Laboratory for Intelligent Integrated Networks of Engineering Systems) to advance the theory and analytics for engineering systems. Below, an overview is provided of the lab's work related to hetero-functional graph theory across various application domains.

1. Mass-Customized Production Systems [2, 6–22]
2. Multi-Modal Transportation Systems [23–25].
3. Electric Power Systems [26–29].
4. (Multi-Modal) Electrified Transportation Systems [30–33]
5. Microgrid-Enabled Production Systems [34, 35]
6. Personalized Healthcare Delivery Systems [36–46]

For a more extensive discussion of these works, the reader is referred to Chapter 6 of the book on hetero-functional graph theory [5] and to the website of the LIINES [47].

This thesis establishes the contributions that the author has made to advance the literature in the field of Engineering Systems through hetero-functional graph theory and its applications. The list below presents an overview of the chapters in the dissertation and their associated publications. Chapters 1 and 2 incorporate and rearrange multiple existing publications to establish the story of the thesis with clarity. The remainder of the chapters integrate relevant publications in a straightforward manner.

- **Chapter 1: Introduction** draws on the first chapter in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5] (Chapter 1 entitled *"Introduction"*). Furthermore, the chapter also draws on the 23rd chapter in

the *"Handbook for Engineering System Design"* [48] (Chapter 23 entitled *"Evaluating Engineering System Interventions"*).

- **Chapter 2: Background** draws on the 23rd chapter in the *"Handbook for Engineering System Design"* [48] (Chapter 23 entitled *"Evaluating Engineering System Interventions"*). Furthermore, the chapter draws on Chapters 2 and 3 in the book on hetero-functional graph theory [5] (Chapter 2 entitled *"The Need for Hetero-functional Graph Theory"*, and Chapter 3 entitled *"Hetero-functional Graph Theory Preliminaries"*).

- **Chapter 3: Hetero-functional Graph Theory** incorporates the fourth chapter from the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5] (Chapter 4 entitled *"Hetero-functional Graph Theory"*) into the thesis.

- **Chapter 4: Modeling Interdependent Smart City Infrastructures with Hetero-functional Graph Theory** incorporates the fifth chapter and the Appendix into the thesis, both drawn from the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5] (Chapter 5 entitled *"Modeling Interdependent Smart City Infrastructure Systems with HFGT"* and Appendix A entitled *"Representing a Four-Layer Network in Hetero-functional Graph Theory"*).

- **Chapter 5: Dynamic Systems Modeling with Hetero-functional Graph Theory** presents the author's first strides towards advancing hetero-functional graph theory. The chapter incorporates an article published in the Journal of Cleaner Production entitled *"A Dynamic Model for the Energy-Management of Microgrid-Enabled Production Systems"* [35].

- **Chapter 6: Optimization of Dynamic Systems with Hetero-functional Graph Theory** seeks to advance an optimization program for a hetero-functional graph

theory-based dynamic model. The work in this chapter is to be published.

- **Chapter 7: Conclusion** concludes the thesis and draws on the different chapters and publications listed above.

Finally, as engineering systems span a broad set of engineering disciplines, the advancement of hetero-functional graph theory requires demonstration through a variety of applications. The three most prominent application domains featured in this thesis are:

1. Interdependent Smart City Infrastructures,
2. Microgrid-Enabled Production Systems, and
3. Hydrogen-Natural Gas Systems.

Wester C.H. Schoonenberg

Hanover, NH, USA

February 2021

Contents

List of Tables

List of Figures

Nomenclature

Ontological Symbols

$\mathcal{A}$	Abstraction, equals the mental conceptualization of the modeller. . . . . *p. 43*
$\mathcal{C}$	Domain Conceptualization, equals the understanding of reality. . . . . *p. 43*
$\mathcal{D}$	Real Domain, equals reality. . . . . *p. 43*
$\mathcal{L}$	Language, equals the description of reality. . . . . *p. 43*
$\mathcal{M}$	Model, equals the description of the abstraction. . . . . *p. 43*

Sets

$\mathbf{M}_{l_i}$	Set of service net arcs: (service states to service activities) and (service activities to service states) . . . . . *p. 98*
$\mathbf{M}_{PN}$	Set of arcs from places to transitions and from transitions to places in the Petri net graph. . . . . *p. 228*
$\overline{R}$	Set of Aggregated Resources . . . . . *p. 72*
$\overline{R}_P$	Set of Aggregated Physical Resources . . . . . *p. 89*
$\mathbf{M}_{\mathcal{C}}$	Set of ac-CPN Arcs . . . . . *p. 262*
$\text{Bag}(cd(s_c))$	Multiset of Color Classes for ac-CPN place s_c . . . . . *p. 262*
θ_ψ	The associated voltage angle relative to a predefined reference bus . . . . . *p. 238*
B	Set of Independent Buffers . . . . . *p. 64*
B_S	Set of Buffers . . . . . *p. 65*
H	Set of Transportation Resources . . . . . *p. 64*
k_d	Set of transition durations between the negative and positive firing vectors . *p. 258*
L	Set of Services . . . . . *p. 96*
M	Set of Transformation Resources . . . . . *p. 64*

$\mathcal{E}_{LS}$ Set of Service Degrees of Freedom . . . . . p. 110

$\mathcal{Z}$ Set of System Activity Strings . . . . . p. 81

Set Elements

$\overline{r}_a$ Aggregated resource a in set $\overline{R}$ . . . . . p. 224

b Independent buffer in set of independent buffers B . . . . . p. 64

b_{sy_1} Origin buffer y_1 in set of buffers B_S . . . . . p. 65

b_{sy_2} Destination buffer y_2 in set of buffers B_S . . . . . p. 65

d_ψ Event duration of action ψ . . . . . p. 229

e_{wv} System action w, v in set of system actions $\mathcal{E}$ . . . . . p. 71

e_{xl_i} Service activity x in Service l_i . . . . . p. 96

h Transporter in set of transporters H . . . . . p. 64

$k_{d\psi}$ Transition duration of DOF ψ in the Set k_d . . . . . p. 258

l_i Service i in the set of services L . . . . . p. 96

m Machine in set of machines M . . . . . p. 64

p Process in set of system process P . . . . . p. 65

p_Q Decision algorithm in the set of decision algorithms P_Q . . . . . p. 86

$p_{\bar{\eta}\varphi}$ Refined transportation process φ in set of refined transportation processes $P_{\bar{\eta}}$ p. 67

$p_{\eta u}$ Transportation process u in set of transportation processes P_η . . . . . p. 65

$p_{\gamma g}$ Holding process g in set of holding processes P_γ . . . . . p. 66

$p_{\mu j}$ Transformation process j in set of transformation processes P_μ . . . . . p. 65

q Cyber-resource in the set of cyber-resources Q . . . . . p. 86

r Resource in set of system resources R . . . . . p. 64

$s_{\zeta l_i}$ Service place ζ for service l_i . . . . . p. 98

t_κ Time at time step $t_k + d_\psi$ . . . . . p. 229

t_k Time at time step k . . . . . p. 229

$u_\psi[k]$ Elements in the firing vector at associated times t_k . . . . . p. 229

z_{χ_1,χ_2} String of two sequential activities χ_1 and χ_2 in set of strings $\mathcal{Z}$ . . . . . p. 81

$u_{\psi,l}[k]$ Element of product firing matrix $\mathcal{U}[k]$ . . . . . p. 237

Indices

χ Index in $[1 \dots \sigma(R)\sigma(P)]$ . . . . . p. 81

ψ Index of the elements in the set of structural degrees of freedom $\mathcal{E}_S$ . . . . . p. 83

φ Index of refined transportation process $p_{\bar{\eta}\varphi}$ in $P_{\bar{\eta}}$ . . . . . p. 67

ζ Index of service place $s_{\zeta l_i}$ in the set of service places S_{l_i} . . . . . p. 98

a Index for aggregated resource $\overline{r}_a$ in set $\overline{R}$ . . . . . p. 224

g Index of holding process $p_{\gamma g}$ in set of holding processes P_γ . . . . . p. 66

i Index of Service l_i in the set of services L . . . . . p. 96

j Index of transformation process $p_{\mu j}$ in set of transformation processes P_μ . . . . p. 65

k Index of time . . . . . p. 99

u Index of transportation process $p_{\eta u}$ in set of transportation processes P_η . . . . . p. 65

v Index of physical resource r_v in set of physical resources R . . . . . p. 71

w Index of system process p_w in set of system processes P . . . . . p. 71

x Index of service activity e_{xl_i} in the set of service activities $\mathcal{E}_{l_i}$ . . . . . p. 96

y_1 Index of origin buffer b_{sy_1} in set of buffers B_S . . . . . p. 65

y_2 Index of destination buffer b_{sy_2} in set of buffers B_S . . . . . p. 65

Mathematical Symbols

$\Lambda_{\gamma i}$ Service Transportation Feasibility Matrix for product l_i . . . . . p. 104

$\Lambda_{\mu i}$ Service Transformation Feasibility Matrix for product l_i . . . . . p. 104

Λ_{Hi} Transportation service selector matrix . . . . . p. 108

Λ_{HL} Transportation service line selector matrix . . . . . p. 108

Λ_{Hxi} Transportation service activity selector matrix . . . . . p. 108

Λ_i Service Feasibility Matrix for product l_i . . . . . p. 105

Λ_{Mi} Transformation service selector matrix . . . . . p. 108

Λ_{ML} Transformation service line selector matrix . . . . . p. 108

Λ_{Mxi} Transformation service activity selector matrix . . . . . p. 108

Λ_{SHi} System Transportation Service selector matrix . . . . . p. 108

Λ_{SHxi} System Transportation service activity selector matrix . . . . . p. 108

Λ_{Si} System Transformation Service selector matrix . . . . . p. 108

Λ_{SL} System Service Line selector matrix . . . . . p. 108

Λ_{SMxi} System Transformation service activity selector matrix . . . . . p. 108

Λ_{Sxi} System Transformation and Transportation service activity selector matrix . p. 108

$\mathbf{V}$ Vector of nodal voltage levels . . . . . p. 240

$\mathbf{Y}$ Bus admittance matrix . . . . . p. 240

$\mathbb{1}^n$ Ones-vector of length n . . . . . p. 73

$\mathbb{A}$ System Adjacency Matrix . . . . . p. 112

$\mathbb{A}_L$ System Service Adjacency Matrix . . . . . p. 112

$\mathbb{A}_{\rho C}$ System Controller Agency Matrix . . . . . p. 112

$\mathbb{A}_{\rho L}$ Service System Feasibility Matrix . . . . . p. 112

$\mathbb{A}_{C\rho}$ Controller System Agency Matrix . . . . . p. 112

$\mathbb{A}_{L\rho}$ System Service Feasibility Matrix . . . . . p. 112

$\mathbb{P}_H$ Projection matrix to select transportation degrees of freedom . . . . . p. 240

$\mathbb{P}_M$ Projection matrix to select transformation degrees of freedom . . . . . p. 240

$\mathbb{P}_S$ A (non-unique) projection matrix for the vectorized knowledge base . . . . . p. 83

$\overline{A}_Q$ Independent Controller Agency Matrix that shows jurisdiction of Q_I over R_P p. 89

Φ State transfer function dependent on $Q_{PN}[k]$ and $U[k]$) . . . . . p. 228

Φ_T State Transition Function for a timed Petri net . . . . . p. 99

$\Phi_{\mathcal{C}}$ Ac-CPN state transition function . . . . . p. 262

$\widehat{\mathcal{M}_\rho^+}$ Positive 3rd-order Device Model Refined Hetero-functional Incidence Tensor p. 266

$\widehat{\mathcal{M}_\rho^-}$ Negative 3rd-order Device Model Refined Hetero-functional Incidence Tensor p. 266

$\widehat{\Lambda}^+$ Vertical concatenation of the positive synchronization matrix over $l_i \in L$ . . . p. 270

$\widehat{\Lambda}_i^+$ Positive synchronization matrix for operand $i \in L$ . . . . . p. 267

$\widehat{\Lambda}^{-}$ Vertical concatenation of the negative synchronization matrix over $l_i \in L$ . . . *p. 270*

$\widehat{\Lambda}_i^{-}$ Negative synchronization matrix for operand $i \in L$ . . . . . *p. 267*

$\widetilde{\mathcal{M}}_\rho^{+}$ Projected Positive 3rd Order Hetero-functional Incidence Tensor . . . . . *p. 256*

$\widetilde{\mathcal{M}}_\rho^{-}$ Projected Negative 3rd Order Hetero-functional Incidence Tensor . . . . . *p. 256*

$\widetilde{\mathcal{M}}_\rho$ Projected 3rd Order Hetero-functional Incidence Tensor . . . . . *p. 256*

$\widetilde{\Lambda}_i^{+}$ Positive Service-Capability Feasibility Matrix . . . . . *p. 260*

$\widetilde{\Lambda}_i^{-}$ Negative Service-Capability Feasibility Matrix . . . . . *p. 260*

$\widetilde{\Lambda}_i$ Service-Capability Feasibility Matrix of size $\sigma(\mathcal{E}_{l_i}) \times \sigma(\mathcal{E}_S)$ . . . . . *p. 260*

$\widetilde{A}_\rho$ Hetero-functional Adjacency Matrix w/ eliminated row and column sparsity . *p. 83*

$\widetilde{M}_\rho$ Projected 2nd Order Hetero-functional Incidence Tensor . . . . . *p. 257*

$\widetilde{M}_\rho^{+}$ Projected Positive 2nd Order Hetero-functional Incidence Tensor . . . . . *p. 257*

$\widetilde{M}_\rho^{-}$ Projected Negative 2nd Order Hetero-functional Incidence Tensor . . . . . *p. 257*

Ξ Resource Aggregation Matrix . . . . . *p. 72*

A_C Controller Adjacency Matrix . . . . . *p. 94*

A_Q Controller Agency Matrix . . . . . *p. 89*

A_S System Concept . . . . . *p. 76*

A_ρ Hetero-functional Adjacency Matrix . . . . . *p. 80*

$A_{i,j}$ Matrix used for the product transformation feasibility matrix . . . . . *p. 237*

A_{QP} Equality Constraints Matrix of the HFNMCF Program . . . . . *p. 275*

B_{QP} Equality Constraints Vector of the HFNMCF Program . . . . . *p. 275*

C_C Capacity vector of size $\sigma(\mathcal{E}_S) \times 1 \in \mathbb{N}^{\sigma(\mathcal{E}_S)}$ . . . . . *p. 235*

C_U Capacity Constraints on the Engineering System Net Transitions . . . . . *p. 273*

C_{B1} Initial Conditions of the Engineering System Net Places . . . . . *p. 272*

C_{BK} Final Conditions of the Engineering System Net Places . . . . . *p. 272*

C_{Bn} Demand data constraint vector . . . . . *p. 272*

C_{Bp} Supply data constraint vector . . . . . *p. 272*

$C_{\mathcal{E}1}$ Initial Conditions of the Engineering System Net Transitions . . . . . p. 272

$C_{\mathcal{E}K}$ Final Conditions of the Engineering System Net Transitions . . . . . p. 272

C_{SL1} Initial Conditions of the System Service Net Places . . . . . p. 272

C_{SLK} Final Conditions of the System Service Net Places . . . . . p. 272

D_R^+ Positive Device Model Matrix . . . . . p. 266

D_R^- Negative Device Model Matrix . . . . . p. 266

D_{Bn} Transition selector matrix for the output transitions . . . . . p. 271

D_{Bp} Transition selector matrix for the input transitions . . . . . p. 271

D_{QP} Inequality Constraints Matrix of the HFNMCF Program . . . . . p. 275

DOF_ρ Sequence-Dependent Degrees of Freedom . . . . . p. 83

DOF_H Transportation Degrees of Freedom . . . . . p. 76

DOF_M Transformation Degrees of Freedom . . . . . p. 76

DOF_S Structural Degrees of Freedom . . . . . p. 76

$DOF_{HH\rho}$ Measure of Type IV Sequence-Dependent Production Degrees of Freedom p. 82

$DOF_{HM\rho}$ Measure of Type III Sequence-Dependent Production Degrees of Freedom p. 82

DOF_{LH} The number of transportation capabilities utilized by all services . . . . . p. 110

DOF_{LM} The number of transformation capabilities utilized by all services . . . . . p. 110

DOF_{LS} The number of capabilities utilized by all services . . . . . p. 110

$DOF_{MH\rho}$ Measure of Type II Sequence-Dependent Production Degrees of Freedom p. 82

$DOF_{MM\rho}$ Measure of Type I Sequence-Dependent Production Degrees of Freedom p. 82

e_i^n i^{th} elementary basis vector of predefined length n . . . . . p. 226

E_{QP} Inequality Constraints Vector of the HFNMCF Program . . . . . p. 275

F_{QP} Quadratic Cost Matrix of the HFNMCF Program . . . . . p. 274

f_{QP} Linear Cost Vector of the HFNMCF Program . . . . . p. 274

g_ψ Complex power assigned to the structural degree of freedom ψ . . . . . p. 238

I^n Identity Matrix of size $n \times n$ . . . . . p. 89

J_γ Holding Knowledge Base . . . . . p. 73

M_{PN} Petri net incidence matrix of size $\sigma(S) \times \sigma(\mathcal{E})$ . . . . . p. 228

M^{+}_{PN} Positive (ingoing) Petri net incidence matrix $(w(e_{wv}, b_y))$ . . . . . p. 228

M^{-}_{PN} Negative (outgoing) Petri net incidenceE matrix $(w(b_y, e_{wv}))$ . . . . . p. 228

$M_{\mathcal{E}_{\psi H}}$ Transportation degree of freedom incidence matrix . . . . . p. 240

N_{l_i} Service Net for product l_i . . . . . p. 98

$Q_{\mathcal{C}}$ Ac-CPN marking vector . . . . . p. 262

$Q_{\mathcal{E}}[k]$ Transition state marking vector of size $\sigma(\mathcal{E})\times 1$ . . . . . p. 229

$Q_{El_i}[k]$ Marking of Service Transitions for product l_i at time k . . . . . p. 99

Q_{l_i} Service net marking representing the set of service states . . . . . p. 98

$Q_{PN}[k]$ Marking (or discrete state) vector of size $\sigma(S)\times 1$ . . . . . p. 228

$Q_{PN}[k]$ Petri net marking vector, $Q_{PN}[k] = [Q_S[k]; Q_{\mathcal{E}}[k]]$ . . . . . p. 229

$Q_{Sl_i}[k]$ Marking of Service States for product l_i at time k . . . . . p. 99

Q_{SL} Vertical concatenation of the service net place markings for all $l_i \in L$ . . . . . p. 269

$Q_{\mathcal{E}L}$ Vertical concatenation of the service net transition markings for all $l_i \in L$ . . . p. 269

$U[k]$ Binary firing vector of size $\sigma(\mathcal{E})\times 1$ . . . . . p. 228

$U^{+}[k]$ Binary input firing vector of size $\sigma(\mathcal{E})\times 1$ . . . . . p. 229

$U^{+}_{l_i}[k]$ Binary Input Firing Vector for product l_i at time k . . . . . p. 99

$U^{-}[k]$ Binary output firing vector of size $\sigma(\mathcal{E})\times 1$ . . . . . p. 229

$U^{-}_{l_i}[k]$ Binary Output Firing Vector for product l_i at time k . . . . . p. 99

$U^{+}_{\mathcal{C}}[k]$ Ac-CPN Positive Firing Vector at time k . . . . . p. 262

$U^{-}_{\mathcal{C}}[k]$ Ac-CPN Negative Firing Vector at time k . . . . . p. 262

$U^{+}_{L}[k]$ Vertical concatenation of the service net positive firing vectors for all $l_i \in L$ p. 269

$U^{-}_{L}[k]$ Vertical concatenation of the service net negative firing vectors for all $l_i \in L$ p. 269

$y_{\psi h}$ An admittance assigned to the transportation structural degree of freedom . . p. 238

y^{*}_{ψ} An admittance assigned to the structural degree of freedom ψ . . . . . p. 238

z_{l_i} Sequence of Service Activities $e_{xl_i} \forall x \in [1, \ldots, \sigma(\mathcal{E}_{l_i})]$ . . . . . p. 97

$\mathcal{F}_v$ Resource Flexibility . . . . . p. 72

$\mathcal{N}$ Marked Petri Net (Graph) . . . . . p. 228

$\mathcal{N}_{\mathcal{C}}$ Arc-Constant Colored Petri Net (ac-CPN) . . . . . p. 262

$\mathcal{R}_w$ Process Redundancy . . . . . p. 72

$\mathcal{S}$ Scheduled Event List . . . . . p. 229

$\mathcal{U}[k]$ Binary product firing matrix at time step k of size $\sigma(\mathcal{E}_S) \times \sigma(L)$ . . . . . p. 237

$\mathcal{Y}$ Transportation degree of freedom admittance matrix . . . . . p. 238

Mathematical Operators

$()^V$ Shorthand for vectorization (i.e. vec()) . . . . . p. 81

$\cdot$ Hadamard product . . . . . p. 73

$\circ$ Negative 3$^{\text{rd}}$-order Outer Product . . . . . p. 266

$\circledast$ Matrix Aggregation Operator . . . . . p. 72

$\langle A, B \rangle_F$ Frobenius Product of matrices A and B . . . . . p. 76

$\odot$ Matrix boolean multiplication . . . . . p. 71

$\ominus$ Matrix boolean subtraction . . . . . p. 76

$\otimes$ Kronecker Product . . . . . p. 73

$\sigma()$ The size of the set (). . . . . p. 65

$\times$ Cartesian Product . . . . . p. 67

$cd()$ Ac-CPN Color Domain Mapping . . . . . p. 262

$\mathcal{F}_M$ Matricization function [52]. . . . . p. 257

Chapter 1

Introduction and Purpose

1.1 Context

In the context of 21^{st} century grand challenges, the field of engineering systems has emerged at the intersection of engineering, management, and the social sciences. Over the past decades, engineering solutions have evolved from engineering artifacts that have a single function, to systems of artifacts that deliver a specific service, and finally, to engineering systems that deliver multiple services within a societal and economic context. In order to understand engineering systems, a holistic approach is required that assesses their impact beyond technical performance. Engineering systems are defined as:

Definition 1.1: Engineering System [1]: A class of systems characterized by a high degree of technical complexity, social intricacy, and elaborate processes, aimed at fulfilling important functions in society. ■

Furthermore, there are a number of characteristics that distinguish engineering systems from other systems. Engineering systems...

- ... exist in the real world. They always have physical components, but are also likely to contain informational components.
- ... are artificial. Engineering systems are man-made, but often integrate into the

natural world.

- ... have dynamic properties. Engineering systems change over time, and have a sense of temporality.
- ... have a hybrid state. The states of engineering systems are usually both discrete and continuous.
- ... contain some human control.

A prime example of an engineering system is the New York City rail system: First, the metro system is a physical system that transports people throughout the city. Second, the rail system is clearly man-made and thus artificial. Third, the system is dynamic. Trains move through the system and the system itself also changes as a result of failure or planned maintenance. Fourth, the metro system incorporates both discrete and continuous dynamics. For example, discrete states describe signals, that are either red or green. The continuous dynamics describe the travel of a train through the system. The position of the train is a continuous state as the train drives from one station to the next. Finally, the system is controlled by a combination of centralized and distributed control, where the central control room decides on routing and signaling, but the trains also have drivers that have direct control over the train in the system.

When this example is placed within its larger context, the city of New York, it becomes clear that the rail system is dependent on other urban infrastructure systems to function. The rail system relies on the electric power grid for propulsion, on the pedestrian infrastructure for its passengers, and on the IT infrastructure for controls, monitoring, and passenger convenience. These infrastructures serve more than one purpose, the electric power grid also delivers electric power to homes, the pedestrian infrastructure is also used to walk to other places than a metro station, and the IT infrastructure may be shared with emergency services. It now becomes clear that as cities become more crowded, their infrastructures need to fulfill more services. Systems theory [53] and design theory [3] recognize that a large number

of functional requirements imposed on a confined space (the city) inherently couples the elements of the systems. Consequently, the infrastructures become interdependent [54–62].

The combination of the characteristics of engineering systems and the interdependence of such systems raises timely and poignant questions about life-cycle properties, such as resilience and sustainability [1, 63, 64]. For example, there is a growing recognition that in order to achieve long term sustainability through deep decarbonization, cities will have to invest heavily in electrified transportation [31, 32, 50, 65]. And yet, the experience of hurricane Sandy reminded us that the dependence of the rail system on the electric power grid reduced the resilience of the transportation system: It was impossible to evacuate a city as large as New York City without the electrified rail system [66]. In other words, whereas electrified transportation can serve to bring about greater sustainability, it may indeed hinder a cities' resilient response to natural disasters. In conclusion, engineering systems need to balance trade-offs of operating efficiency with sustainability and resilience (and other life-cycle properties). The engineering systems literature has developed the *"ilities"* (or *life-cycle properties*) as properties that are defined beyond the core function of the systems and provide support so as to understand the complex set of requirements surrounding these engineering systems. The *"ilities"* are defined as:

Definition 1.2: ilities [1]: The ilities are desired properties of systems, such as flexibility or maintainability (usually but not always ending in "ility"), that often manifest themselves after a system has been put to its initial use. These properties are not the primary functional requirements of a system's performance, but typically concern wider system impacts with respect to time and stakeholders than are embodied in those primary functional requirements. The ilities do not include factors that are always present, including size and weight (even if these are described using a word that ends in "ility"). ■

1.2 Problem Identification

It is clear that engineering systems are more complex than the systems of old (mere combinations of artifacts). These engineering systems require a novel way to be measured, managed, and optimized. For the analysis of these systems, experimentation is nearly impossible as changes are invasive, potentially disruptive to the services they deliver, and costly. A data-based approach to analyzing engineering systems may seem to be a better option, but as engineering systems require structural interventions, the predictive power of data-based approaches is limited. A model-based analysis of engineering systems requires a deep knowledge of the system and enables the study of structural and behavioral interventions at a low cost without having to disrupt a functioning system. This work concentrates on the advancement of a modeling framework to enable the analysis of engineering systems. Such a modeling framework is required to accommodate both the extreme heterogeneity of engineering systems as well as their quantitative nature to enable study of engineering system structure, dynamic behavior, and optimal control.

1.3 Existing Solutions and Their Shortcomings

Though the field of engineering systems has been defined within the past decade, the modeling frameworks stem from a variety of other disciplines, including systems engineering, mathematics, and product design [48,67]. The dominant modeling framework is the Systems Modeling Language (SysML) that leverages intuitive graphical models to represent a wide variety of heterogeneous systems [68–70]. The major limitation of SysML is its lack of support for quantitative analysis of the system.

On the other hand, quantitative modeling approaches often limit the heterogeneity of the modeled systems. Many measures of resilience and sustainability rely on a graph theoretic framework [71–89]. In such cases, the graph defined as a tuple of identical nodes and identical edges that connect them, is well-suited to the study of a single homogeneous

engineering system [88–92]. However, the assumption of a set of identical nodes and edges is ill-suited to an interdependent engineering system composed of multiple systems of fundamentally different functions [5, 93].

The field of multilayer networks has been founded in an effort to enhance the modeling power of graph theory by expanding the definition of the modeling elements in a graph [94, 95]. Over the past decade, numerous methods have tried to provide a consistent approach to model these networks-of-networks. However, as discussed by Kivelä et al, all these multilayer network methods have their respective modeling limitations [95].

1.4 Proposed Solution

In contrast to the aforementioned modeling frameworks, Hetero-functional Graph Theory has emerged over the past decade to be the first quantitative structural modeling framework that captures all five parts of system structure (i.e. the system boundary, the formal elements of the system, the connections between them, the functional elements of the system, and their allocation to the formal elements) [5]. It enables the structural modeling of a heterogeneous large flexible engineering system and explicitly accommodates all five types of system processes (i.e. Transform, Transport, Store, Exchange, and Control) and all five types of operands (i.e. Living Organisms, Matter, Energy, Information, and Money) that regularly appear in engineering systems [1]. Furthermore, Hetero-functional Graph Theory has been used as the underlying structure for structural and dynamic system models across many different application domains including manufacturing systems [2, 6–22], multi-modal transportation systems [23–25], electric power systems [26, 27, 27], multi-modal electric transportation systems [30–33], microgrid-enabled production systems [34, 35], personalized healthcare delivery systems [36–45].

The development of Hetero-functional Graph Theory, however, has been distributed over many works and has not yet been consolidated in a single work. This thesis thus chooses

to advance a Hetero-functional Graph Theory. The thesis aims to root the mathematical concepts of Hetero-functional Graph Theory in the engineering systems and systems engineering literature, it aims to provide a single consistent overview of the theory, it aims to explore new application domains for dynamic models based on Hetero-functional Graph Theory, and it aims to develop a framework to optimize such dynamic models.

1.5 Thesis Statement and Research Questions

In order to address the challenges in the literature as identified above, this thesis aims to fulfill the following Thesis Statement:

Thesis Statement: *A Hetero-functional Graph Theory provides a novel approach to modeling the structure of large flexible engineering systems such that it enables simulation and optimization of the behavior of such systems.*

This thesis statement is fulfilled through answers to four central research questions:

- **Research Question 1:**

  Why are existing modeling frameworks for engineering system structure inadequate?

  The first research question seeks to provide rationale and evidence to support the need for a new modeling framework to describe engineering system structure.

- **Research Question 2:**

  What is an ontologically clear structural model of a hetero-functional engineering system?

  The second research question seeks to provide a novel, internally consistent structural modeling framework for hetero-functional engineering systems. Furthermore, to answer this research question, a demonstration of such a modeling framework is required.

- **Research Question 3:**

  How can a Hetero-functional Graph Theory structural model be revised with dynamic behavior so as to simulate an engineering system?

  The third research question seeks to provide a dynamic model based on the Hetero-functional Graph Theory structural modeling framework. This dynamic model then demonstrates the dynamic behavior of an engineering system through simulation.

- **Research Question 4:**

  How can a Hetero-functional Graph Theory dynamic model be optimized?

  The final research question seeks to provide an optimization program based on the principles of the dynamic simulation environment as developed for Research Question 3.

1.6 Outline

The thesis consists of seven chapters that, together, aim to seek answers to the thesis statement and research questions listed above. Each of these chapters (except Chapter 6) relies heavily on previously published work. The outline below serves to provide the high-level thread of this thesis.

- **Chapter 2: Background:** consists of three sections that provide the reader with *1)* context to the methods of analysis in the field of engineering systems, *2)* detail regarding the shortcomings of existing modeling methods, and *3)* prerequisite information that support the introduction and advancement of Hetero-functional Graph Theory in the remainder of the thesis. The work in this chapter is drawn from three publications: 1. Book Chapter 23 entitled: *"Evaluating Engineering Systems Interventions"*, in the book *"Handbook of Engineering System Design"* [48], 2. Book Chapter 2, called *"The Need for Hetero-functional Graph Theory"*, in the book *"A Hetero-func-*

tional Graph Theory for Modeling Interdependent Smart City Infrastructure" [5], and 3. Book Chapter 3, called *"Hetero-functional Graph Theory Preliminaries"*, in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5].

- **Chapter 3: Hetero-functional Graph Theory:** provides the first complete overview of hetero-functional graph theory. It present the seven mathematical models in hetero-functional graph theory and establishes their associations to SysML diagrams. The work in this chapter is drawn from Book Chapter 4, called *"Hetero-functional Graph Theory"*, in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5].

- **Chapter 4: Modeling Interdependent Smart City Infrastructures with Hetero-functional Graph Theory:** applies a hetero-functional graph theory to two test cases to demonstrate how the theory can be used, and that the theory does not impose the ontological and modeling constraints imposed by other quantitative structural models. The test cases are instantiations of interdependent smart city infrastructure systems. The work in this chapter is drawn from Book Chapter 5, called *"Modeling Interdependent Smart City Infrastructure Systems with HFGT"*, and Appendix A, called *"Representing a Four Layer Network in Hetero-functional Graph Theory"*, in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5].

- **Chapter 5: Dynamic Systems Modeling with Hetero-functional Graph Theory:** demonstrates how a hetero-functional graph theory structural model supports the development of a dynamic system model. The chapter develops a dynamic model of a microgrid-enabled production system and describes the hybrid-dynamic production system and microgrid dynamics holistically. The work in this chapter is drawn from a 2017 journal article entitled *"A Dynamic Model for the Energy Management of*

Microgrid-Enabled Production Systems" in the *Journal of Cleaner Production* [35].

- **Chapter 6: Optimization of Dynamic Systems with Hetero-functional Graph Theory:** aims to develop an optimization program for a dynamic, hetero-functional graph theory-based model of an engineering system. It establishes a connection to the Petri net literature and demonstrates the theoretical advancement through a hydrogen-natural gas infrastructure system test case. The work in this chapter is to be published.

- **Chapter 7: Conclusion:** provides the main conclusions for each of the research questions posed above, an overview of the contributions of this thesis to the engineering systems and hetero-functional graph theory literature, and a discussion of limitations of and future work based on the research in this dissertation

Chapter 2

Background

Chapter Abstract:

This chapter develops the context and preliminary information for the thesis in three parts:

1. Section 2.1 (Page 11): Background to Engineering Systems Analysis,

2. Section 2.2 (Page 26): Literature Gap, and

3. Section 2.3 (Page 43): Hetero-functional Graph Theory Preliminaries.

Section 2.1 develops an overview and comparison of three high-level engineering systems evaluation approaches: 1. experimental approach, 2. data-driven approach, and 3. model-based approach. The section concludes that the model-based approach is especially valuable for engineering systems analysis. This section is directly adopted from Book Chapter 23 entitled: *"Evaluating Engineering Systems Interventions"*, in the book *"Handbook of Engineering System Design"* [48].

Section 2.2 contains two main topics: (1) model-based engineering systems analysis and (2) the need for hetero-functional graph theory. First, Subsection 2.2.1 focuses on model-based engineering systems analysis approaches and their associated pros and cons. It establishes that Hetero-functional Graph Theory is especially well-suited to describe engineering system structure. This subsection is directly adopted from Book Chapter

23 entitled: *"Evaluating Engineering Systems Interventions"*, in the book *"Handbook of Engineering System Design"* [48]. Second, Subsection 2.2.2 details the need for the advancement of a novel quantitative structural modeling approach to enable engineering systems analysis, called *Hetero-functional Graph Theory*. This discussion is directly adopted from Chapter 2, called *"The Need for Hetero-functional Graph Theory"*, in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5].

Finally, Section 2.3 provides an overview of the preliminary theory to support the advancement of Hetero-functional Graph Theory and its applications in the remainder of this thesis. This foundation draws upon the theory of ontologies and the field of systems engineering and engineering systems. This section is directly adopted from Chapter 3, called *"Hetero-functional Graph Theory Preliminaries"*, in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5].

2.1 Background to Engineering Systems Analysis

The first section in this chapter provides a background to the analysis of (engineering) systems in three parts. Subsection 2.1.1 defines a common framework for describing systems and using systems thinking abstractions their associated intervention types. Subsection 2.1.2 discusses measurement theory and its application to engineering systems. Subsection 2.1.3 compares different types of evaluation methods for engineering systems analysis.

2.1.1 Describing Systems and Interventions

2.1.1.1 Describing Systems

Engineering Systems, also referred to as socio-technical systems, are complex systems at the intersection of physics, management, and social sciences [1]. The evaluation of engineering

system interventions relies on accurate and consistent measurement of the system. As shown in Figure 2.1, this chapter adopts the approach of many STEM disciplines where systems are mathematically described as a system of differential algebraic equations (DAEs) that define the relationship between the inputs u and the outputs z [96, 97]. The system is also said to have states x, algebraic states y, and parameters λ. The vector functions $f(\cdot)$, $g(\cdot)$, and $h(\cdot)$ are differential equations, algebraic equations, and output equations respectively. While a more complex model based upon the hybrid dynamic systems literature is possible, a system of differential algebraic equations serves the purposes of this discussion.

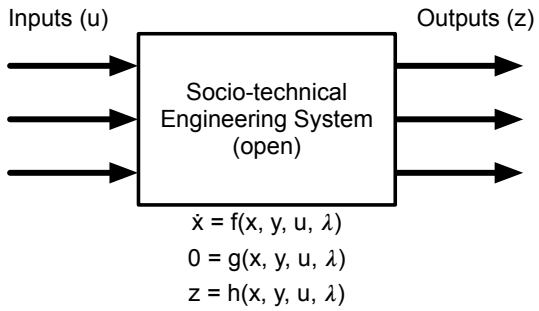


Fig. 2.1: A mathematical and graphical representation of an arbitrary engineering system.

In addition to the above description, this chapter requires the introduction of four systems thinking abstractions: (1) system context, (2) system function, (3) system form, and (4) system concept [1, 98]. These abstractions support the classification of intervention types and their accompanying evaluation methods.

2.1.1.1.1 System Context The system context is the set of interrelated conditions in which the system exists or occurs [99]. Sometimes, it is also referred to as the system environment: *"All that is external to the system"* [100]. The field of Engineering Systems emphasizes that the system does not operate in a vacuum, but is instead solidly placed in its context. When an intervention is evaluated, the impact of a system on its context is critical to truly understand the system's performance. Sometimes, system outputs are neglected with severe consequences (e.g. climate change). Naturally, the context also influences the

system itself and often, it determines the success of the intervention.

2.1.1.1.2 System Behavior System behavior is the response of system outputs to a change in system inputs or parameters. It reflects the processes, or function of the system: "what the system does." The system inputs are predominantly a result of the system context, whereas the parameters are internal to the system. In the context of *Engineering Systems*, the system behavior consists of the behavior of the engineering artifacts and the humans that interact with the system.

2.1.1.1.3 System Form System form is the description of a system's component elements and their relationships. The system structure also defines the presence (but not values) of system states x, algebraic states y, parameters λ, inputs u, and outputs z. By adding or removing elements to/from the system, the number of equations in the vector functions $f(\cdot)$, $g(\cdot)$, and $h(\cdot)$ changes.

2.1.1.1.4 System Concept The description of the system as a whole relies on the combination of the system behavior, and the system structure. System concept is the mapping of system function onto system form (also called the allocated architecture [67]). Consequently, system concept can be represented by a system of equations. The behavior of the system results from the coupled equations.

2.1.1.2 Describing Interventions

For the purposes of this chapter, *"intervention"* is defined as:

Definition 2.1: Intervention: [99] The act of interfering with the outcome or course especially of a condition or process (as to prevent harm or improve functioning). ■

In the context of engineering systems, interventions intend to change the system so as to improve the outcome of the engineering system. Two types of interventions are recognized: behavioral and structural.

2.1.1.2.1 Behavioral Interventions Behavioral interventions aim to change the outcomes of a system by adjusting the values of the system inputs and system parameters while the structure of the system is untouched. As a result, behavioral interventions are often relatively affordable. Decisions to change the operating procedure or policies around a system may take a long time and are sometimes hard to implement, but the upfront capital investment is limited because no fundamental changes in the system are necessary.

An example of an intervention based on system inputs is a policy change that increases the ethanol percentage in gasoline. When a different ethanol/gasoline mixture enters the system, the emissions of the transportation system will change as a consequence.

An example of an intervention based on system parameters is the reduction of ticket prices in a public transit system. Ticket prices are internal to the engineering system and are set as a result of a policy decision. As a consequence of this parameter change, the total public transit ridership may increase/decrease, with cascading impacts such as: less/more traffic, less/more emissions, etc.

2.1.1.2.2 Structural Interventions Structural interventions aim to change the structure of the system, which consists of the systems' parts and their relationships. These changes are often physical and require large upfront capital investments. Furthermore, structural interventions require a revision of the operating procedures and policy around the system, since the policies of the old system may no longer apply.

An example of adding elements to the system is the addition of a road in a town. This road adds an "equation" and a "state." The *equation* describes the flow of traffic on the road as a result of the *state*, the number of vehicles on the road. The addition of a road is a structural intervention and it may also lead to a revision of the local traffic ordinances and the behavior of vehicles on the road. For example, at the connecting intersections a new speed limit may be introduced to reduce risk for turning vehicles.

An example of adding *variables* is the consideration of electric vehicles in parking lot

design. Electric vehicles require charging facilities in the parking lot, which changes the calculation of the required parking spots in building code.

2.1.2 Requirements for Evaluating Interventions

This section discusses measurement as a foundation for the evaluation of engineering system interventions. Interventions aim to improve the existing engineering system. Consequently, the evaluation of interventions requires a comparison of (at least) the current system and the system with the intervention. Such a comparison requires the definition of a common mathematical framework (or standardizing space) to describe both systems. The process of first defining this framework and then describing the systems within the framework is called *"measuring."*

This section first discusses the fundamentals of measurement including an overview of the generic measurement process, measurement scales, and different measurement strategies. The second part of this section then discusses different approaches to measurement, and specifically, the differences between measuring a technical system and an engineering system. Based on this foundation in the measurement of engineering systems, Section 2.1.3 discusses the evaluation methods for engineering system interventions.

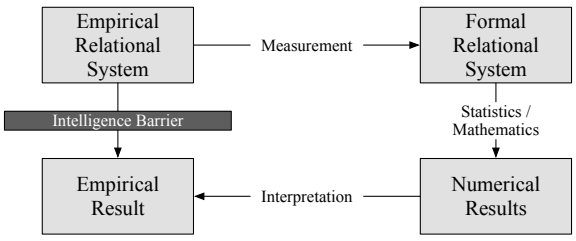


Fig. 2.2: A Generic Measurement Process [2]

2.1.2.1 Measurement Fundamentals

The measurement of engineering systems is critical for informed decision-making. Without an accurate and consistent approach to measuring the empirical system, the foundation for the

decision-making process is flawed. As shown in Figure 2.2, without measurement, the real world presents us with an empirical system that exhibits certain phenomena called empirical results. These results can be viewed as qualitative or anecdotal evidence. The link between the empirical system and its empirical results is often not well-understood and prevents effective decision-making. Instead of a direct translation, the empirical system should be first *measured* so that real-world phenomena are assigned their associated numerical values in a formal (mathematical) system. Mathematics, and statistics more specifically, are then used to determine numerical results in the formal system. These are, in turn interpreted, to become empirical results. The empirical and formal systems must possess methods by which their respective objects can be *related* and ultimately compared.

More specifically, the **Empirical Relational System** contains a nonempty set of empirical objects that are to be measured, with relations between and closed binary operations on the empirical objects. Note that these relations are *independent* of the measure function. The **Formal Relational System** is a nonempty set of formal objects with relations between and closed binary operations on the formal objects.

Definition 2.2: Measurement [101, 102]: "Measurement is the process of empirical, objective assignment of symbols to attributes of objects and events of the real world, in such a way as to represent them, or to describe them." – Finkelstein, 1982 ■

Measurement consists of three elements: (1) a set of measurables, (2) a standardizing space, and (3) a measure function. The set of measurables is defined as a set of objects with a specific attribute type. The standardizing space is a basic construct to which all the measurements can be compared. Finally, the measure function performs the empirical and objective assignment as mentioned in the definition of measurement. A consistent measure function ensures a consistent measurement of empirical relational systems to formal relational systems. If two empirical systems have been translated to formal systems with the same measure function, the formal systems can be compared rather than their respective empirical systems.

Table 2.1: Classification of measurement scales [2]

Scale Type	Applicable Statistics	Example
Nominal	Non-parametric	Football player uniform numbers
Ordinal	Rank Order & above	IQ
Interval	Arithmetic Mean & above	Celsius Scale
Ratio	Percentage & above	Kelvin Scale
Absolute	Additivity & above	Counting

Definition 2.3: Measure [2]: A measure (or measure function) is a one-to-one function that acts on a set of (empirical) objects and returns a formal object. ■

Note that often the term “measure" and “metric" are confused. Metric, however, is defined as:

Definition 2.4: Metric [103]: A metric, also called a distance function, defines the distance between a pair of elements in a set. ■

Not all empirical relational system can be measured in the same way. For example, human behavior and a block of iron do not have the same attributes. The type of empirical system, with its related attributes, determines the type of measurement scales that can be used to measure the system. This impacts the type of numerical results downstream in the measurement process, because not all mathematics and statistics can be used for all measurement scales. The scale types, with applicable statistics and examples are presented in Table 2.1. Engineering systems inherently combine physics-based systems with human behavior and economics. Consequently, the measurement of the engineering system requires a combination of the measurement scales.

From a practical perspective, there are two measurement strategies: (1) direct measurement and (2) indirect measurement. Direct measurement is applied when the desired property is both “simple" and an “output" of the system. As a result, the property is easily accessible and there are often sensors that directly convert the desired property into a numerical result. Examples of fundamental measures are length, time, voltage, and current. However, these properties are rare, especially for engineering systems. Indirect measurement applies to

properties that are not fundamental. The measurement of such indirect properties requires a formal model that integrates fundamental properties (these are considered “internal" to the system). The formal model is considered the standardizing space and mathematics and statistics are applied to this model to extract the desired numerical results.

2.1.2.2 Engineering System Measurement

During the past century, engineering solutions have evolved from engineering artifacts to engineering systems. Consequently, the solution requirements have changed. Instead of merely “functioning" artifacts that performed their (singular) task, engineering systems perform many services composed of separate tasks. Furthermore, engineering systems include non-technical elements, such as humans. It is, therefore, essential to evaluate engineering systems beyond their technical aspects and include impacts of the system on its environment.

This section describes Engineering System measurement with a tiered approach. Engineering Systems are evaluated at several levels of granularity. First, the fundamental artifacts are evaluated based on the performance of their specific task with Technical Performance Measures (TPMs). Then, the combination of these artifacts provides a service. The performance of these services is measured with Measures of Performance (MOPs). The first two types of measures, however, do not truly address the socio-technical nature of Engineering Systems. Therefore finally, Measures of Effectiveness (MOEs) were developed at the highest level of granularity for Engineering Systems. These consist of multiple services and socio-technical interfaces. For the Engineering Systems literature, a subset of these measures is especially important; *life cycle properties* or *ilities*.

Definition 2.5: Technical Performance Measures [104]: “TPMs measure attributes of a system element to determine how well a system or system element is satisfying or expected to satisfy a technical requirement or goal." ■

Definition 2.6: Measure of Performance [104]: “The measures that characterize physical

or functional attributes relating to the system operation, measured or estimated under specified testing and/or operational environment conditions." ■

Definition 2.7: Measure of Effectiveness [104]: "The operational measures of success that are closely related to the achievement of the mission or operational objective being evaluated, in the intended operational environment under a specified set of conditions; i.e., how well the solution achieves the intended purpose." ■

Overall operational success criteria (Measures of Effectiveness) include: Mission performance, safety, operability, operational availability, etc. These measures of effectiveness are often a quantitative means of measuring a degree of adherence to requirements.

Finally, in the context of engineering systems, life cycle properties or *ilities* need to be addressed as a subset of the MOEs. The definition of ilities is:

Definition 2.8: "ilities" [1] "The ilities are desired properties of systems, such as flexibility or maintainability (usually but not always ending in "ility"), that often manifest themselves after a system has been put to its initial use. These properties are not the primary functional requirements of a system's performance, but typically concern wider system impacts with respect to time and stakeholders than are embodied in those primary functional requirements. The ilities do not include factors that are always present, including size and weight (even if these are described using a word that ends in "ility")." ■

The measurement of engineering system interventions often relies on a large number of measures that aim to capture the full impact of the intervention. It thus becomes challenging to weigh all the trade-offs appropriately when comparing intervention options. In these situations, the measures are often summarized or combined through Weighted Objective Methods (WOM). A WOM aims to reduce the complexity by, first, assigning a value to the performance of the measures in the analysis and then manipulating (adding, averaging, etc.) these values to combine them into a single score for the intervention option. When executed thoughtfully, this approach can provide a valuable, high-level overview of the intervention options.

The downside of a WOM is that it relies on the *combination* of measurement functions for many different measures. These measurement functions generally do not have the same scale. Consequently, the combination of the results of the individual measures into a single measure may be flawed. This can be illustrated as follows: for each measure, the WOM may rank the intervention options from best to worst. The most resilient intervention receives rank 1 on the measure “resiliency", the least resilient intervention receives rank 5. When these ranks are combined in a WOM by averaging the rank of an intervention over all the measures, the statistics of the ordinal scale are violated. The ordinal scale, as mentioned in Table 2.1, does not allow for additive statistics. Therefore, the WOM that ranks the interventions on the measures and then takes the “average rank," uses flawed statistics.

This section described (1) how to measure and (2) what to measure in engineering systems. The former was described through the process of measurement and the latter was described through three categories of engineering system measures in increasing scale. The chapter now builds on this knowledge to compare evaluation methods for Engineering Systems.

2.1.3 Comparing Evaluation Methods

This section discusses the different types of evaluation methods for engineering system interventions. As discussed in Section 2.1.1, Figure 2.1, engineering systems create a relationship between inputs and outputs. The interventions aim to improve the outputs of the system, given a set of inputs. The goal of the evaluation methods is to predict how an intervention changes the outcome of the engineering system. Generally, the relationship between inputs and outputs of systems have been studied using one (or a combination) of three approaches: the 1. experimental, 2. data-driven, and 3. model-based approach.

The experimental approach originated with science. The experimental approach tests a hypothetical relationship between inputs and outputs through a set of experiments in which either the input or the system is changed [105]. The experimental approach is generally

performed in a controlled environment (context).

With the rise of widely available (historical) data on engineering systems, the data-based approach became viable. The data-based approach leverages existing data to derive a relationship between inputs and outputs of the system [106].

Finally, if all the parts of the engineering system are well-understood, a theoretical model can be built to explain the relationship between system inputs and outputs [1]. The model-based approach integrates models of the elemental parts of the system to define a single overarching model for the system as a whole. This overarching model couples system inputs to system outputs.

These three approaches are not mutually exclusive and are often combined to grasp the full complexity of engineering systems. Each of these evaluation methods has been adopted across academia and industry. Note that all approaches can be used to study interventions both qualitatively and quantitatively. The measurement scale depends on the type of intervention and the desired analyses that support the interpretation of the results. Each of the three previously introduced approaches are now discussed in greater detail. Section 2.1.3.1 covers the experimental approach, Section 2.1.3.2 covers the data-based approach, and Section 2.1.3.3 covers the model-based approach.

2.1.3.1 Experimental Approach

The experimental approach for the evaluation of engineering system interventions relies on the comparison of two sets of empirical results: before and after the intervention. The main benefit of this approach is that the results are real. As long as the measurement process is kept constant for measurements before and after the intervention, the empirical results reflect a change in the objects of the empirical relational system (or real world) [107]. Furthermore, the results from such an experimental approach hold for both behavioral and structural interventions. Note that experiments are also valuable to study specific pieces of engineering systems with small scale experiments, often in a well-controlled environment.

The experimental approach, however, has numerous disadvantages. Engineering systems are generally large, critical systems intertwined with the daily routine of the population [108]. Experimenting with these systems to find out which approach works best, potentially rebuilding systems multiple times, is a tremendous waste of money [105]. Furthermore, the execution of such an experiment is time consuming and potentially reckless. The experimental approach should therefore be used sparingly and mainly to inform the planning of future interventions (e.g. as in the case of pilot-projects) [109]. The value to provide "lessons learned" to future interventions should not be overestimated. Another downside of the experimental approach is that it is a black box model. The system as a whole is overhauled, but it may be unclear how external factors have changed between the time of the baseline and post-implementation measurement.

2.1.3.2 Data-Driven Approach

The data driven approach to the evaluation of engineering system interventions relies on the definition of a statistics-based formal relational model between inputs and outputs. This model can be used to evaluate a behavioral intervention by estimating the response of the system to changing inputs. Generally, there are six distinct types of data analysis [110]:

Descriptive data analysis aims to describe the data without interpretation [111]. The most commonly used statistics in quantitative descriptive analyses are the sample mean and the sample standard deviation. A summary statistic for nominal measurements is a frequency analysis.

Exploratory data analysis provides a description and interpretation of the data aimed at providing insight into a problem [112]. The goal of exploratory data analysis is to find the "story" of the data, detect patterns and trends, and inform deeper study of the data. Some of the most common techniques include graphical representation of the data with boxplots, dotplots, or kernel density functions. Exploratory data analysis can also include preliminary model building and subset analyses.

Inferential data analysis aims to provide general facts about a certain type of systems given a limited amount of data [113]. It quantifies the correlation between measurements to provide insight to the generalizability of the data patterns. The two major branches in inferential data analysis are estimation and hypothesis testing. The former contains the main methods of point estimation and interval estimation. The latter contains a wide range of tests appropriate for different types of analyses. A non-exhaustive list of hypothesis tests is provided below [114]: 1. t-Test for independent means, 2. t-Test for Correlation Coefficients, 3. One-way ANOVA, 4. Analysis of Covariance, 5. Two-way ANOVA, 6. One-way repeated Measures ANOVA, 7. t-Test for Regression Coefficients, 8. Chi-Square for Contingency Tables.

Predictive data analysis uses measurements of a subset of the data to predict the measurement on a single person or unit in the remainder of the data. The algorithms in this field are evolving quickly and are often classified into supervised learning and unsupervised learning. Supervised learning aims to learn a function that couples inputs to outputs from data that contains both inputs and outputs. A non-exhaustive list of supervised learning algorithms is [115]: 1. Support Vector Machines, 2. Neural nets, 3. Logistic Regression, 4. Naive bayes, 5. Memory-based learning, 6. Random forests, 7. decision trees, 8. bagged trees, 9. boosted stumps. Unsupervised learning is predictive data analysis without pre-identified output or feedback. Some typical unsupervised learning examples are [116]: 1. Clustering, 2. Association rules, 3. Self-organizing maps.

The final two methods, **Causal data analysis** and **Mechanistic data analysis** rely on a theoretical understanding of the measured system and are used in conjunction with model-based evaluation approaches. Causal data analysis derives an average effect of one measurement on another, whereas Mechanistic data analysis aims to determine the relationship between two measurements under all conditions.

All analyses can be used to inform the design of the intervention. However, for the definition of the formal relational model that “predicts" the relationship between inputs

and outputs after the intervention, only the last three types are appropriate (predictive, causal, and mechanistic data analysis). Note that the statistical model requires data beyond historical data of the original system [106]; for example from other systems comparable to the post-intervention system.

The benefits of a data-driven approach are that it is both cheap and quick. Data availability has soared and the cost of collecting and storing data has plummeted. In combination with rapidly evolving computational resources that can analyze the data, the creation of a data based model has become very affordable. Furthermore, the rise of cloud computing enable extremely fast analysis of the data.

The downsides of the data-driven approach are related to the fact that statistical models are a black box [106, 117]. As a result, it is impossible to truly understand the elemental dynamics that define the overall system behavior. This is especially true for more advanced and automated statistical models based on neural networks and deep learning [118]. As a result of the opaque nature of the model, the study of structural interventions is not possible. The model loses its generalizability when the basic equations (or assumptions) are changed. Finally, the data-driven models rely on the assumption that the system is stationary. In order to analyze interventions that break the “business-as-usual" case, data-based approaches to intervention evaluation are insufficient.

In conclusion, data driven models are predominantly appropriate to analyze behavioral interventions in systems where the “mechanistic" science is not fundamentally understood. However, the analysis of structural interventions, or interventions that break the assumption of “business-as-usual," cannot be performed with data driven evaluation approaches.

2.1.3.3 Model-Based Approach

The model-based approach to the evaluation of engineering system interventions relies on the construction of a formal relational system based on knowledge of the empirical system [49]. The formal relational system is constructed to represent the dynamics of each

of the elements in the empirical relational system. The combination of each of the elemental models creates full system results that match the observed numerical results as derived from the measurement of the real-world system. The intervention is evaluated by implementing new or changed elemental models in the formal relational system. The empirical results interpret the numerical results of the two formal relational systems. Section 2.2 provides a closer look at the different model-based approaches to evaluating engineering system interventions.

The main benefit of the model-based approach is its transparency [5]. The elements in the models are known and have individual properties. The properties may include first principle-based dynamics. Furthermore, the model-based approach supports the evaluation of both behavioral and structural interventions. The model elements may be adjusted in their behavior, or be changed altogether.

The main downside of the model-based approach is that a deep knowledge of the engineering system is required to build a model that matches the real-world measurements [5].

In conclusion, model-based intervention evaluation is specifically valuable when used to represent a system that is well-known. It provides a transparent approach to the evaluation of both structural and behavioral interventions. In recent years, a discussion around the “end of theory" has emerged. The chapter addresses this discussion explicitly in the next section (Section 2.2), together with an in-depth discussion of the model-based intervention evaluation methods.

2.1.4 Summary

Subsection 2.1.1 on Page 11 provided background on the description of systems and interventions. The section first introduced a common framework for describing systems using four systems thinking abstractions: (1) system context, (2) system behavior, (3) system form, and (4) system concept. This framework lead to the distinction of two types of interventions: (1)

behavioral and (2) structural. The distinction is important for the evaluation of interventions, because it ensures that the intervention is evaluated with the appropriate method.

Subsection 2.1.2 on Page 15 introduced a discussion of the fundamentals of measurement, including an overview of the measurement process, measurement scales, and different measurement strategies. The section furthermore differentiated between the measurement of a technical system and and engineering system.

Subsection 2.1.3 on Page 20 discussed a comparison of evaluation methods. The section discussed three central approaches to the evaluation of engineering system interventions: the (1) experimental, (2) data driven, and (3) model-based approach. Each of these approaches have their strengths and weaknesses. However, these methods can also be combined to leverage their strengths where appropriate. Engineering systems are inherently interdisciplinary and that requires a thorough approach to the measurement and evaluation of these systems.

This thesis operates from the assumption that a model-based approach to enable the evaluation of engineering systems is desired. The model-based evaluation approach is particularly valuable when analyzing engineering systems and interventions. The requirement for a deep understanding of the system is satisfied as the properties, dynamics, and externalities of the engineering artifacts in the system are well-known. Furthermore, decision-making around engineering systems often results in invasive, structural interventions, in which case a purely data-driven approach is not sufficient. Section 2.2 investigates the different types of models and identifies the need for hetero-functional graph theory.

2.2 Literature Gap

The second section to this chapter provides a closer look at the gap in the literature of model-based approaches for the analysis of engineering systems. First, the section provides an overview and comparison of the different types of modeling methods for engineering systems (Section 2.2.1). Then, the section focuses on quantitative structural modeling

methods (Section 2.2.2). It highlights the need for hetero-functional graph theory as a result of the modeling restrictions imposed by other quantitative structural modeling methods.

2.2.1 Model-Based Engineering System and Intervention Evaluation

The previous section provided a comparison of the different methods for the evaluation of engineering system interventions. This section takes a closer look at the model-based intervention evaluation methods. Some literature has posited the “end-of-theory" given the explosion in the availability of data [119]. This section, however, demonstrates that theory plays an essential role in the future of engineering systems [120–122]. The discussion is structured in congruence with the classification of modeling methods as displayed in Figure 2.3. Section 2.2.1.1 discusses graphical models, Section 2.2.1.2 discusses quantitative structural models, and Section 2.2.1.3 discusses quantitative behavioral models.

The development of theory is critical to the future of engineering system design and intervention evaluation because ...

- ... it defines meta-data features in data collection.
- ... it ensures a deep understanding of the modeled system so that both structural and behavioral interventions are understood.
- ... it ensures a deep understanding of the modeled system such that the knowledge gaps are explicit. It requires assumptions and has the ability to inform future research (to test those assumptions).

Model-based evaluation of interventions does not forego the use of data and experiments; it can leverage those in testing assumptions and creating a deeper understanding through extensive simulation and testing.

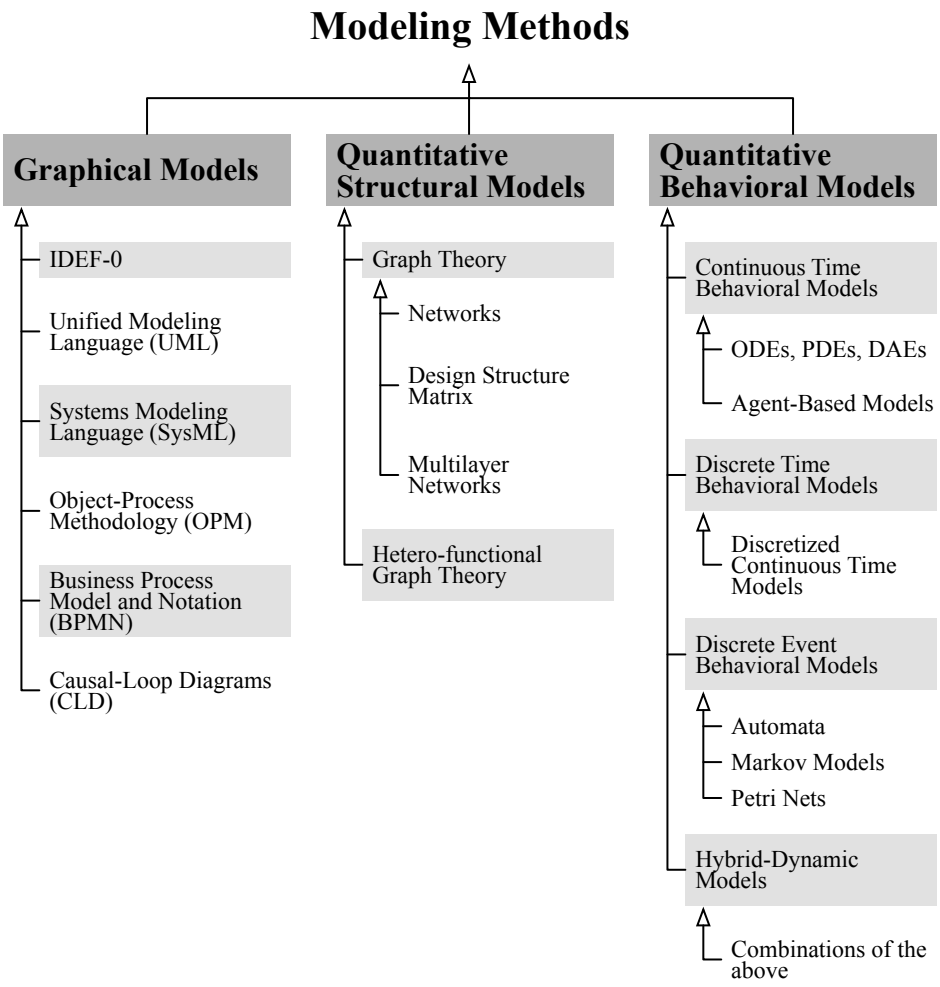


Fig. 2.3: Classification of modeling methods for the evaluation of engineering system interventions.

2.2.1.1 Graphical Models

Graphical models have been used to describe a wide range of systems, from technical to socio-economic[1]. Graphical models are qualitative in nature and often used to communicate the structure of a system, qualitative information, and the ontology of a system or a class of systems. Notably, graphical models are not limited in the heterogeneity of the modeled system.

The general downside of graphical models is the lack of support for quantitative analyses

[1]Note that this definition is distinct from the graphical models in the field of machine learning in which graphical models refer to “graph-based" models, as described in Section 2.2.1.2.

of the models. However, some methods have been developed to gain quantitative insights based on graphical modeling methods. These are often developed as part of a specific software package for the modeling method.

Below, a number of graphical modeling methods are introduced as a rough overview of the landscape. This list is not exhaustive but it provides the reader with a starting point.

IDEF0 diagrams enable the decomposition and architecture of system function [123]. For each function, IDEF0 lays out the inputs, controls, and mechanisms required to create the output. For clarity, the method relies on aggregation and decomposition of processes to limit the number of processes to six per layer of modeling abstraction. IDEF0 is one of the IDEF family of modeling languages. These languages originated in the 1970's with funding from the U.S. Air Force.

Unified Modeling Language (UML) was developed to provide a consolidated approach to object-oriented modeling methods [124]. UML was originally intended for software and firmware, but its strengths have since been recognized and the methods were applied to other fields.

Systems Modeling Language (SysML) borrows many features of UML and customizes them for cyber-physical systems. These include block definition diagrams and activity diagrams. SysML also includes a new set of diagrams to address the physical nature of these systems (e.g. the internal block definition diagram) and direct support for requirements engineering [104]. SysML is the most commonly used modeling language among systems engineers.

Model-based Systems Engineering created the Systems Modeling Language (SysML) as an abstracted graphical model with sufficient ontological breadth to integrate and synchronize more detailed domain-specific engineering models. SysML is qualitative and graphical in nature and is not meant to develop complex mathematical models that provide engineering insight. Rather, SysML provides systems engineers and project managers with a tool by which to quickly understand the overall structure and behavior of a system and its component

modules so as to coordinate its engineering development in large organizations and across multiple engineering teams.

SysML leverages multiple modeling frameworks to represent the full breadth and complexity of an engineering system. This multitude of diagrams allows the modeler to separate, for example, form from function to study processes in a solution neutral environment. The downside of using SysML is that the modeler needs to leverage the right diagrams to model the system. This thesis leverages the various SysML concepts extensively. SysML and Model-Based Systems Engineering are more extensively discussed in Section 2.3.2.

Object-Process Methodology (OPM) has been developed explicitly for the modeling of general purpose systems with both system form and behavior in mind [125]. OPM has the benefit of having a single hierarchical model, and using a single type of diagram to represent the full system. However, OPM is missing the breadth to capture all aspects of a system.

Business Process Model and Notation (BPMN) is developed to support decision making around business processes [126]. The goal is to provide a language that can be intuitively understood by all stakeholders of the process. BPMN has overlap in functionality with activity diagrams in SysML, but BPMN is specifically and more narrowly designed for business processes.

Causal-Loop Diagrams have been used to describe socio-technical systems. These use a directed graph approach to connect (hard and soft) variables as feedback loops. Causal-loop diagrams are easily understood by stakeholders and can enable conversations about the dynamics of a system. Two downsides are that causal-loop diagrams quickly become complex and the method doesn't lend itself for a hierarchical decomposition of the system. *"System Dynamics"* is a quantification of causal-loop diagrams. It was first developed in the '50s at MIT to model nonlinear behavior of stocks, flows, and feedback loops [127]. Over time, it has evolved to address a variety of dynamically complex systems. System Dynamics can be used both qualitatively to describe and model systems, or quantitatively to simulate dynamic behavior with the VenSim or Stella software packages.

2.2.1.2 Quantitative Structural Models

Quantitative structural models mathematically describe a system's structure.

Definition 2.9: System Structure [2, 5] is defined by the parts of a system and the relationships amongst them. It is described in terms of 1.) the system boundary, 2.) the formal elements of the system 3.) the connections between the formal elements 4.) the functional elements of the system and 5.) the allocation of the functional elements to the formal elements. ■

Quantitative structural models have been used extensively to describe both social and technical systems. In all cases, they rely heavily on concepts from graph theory.

2.2.1.2.1 Graph Theory

A **Network** (or graph $\mathcal{G}$) is a general means of representing patterns of connections or interactions between parts of a system [93]. The parts of the system are represented as nodes (or vertices $\mathcal{V}$). The connections or interactions are represented as lines (or edges $\mathcal{E}$). In addition to this set-theoretic definition, graph theory provides incidence and adjacency matrices as a means of algebraic analysis. Networks are used to study systems across a wide variety of disciplines. Examples include the Internet, power grids, transportation networks, social networks, citation networks, biochemical networks, and neural networks. Objectively speaking, the definition of a graph $G = \{V, E\}$ captures only the first three (of five) parts of system structure. Consequently, one of the major shortcomings of Graph Theory is the failure to represent heterogeneity in networks as a result of the simplicity of its mathematical structure. Instead, many works attribute additional data *features* to graphs to expand their utility.

The **Design Structure Matrix**, for example, is a type of network modeling tool [128] that seeks to distinguish the different types of interconnections within a system. The four types of Design Structure Matrix models are: 1. product architecture, 2. organization architecture, 3. process architecture, and 4. multidomain architecture.

Multilayer Networks expand on existing network theory to accommodate the study of networks with heterogeneity and multiple types of connections [95]. Over the past decade, numerous methods have tried to provide a consistent approach to model these networks-of-networks. However, as discussed by Kivelä et al, all these multilayer network methods have their respective modeling limitations. Section 2.2.2 discusses Multilayer Networks and their modeling limitations in more detail.

2.2.1.2.2 Hetero-functional Graph Theory

Hetero-functional graph theory has emerged over the past decade to be the first quantitative structural model that captures all five parts of system structure [5]. It enables the structural modeling of a heterogeneous large flexible engineering system and explicitly accommodates all five types of system processes (i.e. Transform, Transport, Store, Exchange, and Control) and all five types of operands (i.e. Living Organisms, Matter, Energy, Information, and Money) that regularly appear in engineering systems [1]. Furthermore, Hetero-functional Graph Theory has been used as the underlying structure for dynamic system models across many different application domains including power, water, transportation, production, and healthcare systems. It has also been used to study the interdependencies of these systems within the context of interdependent smart city infrastructures[2].

2.2.1.3 Quantitative Behavioral Models

Quantitative behavior models can be broadly classified as 1. Continuous Time Behavioral Models, 2. Discrete Time Behavioral Models, 3. Discrete Event Behavioral Models, and 4. Hybrid Dynamic Behavioral Models. The discussion below provides more detail and a sample of tools for each class of quantitative behavioral models.

[2]The text in this section is currently in press as Chapter 23 in the *Handbook of Engineering System Design* [48]. Here, the text refers to work in the remainder of this thesis. More explicitly: Chapter 3 presents the compilation of HFGT [5], Chapter 4 discusses HFGT for interdependent smart city infrastructures [5], and Chapter 5 presents a dynamic model based on HFGT for industrial energy systems [35]. Additionally, Chapter 6 also develops an optimization program for a HFGT-based dynamic model, which has yet to be published.

2.2.1.3.1 Continuous Time Behavioral Models

Continuous-Time and Discrete-Time Behavioral Models are closely related and can both be further classified into time-varying vs. time-invariant and linear vs. non-linear models. For more detail about that decomposition the authors refer the reader to the first chapter in the book *Introduction to Discrete Event Systems* [129].

Systems of Ordinary and Partial Differential Algebraic Equations (ODEs, PDEs & DAEs) are used to describe continuous time behavioral models. ODEs are often used to describe “lumped" systems while PDEs are used to describe distributed behavior (e.g. the traffic density along a stretch of road). Because it is often analytically or computationally intractable to use a truly distributed PDE, systems of ODEs arranged in a graph structure are often used instead. Bond graphs and linear graphs, for example, are well-known techniques that superimpose the constitutive laws of engineering physics onto the structure of a physical engineered system. Furthermore, pseudo-steady-state assumptions are often made so that a subset of the differential equations are effectively replaced by algebraic equations to form differential algebraic equations. Several software packages have been developed to simulate the systems of DAEs. These include Simscape by Matlab, OpenModelica and Dymola based on the Modelica language.

Agent-Based Modeling (ABM) goes beyond the dynamic laws of engineering physics to study socio-technical and socio-economic systems. ABM leverages dynamic interactions between autonomous entities called agents [130]. As the agents interact with each other, their individual processes and functions result in an emergent system behavior. This “bottoms-up" approach to modeling results in a number of benefits. ABM has the ability to predict emergent phenomena that often defy normal intuition. Furthermore, ABM provides a natural description of a system, especially for socio-technical systems in which individuals make decisions about their use of technical systems. Finally, ABM is flexible in that it can be expanded for the number of agents and their interactions. It also allows for changing levels of aggregation of agents in agent-groups.

2.2.1.3.2 Discrete Time Behavioral Models In contrast to the continuous time models, discrete time models are based on sampled data points or signals in digital form [131]. Generally, digital systems are more accurately represented with discrete time models whereas engineering physics are more accurately described with continuous-time models. Engineering systems can often be modeled using theory from either continuous or discrete mathematics. The decision to use either continuous or discrete mathematics to model an engineering system depends primarily on the role of data and its discretization. In many cases the data is intrinsically discretized, or the data-collector has made pseudo-steady-state assumptions that force discrete-time step-wise evolution of algebraic equations. In other cases, data is not available and so idealized differential equations can be used. In either case, continuous-time and discrete-time models can be readily transformed from one to the other. In the case of linear systems, discrete-time systems of equations can be solved algebraically with the use of the Z-transform in much the same way that continuous-time systems can be solved algebraically with the Laplace transform.

2.2.1.3.3 Discrete-Event Behavioral Models Discrete-event behavioral models move from a time-driven view of the world to one that is *event-triggered*. In such a case, the system remains in a discrete state until an event causes the system to flip into another state. Many discrete-event engineering systems exist, particularly as a result of automation where the underlying code is itself event-driven. Discrete-event models always have discrete-state that is usually denoted by integers (rather than real or complex numbers).

Automata are one type of discrete-event model that are defined by a finite and countable set of discrete-states that each represent some phenomenon (that is often qualitative in nature). This includes on/off states as well as hot/cold or red/yellow/green. These states are described by nodes. Meanwhile, arcs are used to describe the event triggers that allow a switching behavior from one state to another. These triggers can be either endogenous or exogeneous rules and are often described by Boolean expressions (i.e. if $x \geq 0$ then

switch from State 1 to State 2). While Automata have deep roots in theoretical computer science, they have since found broad application in describing the operational behavior of many engineering systems that have an underlying discrete decision-space. Automata are also often useful to describe operational modes of systems (e.g. normal, emergency, and restore) [129]. Despite these many strengths, the primary weakness of automata is that they have a centralized notion of state; and consequently all the states must first be enumerated in order for the complete automata to be well-defined.

Markov Models are a type of stochastic automata. They have been used to describe decision-making processes in a dynamic and stochastic environment [132]. Markov models have one of a finite number of states and stochastic events causes transitions between states. The evolution of state is tracked with each passing event or decision. Markov Chains are a type of Markov model in which the probabilities of transitions are fixed over time. These Markov models can be used to support decision-making in that they can help to estimate the effects of a certain decision, including subsequent decisions of others actors in the system.

Petri nets are another type of (deterministic) discrete-event model[3]. Unlike automata, they have a decentralized description of state. In their simplest form, Petri nets consist of a set of places that define a state space, transitions that define events between a given pair of places, and a set of directed arcs that connect places and transitions [129]. In effect, these arcs create a *bipartite graph* between the sets of arcs and events. Furthermore, tokens are stored in places and are moved as each transition is "fired". The state of the system as a whole is described by a vector showing the number of tokens in each place. While Petri nets and automata have equal modeling power in that one can be mathematically transformed from the other (without loss), Petri nets can describe a relatively large number of automaton states with a relatively small number of places. Furthermore, because Petri nets are often represented graphically, they often lend themselves to modeling distributed engineering systems such as warehouses, manufacturing systems, or supply chains more

[3]The remainder of this thesis leverages Petri nets in their various forms to represent operand behavior (Section 3.5) and to enable the development of a dynamic system model (Chapters 5 and 6).

generally. Finally, in recent decades, the Petri net literature has expanded to accommodate time-driven dynamics through Timed and Time Petri Nets. They have also incorporated various types of stochasticity with stochastic and fuzzy Petri nets.

2.2.1.3.4 Hybrid Dynamic Behavioral Models Hybrid dynamic behavior models combine the attributes of continuous/discrete time models with discrete-event models [133]. Generally speaking, they consist of a top “layer" described by either an automata or Petri net whose dynamics are either deterministic or stochastic. The bottom layer has a system of differential algebraic equations for each discrete state defined in the top layer. A classic example is the thermostat in a house. When the temperature is above a specified threshold, the heating system is idle. However, as soon as the temperature drops below the threshold, the heating system is activated and starts to heat the house. The model that is used to describe the “idling" state is distinct from the model that describes the “heating" state of the system.

Although hybrid dynamic systems have tremendous relevance to the understanding of engineering systems and their interventions, they remain at the cutting-edge of systems research. First, hybrid dynamic models often rely on discipline-specific DAE models. Consequently, some researchers resort to strapping together multiple (often off-the-shelf) simulators within co-simulation environments. In other cases, researchers develop custom simulators in order to address the specific needs of the engineering system under study. The literature contains many such simulators [134]. Finally, from an analytical perspective, there is a severe lack of theory that combines both discrete and continuous states. Consequently, many of the typical analytical methods applied to continuous-time systems (e.g. stability theory) or discrete-event systems (e.g. reachability analysis) can not be readily applied to hybrid dynamic systems.

2.2.1.4 Summary

This subsection discussed the pros and cons of three types of models for engineering systems: 1) graphical models, 2) quantitative structural models, and 3) quantitative behavioral models. The graphical models are intuitive, the quantitative structural models are generalizable across many domains, and the behavioral models leverage mathematics to understand structure and behavior of specific systems.

For the analysis of engineering systems, quantitative structural models are especially valuable. In contrast to graphical models, they provide a quantitative analysis framework. In contrast to quantitative behavioral models, they are generalizable across engineering domains. Quantitative structural models enable an integrated, quantitative approach to modeling and analyzing engineering systems.

Section 2.2.2 highlights why the dominant quantitative structural approach (multilayer networks) falls short in modeling capability for engineering systems and provides insight to the need for hetero-functional graph theory as a means to model engineering systems.

2.2.2 The Need for Hetero-functional Graph Theory

Subsection 2.2.1 discussed various modeling approaches for engineering systems and emphasized the value of quantitative structural models. This section establishes the need for the advancement of *Hetero-functional Graph Theory* as a quantitative structural modeling framework for engineering systems. In contrast to the multi-layer network approaches, hetero-functional graph theory does not impose limitations on the physical system it describes. This section provides a small scale example that violates all limitations imposed by multi-layer network approaches. The limitations of multi-layer network approaches prevent them from describing the example system. Consequently, it is demonstrated that there is a need for an approach that does not impose those limitations: Hetero-functional Graph Theory.

Recently, the network science literature has advanced the concept of “multi-layer net-

works" where two or more network “layers" interact with each other to represent a system of interdependent critical infrastructures (ICIs) [95, 135, 136]. Much like a conventional graph, a multi-layer network $G_M = \{V_M, E_M\}$ is formally defined as a tuple of nodes V_M and edges E_M. Such a multi-layer network is organized into an integer n number of layers $L_1 \dots L_n$. Here, a given layer $L_\alpha = \{V_\alpha, E_\alpha\}$ is understood as a graph where the nodes V_α and edges E_α have at least one semantic aspect, feature or operand in common (e.g. electricity, water, people etc). Furthermore, the multi-layer network edges $E_M = E_A \cup E_C$ can be classified into intra-layer edges E_A and extra-layer edges E_C [95]. Despite these definitions, the multi-layer network community *“has produced an equally immense explosion of disparate terminology, and the lack of consensus (or even generally accepted) set of terminology and mathematical framework for studying is extremely problematic"* [95]. In a comprehensive review on the common ontological elements and modeling limitations of “multi-layer networks", Kivelä et al. showed that *all* of the reviewed works have exhibited at least one of the following modeling constraints [95]:

1. Alignment of nodes between layers is required
2. Disjointment between layers is required
3. Equal number of nodes for all layers is required
4. Exclusively vertical coupling between all layers is required
5. Equal couplings between all layers are required
6. Node counterparts are coupled between all layers
7. Limited number of modelled layers
8. Limited number of aspects in a layer

The practical limitations of these modeling constraints are best explained graphically by counter-example on the small hypothetical four-layer network shown in Figure 2.4. This

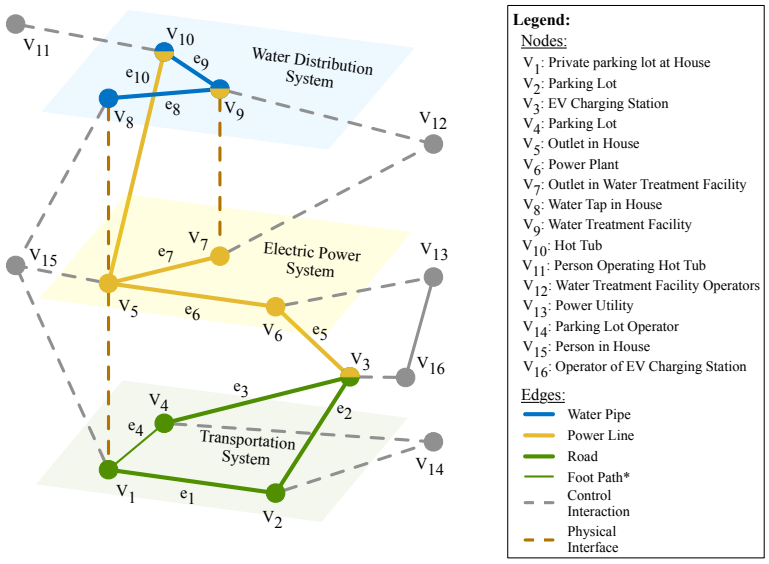


Fig. 2.4: A Hypothetical Four Layer Network: It represents transportation, electric power, and water distribution infrastructure with a super-imposed cyber-control layer. *: The foot path is part of the Transportation System, but differs in modality from the other edges in the system and is represented with a thinner edge.

figure is adopted from Chapter 4.11 in which a more extensive description of the figure is added. The figure consists of three physical layers in green, yellow, and blue representing the transportation, electric power, and water distribution infrastructures respectively. The fourth layer is the cyber layer (in grey) representing cyber-control infrastructure. Lastly, the network contains physical links between layers in dashed red and informatic links in dashed grey. In all, there are 16 nodes (or vertices), 9 intra-layer edges, and 15 cross-layer edges. Each node and edge coloring represents a different infrastructure system. Nodes connected by cross-layer edges sometimes represent a single *resource* (or facility[4]) with presence in several infrastructures. For example, the group of nodes v_1, v_5, and v_8 can represent a house that combines a private parking lot in the transportation system (v_1), an electric power grid connection that supplies power to the house (v_5), and a water tap in the water distribution network (v_8). A resource r_i is, therefore, often defined by a graph $G_{Ri} = \{V_{Ri}, E_{Ri}\}$ composed of "counterpart" nodes V_{Ri} and coupling edges $E_{Ri} \subseteq E_C$. Arbitrary nodes v_i in layer L_α and v_j in layer L_β are said to be "counterparts" of each other if there exists no cross-layer

[4]These resources or facilities are referred to as entities in Kivelä et al. [95]

edges between L_α and L_β involving v_i or v_j other than $e_{xk} = \{v_i, v_j\}$. That said, nodes do not need to be part of resources in order to be connected across layers. For example, v_{10} may represent a hot tub that is connected to the same electric load bus v_5, but draws its water from the water distribution system directly. Furthermore, Node v_3 may represent a charging station as a resource that is part of the transportation and electricity infrastructures in a single node. Node v_{14} acts as a centralized controller agent for v_2, and v_4. It may represent a bus or taxi dispatching authority. Meanwhile, node v_{16} acts as another controller agent for node v_3. It may represent the charging station's cloud-based energy management software. Note that this topology may cause conflict between the transportation objective of node v_{14} and the energy management objective of node v_{16}. The remainder of this chapter uses Figure 2.4 to examine the eight modeling constraints found in multi-layer networks [95]. The example in Figure 2.4 violates all eight constraints.

In *Constraint 1*, some "multi-layer networks" require all layers to have vertically aligned nodes [72, 94, 95, 137–183]. In other words, for any node v_i in layer L_α, there must exist another *connected* node v_j in layer L_β. In Figure 2.4, however, v_4 is not connected to v_7. It, therefore, represents an example infrastructure with modeling requirements greater than those provided by multi-layer networks with vertically aligned nodes.

In *Constraint 2*, some "multi-layer networks" require disjoint layers [74, 95, 173, 177, 184–200]. In other words, any node v_i can only be part of a single layer L_α and no two layers intersect. In Figure 2.4, however, the charging station, v_3, pertains to both the electrical layer as well as the transportation layer. Figure 2.4, therefore, represents a simple example infrastructure with modeling requirements greater than those provided by multi-layer networks with disjoint layers.

In *Constraint 3*, some "multi-layer networks" require the same number of nodes in each

layer [74, 94, 95, 137–173, 180, 182, 187, 191, 192, 201, 202]. In Figure 2.4, however, the water network has three nodes while the transportation network has four. Note that this constraint is similar to the first, except that it does not require for a node in a given layer to be connected to other layers. Figure 2.4, therefore, represents an example infrastructure with modeling requirements greater than those provided by multi-layer networks with an equal number of nodes in each layer.

In *Constraint 4*, some “multi-layer networks" require exclusively “vertical"[5] cross-layer couplings [94, 95, 137–173, 180, 182, 186, 202–206]. In other words, all the interlayer edges $E_C = \cup_i E_{Ri}$ are coupling edges. In Figure 2.4, however, node v_{10} is not a counterpart of node v_5 and therefore there exists a cross-layer edge that is not a coupling edge. Again, Figure 2.4 represents an example infrastructure that violates a constraint imposed by multi-layer network theory; the requirement for “vertical" couplings.

In *Constraint 5*, some “multi-layer networks" require that all nodes in a given layer have identical couplings to nodes in another layer [95, 138–163, 167–173, 180, 182, 186, 202–206]. In Figure 2.4, however, node v_5 and v_8 have a cross-layer edge but v_4 and v_7 do not. Consequently, Figure 2.4 presents an example in which the requirement for identical cross-layer couplings is not met.

In *Constraint 6*, some “multi-layer networks" require that each node is connected to all of its counterparts in other layers [138–142, 146–163, 167–173, 180, 182, 186, 202–206]. In Figure 2.4, however, many nodes (e.g. v_2, and v_4) are not connected to other layers. Figure 2.4, therefore, represents an example infrastructure with modeling requirements greater than those provided by multi-layer networks with all nodes connected to their counterparts.

[5] Kivelä et al. [95] refer to this constraint as “diagonal" couplings. This work adopts the term vertical to more closely reflect the depiction in Figure 2.4.

In *Constraint 7*, some “multi-layer networks" limit the number of layers to two [74, 169–171, 173, 177, 180, 182, 184–213]. In Figure 2.4, however, there are three physical networks and one cyber-network. Dual-layer analysis is likely insufficient in reflecting the number of layers, as is the case for this simple example. Figure 2.4, therefore, exceeds the maximum of two layers as imposed in Constraint 7 on any multi-layer network.

In *Constraint 8*, some “multi-layer networks" require that each layer have no more than one aspect [74, 94, 137–173, 177, 180, 182, 184–200, 202]. However, many networks have multiple aspects. Transportation systems are often multi-modal including passenger vehicles, buses, trains, and pedestrians [31]. Figure 2.4 includes two modes of transportation, a foot path between nodes v_1 and v_4, and roads for all other connections. Therefore, the system represents an example where the modeling requirements exceed those provided by multi-layer networks with a single aspect.

The violation of the eight constraints demonstrate insufficiencies in the existing theory of multi-layer networks to address a simple example of multiple arbitrarily connected infrastructure systems. In other words, there exist ontological limitations to the multi-layer networks models of the physical system. Consequently, a new theory needs to be introduced that does not have the aforementioned limitations. Chapter 3 presents such a theory, herein called Hetero-functional Graph Theory (HFGT). It specifically addresses many of the limitations identified thus far in this chapter. In order to facilitate its explanations the following section (Section 2.3) is dedicated to highlighting the fundamental concepts upon which HFGT is based.

2.3 Hetero-functional Graph Theory Preliminaries

This section is devoted to providing the reader with the fundamental concepts that are required for a solid understanding of the hetero-functional graph theory presented in Chapter 3. This foundation draws upon the theory of ontologies in Section 2.3.1. Section 2.3.2 then draws from the fields of systems engineering and engineering systems to present several definitions and concepts upon which hetero-functional graph theory is based.

2.3.1 Ontological Foundation for Hetero-Functional Graph Theory

The modeling constraints found in multi-layer networks can be viewed from an ontological perspective. In the ontological sciences, the relationship between reality, the understanding of reality, and the description of reality is described by Ullman's Triangle [214]. Figure 2.5 displays Ullman's Triangle on the left, where reality is the *Real Domain* $\mathcal{D}$, understanding of reality is *Domain Conceptualization* $\mathcal{C}$, and the description of reality is *Language* $\mathcal{L}$. These general concepts are instantiated to describe modeling of real systems, where the reality is the *Physical System*, the modeller's understanding (or mental conceptualization) is the *Abstraction* $\mathcal{A}$, and the description of the abstraction is the *Model* $\mathcal{M}$. Figure 2.5 presents the instantiation of Ullman's general concepts on the right.

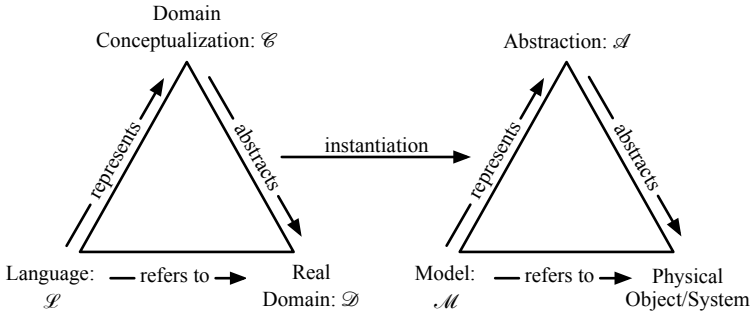


Fig. 2.5: Ullman's Triangle [49]: Its ontological definition. On the left, the relationship between reality, the understanding of reality, and the description of reality. On the right, the instantiated version of the definition.

The modeling process creates an abstraction of the physical system, and represents

the abstraction with a model [49]. The model refers to the physical system, but this reference is always indirect, as an abstraction is always made in the modeling process. The abstraction of reality may be entirely conceptual (residing within the mind) or linguistic (residing within some predefined language). In order for a model $\mathcal{M}$ to truly represent the abstraction $\mathcal{A}$, the modeling primitives of the language $\mathcal{L}$ should "faithfully represent the domain conceptualization $\mathcal{C}$ to articulate the represented abstraction $\mathcal{A}$" [49]. In this definition, modeling primitives directly express relevant domain concepts, creating the domain conceptualization [49]. Visually, the relationship between the Conceptualization, the Modeling Language, the Model, and the Abstraction is presented in Figure 2.6.

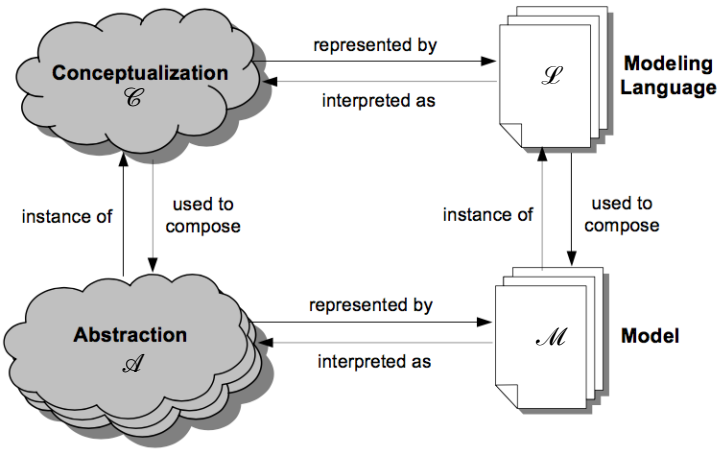


Fig. 2.6: The Relationship Between Four Ontological Science Concepts [49]: Conceptualization, Abstraction, Modeling Language, and Model.

In this work, the abstraction is the class of engineering systems. The model, within the scope of this work, is assumed to be mathematical in nature, and more specifically, graph theoretic. The fidelity of the model with respect to an abstraction is determined by the four complementary linguistic properties shown in Figure 2.7 [49]: soundness, completeness, lucidity, and laconicity [214]. When all four properties are met, the abstraction and the model have an isomorphic (one-to-one) mapping and faithfully represent each other. The four properties, Soundness, Completeness, Lucidity, and Laconicity, are defined as follows:

Definition 2.10 – Soundness [214]: A language $\mathcal{L}$ is sound w.r.t. a domain conceptualiza-

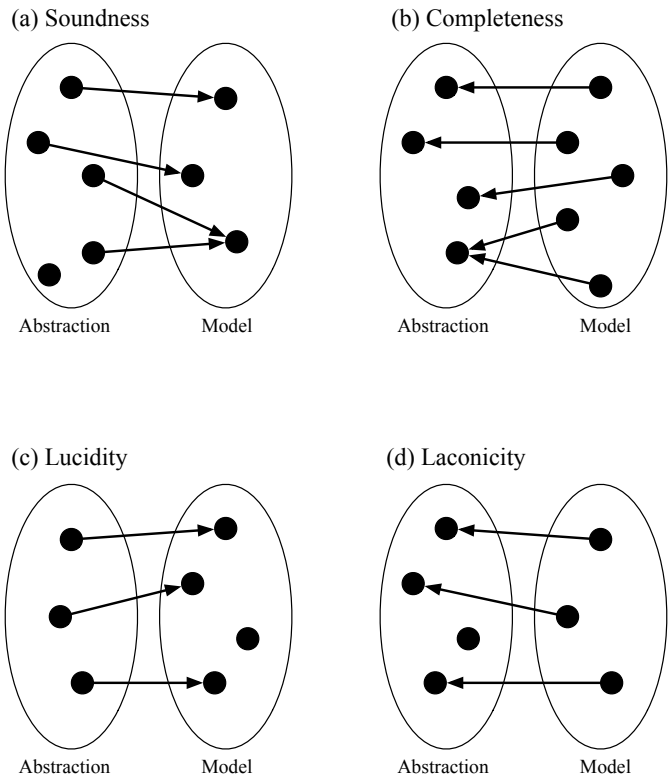


Fig. 2.7: Graphical Representation of Four Ontological Properties As Mapping Between Abstraction and Model: **a** Soundness, **b** Completeness, **c** Lucidity, and **d** Laconicity [49].

tion $\mathcal{C}$ iff every modeling primitive in the language ($\mathcal{M}$) has an interpretation in the domain abstraction $\mathcal{A}$. (The absence of soundness results in the excess of modeling primitives w.r.t. the domain abstractions as shown in Figure 2.7.c on lucidity.) ■

Definition 2.11 – Completeness [214]: A language $\mathcal{L}$ is complete w.r.t. a domain conceptualization $\mathcal{C}$ iff every concept in the domain abstraction $\mathcal{A}$ of that domain is represented in a modeling primitive of that language. (The absence of completeness results in one or more concepts in the domain abstraction not being represented by a modeling primitive, as shown in Figure 2.7.d on laconicity.) ■

Definition 2.12 – Lucidity [214]: A language $\mathcal{L}$ is lucid w.r.t. a domain conceptualization $\mathcal{C}$ iff every modeling primitive in the language represents at most one domain concept in abstraction $\mathcal{A}$. (The absence of lucidity results in the overload of a modeling primitive w.r.t. two or more domain concepts as shown in Figure 2.7.a on soundness.) ■

Definition 2.13 – Laconicity [214]: A language $\mathcal{L}$ is laconic w.r.t. a domain conceptualization $\mathcal{C}$ iff every concept in the abstraction $\mathcal{A}$ of that domain is represented at most once in the model of that language. (The absence of laconicity results in the redundancy of modeling primitives w.r.t the domain abstractions as shown in Figure 2.7.b on completeness.) ■

Given the discussion provided in Section 2.2.2, currently defined multi-layer networks maintain soundness and laconicity but lack completeness and lucidity. Let us reconsider Figure 2.4 on page 39.

- Soundness is maintained. There are no excess modeling primitives than those required by the example.

- Completeness is not maintained. The set of mathematical modeling elements are insufficient to represent all conceptual elements in the abstraction. For example, and as the following sections will discuss at length, multi-layer networks do not introduce the concept of system function. Reconsider nodes v_2 as a parking lot and v_3 as a charging station. Not only are charging stations and parking lots fundamentally different types of facilities, but the former has a superset of the functionality of the latter. While it is possible to reflect the hybrid functionality of the charging station with a mixed yellow and green circle (as in the figure), this graphical depiction is not represented in the mathematics of multi-layer networks.

- Lucidity is not maintained. Not all conceptual elements in the abstraction have unique representations in the mathematical model. The representations are overloaded. For example, consider nodes v_2 as a parking lot and v_3 as a charging station. The same mathematical element of a green node is used to represent two fundamentally different abstractions of reality.

- Laconicity is maintained. There are no redundant or synonymous modeling primitives.

This ontological analysis of multi-layer networks suggests that its underlying modeling has

several fundamental shortcomings. Instead, a different modeling language with a richer ontological foundation is required.

In order to define a Language $\mathcal{L}$, the Domain Conceptualization $\mathcal{C}$ needs to be clearly defined. This chapter continues to define the fundamentals of such a Domain Conceptualization in Section 2.3.2. Afterwards, Chapter 3 defines a language called hetero-functional graph theory.

2.3.2 Systems Engineering Foundations

This thesis looks to the field of systems engineering [104] for its domain conceptualization. Because the modern systems engineering field developed methodologically from industrial origins in the aerospace, communications, and defense sectors, it had to address a tremendous heterogeneity of applications in an immediately practical way.

Definition 2.14 – Systems Engineering (SE [104]): An interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, and then proceeding with design synthesis and system validation while considering the complete problem: operations, cost and schedule, performance, training and support, testing, manufacturing, and disposal. SE considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user's needs. ■

Originally, the practice of systems engineering was document-centric so as to capture customer requirements and trace them through the development and delivery of a system. As the field developed, engineering models became the center of practice.

Definition 2.15 – Model-Based Systems Engineering (MBSE [104]): The formalized application of modeling to support system requirements, design analysis, verification, and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases. ■

The integration of modeling into the practice of MBSE immediately faced the practical challenge that no single model or modeling language was sufficient for the development of systems across the entirety of its scope and throughout all stages of its engineering development [67–69]. Consequently, different models addressing different aspects and scopes of the overall system often became inconsistent [215, 216]. For example, a space shuttle requires a thermal-fluidic propulsion model, a structural model, a control system model, and an electrical model. All of these are coupled and must be synchronized as developments are made to each one.

To resolve this seemingly intractable problem, the field of MBSE developed the Systems Modeling Language (SysML) [68, 69] as an abstracted graphical model with sufficient ontological breadth to integrate and synchronize more detailed domain-specific engineering models. Here, the focus was not to develop complex mathematical models that provide engineering insight but rather to provide systems engineers and project managers with a tool by which to quickly understand the overall structure and behavior of a system and its component modules so as to coordinate its engineering development in large engineering organizations across multiple teams [68, 69].

Recently, many have sought to use MBSE beyond the scope of complex products (in the aerospace and defense sectors) and for large scale engineering systems (that form integral parts of smart cities). Examples of these include the power grid, transportation systems, healthcare delivery systems, and the internet.

Definition 2.16 – Engineering System [1]: 1.) A class of systems characterized by a high degree of technical complexity, social intricacy, and elaborate processes aimed at fulfilling important functions in society. 2.) The term engineering systems is also used to refer to the engineering discipline that designs, analyzes, verifies, and validates engineering systems. ■

From an architectural perspective, these engineering systems are classified as large and flexible.

Definition 2.17 – Large Flexible Engineering System (LFES [3, 14]): An engineering system with a large set of system processes that not only evolve over time, but also can be fulfilled by one or more resources[6]. ■

Hetero-functional graph theory was developed for this broad class of systems. Furthermore, large flexible engineering systems, as a class of systems, is characterized by a *meta-architecture* which generalizes and abstracts features found in reference and instantiated system architectures. Figure 2.8 uses a SysML Block Diagram to show how a meta-architecture generalizes a reference architecture which in turn generalizes an instantiated architecture. Consequently, the insights provided by hetero-functional graph theory at the meta-architecture level have direct pertinence to domain specific reference architectures and their instantiations in real-world applications.

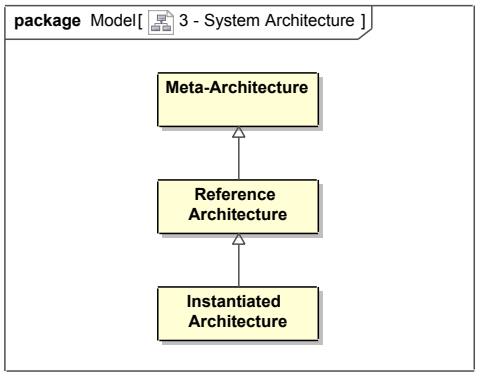


Fig. 2.8: SysML Block Diagram: System architecture can be represented at three levels of abstraction: instantiated, reference, and meta.

For clarity, definitions for these terms are provided. A meta-architecture, a reference architecture, and an instantiated architecture are all types of (system) architectures.

Definition 2.18 – Systems Architecture [98]: A system architecture consists of three parts, the physical architecture, the functional architecture, and their mapping. The *physical architecture* is a description of the partitioned elements of the system without any specification of

[6]The original definition of large flexible engineering systems found in the Axiomatic Design literature [3] used the term functional requirements instead of system processes and the term design parameters instead of system resources. Recent works on hetero-functional graph theory [14, 98] have since adopted the definition revised as per the above for consistency of nomenclature.

the performance characteristics of the physical resources that comprise each element. The *functional architecture* is a description of the system processes in a solution-neutral way, structured in serial, or parallel, and potentially in hierarchical arrangements. The *system concept* as a mapping of the functional architecture onto the physical architecture completes the system architecture. ■

Instantiated architectures are the easiest to understand because they represent the wide variety of instantiated products and systems that everyone interacts with on a day-to-day basis. An instantiated systems architecture is now defined as:

Definition 2.19 – Instantiated Systems Architecture: A case specific architecture, which represents a real-world scenario, or an example test case. At this level, the physical architecture consists of a set of instantiated resources, and the functional architecture consists of a set of instantiated system processes. The mapping defines which resources perform what processes. ■

Interestingly, in the case of large flexible engineering systems, the mapping of the instantiated functional architecture to the instantiated physical architecture is *necessarily* one-to-one [98]. Consequently, it adheres to the Independence Axiom in Axiomatic Design theory [3, 4, 217].

Axiom 2.1 – The Independence Axiom: Maintain the independence of the system processes [3, 4, 217] such that: (1) they are mutually exclusive and collectively exhaustive with respect to each other, and (2) they maintain a one-to-one mapping with the system resources in the physical architecture[7]. ■

The adherence to the Independence Axiom is what allows a large engineering system to gain its flexible nature where paired elements of form and function can be routinely added or removed. Furthermore, the requirement for mutually exclusive and collectively exhaustive system processes effectively ensures the properties of completeness and laconicity.

Example 2.1: Consider the IEEE 201-bus power system test case in Figure 2.9 as an example

[7] In certain cases the second condition can be relaxed so as to form a one-to-many mapping. However, the independence axiom explicitly prohibits many-to-many mappings of function to form.

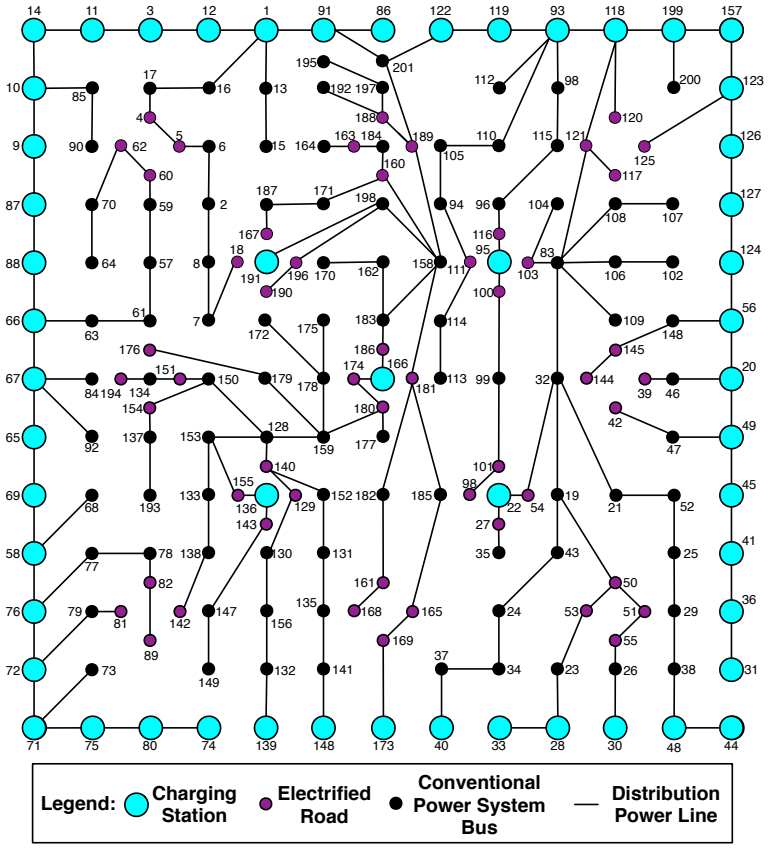


Fig. 2.9: 201-Bus IEEE Test Case One Line Diagram [31, 50].

of an instantiated system architecture. This test case has been used widely to perform power system performance and behavior studies [31, 50, 218, 219]. Based on Definitions 2.18 and 2.19, the instantiated system architecture of the test case has three parts:

- The *Physical Architecture* consists of 201 substation buses to which 1 generator and 95 loads are connected. There are also 200 branches that connect the buses. The physical layout of the test case is presented in Figure 2.9.

- The *Functional Architecture* is comprised of function *instances*. There is one function instance called “Generate Power". There are 95 function instances of “consume power," each is associated with the need to consume power at a given location. There are 201 function instances of “store power" as each substation must inevitably store some – albeit small – amounts of power in order to route the electric power

to other locations. Finally, there are 200 functions of "transport power" between two predetermined locations in space. Note that this description does not specify how these functions are fulfilled by engineering artifacts. The description is thus solution-neutral.

- The *System Concept* is the mapping of of the functional architecture onto the physical architecture and can be derived from the system specification. The generator generates power, and the 95 loads consume power. The 201 substation buses store power, and the 200 branches transport power between their predefined locations in space.

In conclusion, the IEEE 201-bus power system test case represents an instantiated system architecture in that it provides a specific instance as a case study. It includes a unique layout and each element of the system architecture is named and countable. ■

Reference architectures generalize instantiated system architectures. Instead of using individual instances as elements of its physical and functional architecture, it is expressed in terms of *domain-specific* classes of these instances. It is now defined as:

Definition 2.20 – Reference Architecture [220]: *"The reference architecture captures the essence of existing architectures, and the vision of future needs and evolution to provide guidance to assist in developing new instantiated system architectures. . . . Such reference architecture facilitates a shared understanding across multiple products, organizations, or disciplines about the current architecture and the vision on the future direction. A reference architecture is based on concepts proven in practice. Most often preceding architectures are mined for these proven concepts. For architecture renovation and innovation validation and proof can be based on reference implementations and prototyping. In conclusion, the reference architecture generalizes instantiated system architectures to define an architecture that is generally applicable in a discipline. The reference architecture however does not generalize beyond its discipline."* ■

Example 2.2: Reconsider the IEEE 201-bus power system test case described in Example

2.1. The associated reference architecture describes power systems in general. Based on Definitions 2.18 and 2.20, the reference architecture of a power system has three parts:

- The *Physical Architecture* consists of buses, generators, loads, and branches without specifying either the number or names of their instances. The interfaces between these classes are also described in general. Generators and loads connect to individual buses[8] while branches connect to bus pairs. This reference physical architecture is graphically depicted with SysML in Figure 2.10 . A common open-source power system solver uses this reference architecture to for its object-oriented software implementation [221, 222].

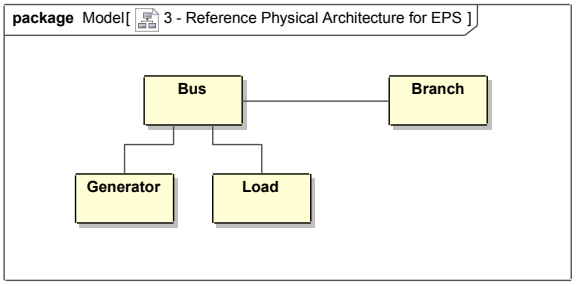


Fig. 2.10: Reference Physical Architecture for Electric Power Systems.

- The *Functional Architecture* comprises of three dominant functions (as types): Generate Power, Transport Power, and Consume Power. For mathematical simplicity and consistency with later sections, the storage of power is considered as transporting power between one place and itself. Again, this description of function does not describe how it is fulfilled and is therefore solution-neutral. This functional architecture is organized into a design pattern to display the system's behavior. Figure 2.11 shows the functional architecture of a power system as a design pattern. Note that the design pattern neither specifies the lay-out nor the number of elements in the power system,

[8] Power system test cases are often used to conduct power flow analysis studies. The underlying model neglects the lead lines from generators and loads to their associated buses. Other models of power systems such as transient stability models do include such lead lines. In such a case, the reference architecture would have no direct connection from the generators and loads to the buses. Furthermore, power flow analysis models usually do not differentiate between energy storage facilities and power generation facilities despite the former's ability to both generate and consume power.

but only incorporates the functional arrangement of the system. The function of every power system, regardless of its physical architecture, is to generate power, transport it to the end users, and consume the power (by the end user). The transportation of power may occur once, but usually occurs multiple times as there very rarely exists direct lines between power plants and end consumers.

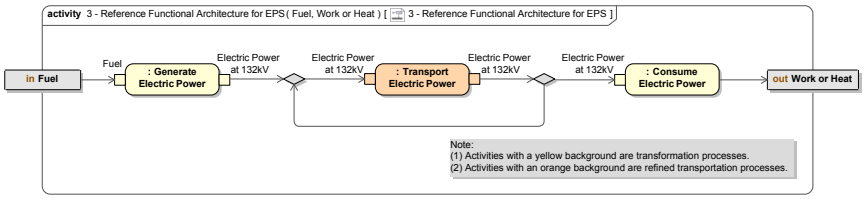


Fig. 2.11: Functional Design Pattern for an Electric Power System Reference Architecture.

- The *System Concept* maps the functional architecture onto the physical architecture. Given the functional and physical architectures at the instantiated and reference levels, and the system concept at the instantiated level, the system concept at the reference level can be straightforwardly deduced. The presence of a mapping between a function instance to a physical elements causes a mapping between their associated function types and physical classes.

In all, the power system reference architectures presents a *domain-specific* representation of a power system's functional architecture, physical architecture, and system concept. ■

Meta-architectures further generalize reference architectures. Instead of domain-specific elements, it is expressed in terms of *domain-neutral* classes.

Definition 2.21 – Meta-Architecture: A reference architecture composed of "primitive elements" that generalize the domain-specific functional and physical elements into their domain-neutral equivalents. ■

While no single engineering system architecture has been developed for all purposes, several modeling methodologies have been developed that span several discipline-specific domains. In the design of dynamic systems, bond graphs [223–225] and linear graphs

[226–230] use generalized capacitors, resistors, inductors, gyrators, and transformers as primitive elements. In business dynamics, stocks and flows are often used as primitives [231, 232]. Finally, graph theory [93, 233] introduces nodes and edges as primitive elements. Each of these has their respective sets of applications. However, their sufficiency must ultimately be tested by an ontological analysis of soundness, completeness, lucidity, and laconicity.

Example 2.3: Reconsider the IEEE 201-bus power system test case described in Examples 2.1 and 2.2. The power system reference architecture can be described in a domain-neutral way. At the meta-level, the physical architecture becomes a graph with nodes that correspond to generators, loads, and buses and edges that correspond to power lines. The domain neutrality of a graph is evidenced by its application across multiple domains. That said, this graph is only a representation of the power system's physical architecture at the meta level. It neither represents the functional architecture nor the system concept. In that regard, it is an ontologically inadequate representation of the system's meta-architecture as a whole. The exposition of hetero-functional graph theory in the following chapter serves to address this issue of ontological sufficiency. ■

In conclusion, the field of systems engineering has evolved tremendously in recent decades; first with the incorporation of modeling tools like SysML to become MBSE, and then second with an expansion of scope to engineering systems to become the engineering systems discipline. In that regard, the application of MBSE tools such as SysML is indispensable. The systems can furthermore be described at several levels of abstraction: the meta-architecture, the reference architecture, and the instantiated system architecture levels respectively. Graph Theory is an example of a meta-architecture, capable of describing domain-specific systems in a general manner. The theory presented in this thesis, called hetero-functional graph theory, presents itself to address the ontological insufficiencies of graphs defined purely in terms of nodes and edges. The thesis now continues with an exposition of hetero-functional graph theory in Chapter 3.

Chapter Summary:

This chapter has developed the context and preliminary information for the thesis. First, it introduced a common framework to describe systems and used that framework to establish the value of the model-based approach to engineering systems analysis. Then, it provided a discussion of the different types of model-based evaluation approaches and established that *Hetero-functional Graph Theory* is especially well-suited to describe engineering system structure so as to enable the analysis of engineering systems. Finally, the chapter provided preliminary information as a foundation for the advancement of Hetero-functional Graph Theory in the remainder of this dissertation. ■

Chapter 3

Hetero-functional Graph Theory

Chapter Abstract:

The previous chapter has established the need for a novel modeling framework to describe engineering system structure. This chapter presents a hetero-functional graph theory as the first internally consistent theory for the description of engineering systems structure and is adopted from Chapter 4, called *"Hetero-functional Graph Theory"*, in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5]. Hetero-functional graph theory is a quantitative structural modeling framework and can be understood as an intellectual fusion of model-based systems engineering and network science. This chapter provides an exposition of hetero-functional graph theory in terms of its seven constituent mathematical models and how they relate to their counterparts in SysML.

The seven mathematical models in hetero-functional graph theory are: (1) the System Concept (Section 3.1 on Page 61), (2) the Hetero-functional Adjacency Matrix (Section 3.2 on Page 79), (3) the Controller Agency Matrix (Section 3.3 on Page 86), (4) the Controller Adjacency Matrix (Section 3.4 on Page 92), (5) the Service as Operand Behavior (Section 3.5 on Page 95), (6) the Service Feasibility Matrix (Section 3.6 on Page 102), and (7) the System Adjacency matrix (Section 3.7 on Page 112. The first two models are assumed to be universal and apply to all types of engineering systems. They form the structural model.

The last four apply when it is necessary to differentiate systems. Models 3 and 4 constitute the system control model. Together, they differentiate systems based upon the structure of their control and decision-making. Models 5 and 6 constitute the service model. They differentiate systems based upon the behavior of their operands. These six models are then ultimately combined together for holistic analysis of cyber-physical engineering systems. When integrated together, these models constitute the final product of hetero-functional graph theory: the System Adjacency Matrix.

Throughout this chapter, a small scale example is used to demonstrate the theory. More expansive examples in Chapter 4 demonstrate the application of hetero-functional graph theory at scale: for an interdependent smart city infrastructure system (Section 4.2 on Page 126) and for the four-layer system in Section 4.11 on Page 203 (originally presented in Section 2.2.2 on Page 37).

Hetero-functional graph theory can be viewed as an intellectual fusion of model-based systems engineering and network science. The theory has origins elsewhere: the original target application of hetero-functional graph theory was the automated mass-customized production system literature [234–238]. There, the need to compete in dynamic marketplaces with products of increasingly short product life-cycle drove mass-customized production systems to explicitly foster reconfigurability as a life-cycle property of their integrated automation solutions. The first publication on hetero-functional graph theory emerged in 2006 to address system degrees of freedom (as defined in Definition 3.13) [17]. One year later, *reconfigurability measurement in automated manufacturing systems* was published in 2007 as the first comprehensive treatment of hetero-functional graph theory [2]. It utilized a design structure matrix as a type of graph to address the ease of reconfiguration [7, 19]. Furthermore, it drew the concept of a knowledge base from the Axiomatic Design literature [3] and quantified it as a bipartite graph that allocates function to form [6, 20]. Beyond these

characteristics, the reconfigurability measurement of mass-customized production systems needed to specifically address heterogeneity. Finally, because automation was an essential aspect of mass-customized production systems, the theory was explicitly cyber-physical. Since that time, as discussed in this thesis' Preface, hetero-functional graph theory has been applied in numerous domains with several enhancements to address the peculiarities of each application[1]. As this chapter lays out hetero-functional graph theory, historical footnotes are provided to describe how the theory has developed over time.

This chapter provides an exposition of hetero-functional graph theory in terms of its constituent mathematical models and how they relate to their counterparts in SysML. These models are: (1) the System Concept, (2) the Hetero-functional Adjacency Matrix, (3) the Controller Agency Matrix, (4) the Controller Adjacency Matrix (5) the Service as Operand Behavior, (6) the Service Feasibility Matrix, and (7) the System Adjacency matrix. The first two models are assumed to be universal and apply to all types of engineering systems. They form the structural model. The last four apply when it is necessary to differentiate systems. Models 3 and 4 constitute the system control model. Together, they differentiate systems based upon the structure of their control and decision-making. Models 5 and 6 constitute the service model. They differentiate systems based upon the behavior of their operands. These six models are then ultimately coupled together for holistic analysis of cyber-physical engineering systems. The structural model and the control model have cyber-physical resource interfaces. Similarly, the structural model and the service model have a structure-service coupling. When integrated together, these models constitute the final product of hetero-functional graph theory: the System Adjacency Matrix. Table 3.1 presents a visual overview of the models.

Example 3.1: In order to facilitate the theoretical discussion of hetero-functional graph theory, Figure 3.1 introduces a simplistic example network of a smart city. The network

[1]For a more detailed discussion of the different applications of hetero-functional graph theory, the reader is referred to Chapter 6 in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5]

Table. 3.1: An Overview of the Mathematical Models in Hetero-functional Graph Theory: The shaded area maps mathematical elements to their associated models.

Elements	Mathematical Elements	Structural Model	Control Model	Cyber-Physical Resource Interfaces	Service Model	Structure-Service Coupling	System Adjacency Matrix
I: System Concept ---- Chapter 4.1	System Function (Processes)	■				■	■
	System Form (Physical Resources)	■					■
	System Knowledge Base	■					■
	System Constraints Matrix	■					■
	Structural Degrees of Freedom	■				■	■
II: Hetero-functional Adjacency Matrix ---- Chapter 4.2	System-Sequence Knowledge Base	■		■			■
	System-Sequence Constraints Matrix	■		■			■
	Hetero-functional Adjacency Matrix	■		■			■
	System-Sequence Degrees of Freedom	■		■			■
III: Controller Agency Matrix ---- Chapter 4.3	Physical Resources		■	■			■
	(Cyber-)Resources		■	■			■
	Controller Agency Matrix		■	■			■
IV: Controller Adjacency Matrix ---- Chapter 4.4	Controller Adjacency Matrix		■	■			■
V: Service as Operand Behavior ---- Chapter 4.5	Services				■	■	■
	Service Activities				■	■	■
	Service String and Service Petri net				■	■	■
VI: Service Feasibility Matrix ---- Chapter 4.6	Service Transformation Feasibility Matrix				■	■	■
	Service Transportation Feasibility Matrix				■	■	■
	Service Line Feasibility Matrix				■	■	■
	Service Degrees of Freedom				■	■	■
VII: System Adjacency Matrix ---- Chapter 4.7	System Adjacency Matrix	■	■	■	■	■	■

consists of four nodes and four edges. The nodes are: (1) water treatment facility, (2) solar PV system, (3) house, and (4) work location. The edges are: (1) water pipeline from the water treatment facility to the house, (2) electric power line from the solar PV system to the water treatment facility, (3) electric power line from the solar PV system to the house, and (4) road from the work location to the house, and back. ■

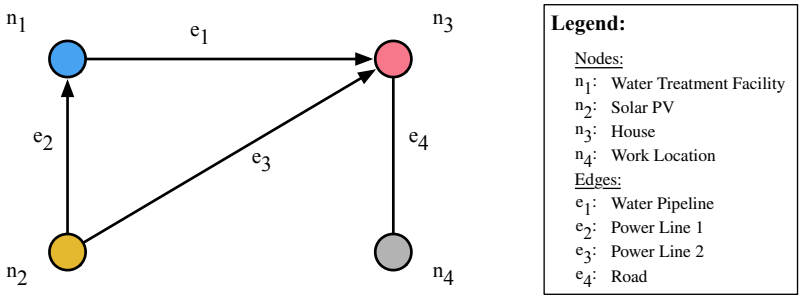


Fig. 3.1: An Example 4-Node Smart City Network: A simplistic smart city network that is used as an example throughout Chapter 3.

3.1 System Concept

Building upon the Definition 2.18 of System Architecture in Section 2.3.2 on page 49, a system's concept is the allocation of a system's *function* to its *form* [67]. The system function is described by elements that include verbs acting upon their operands. The system form is described with elements that include nouns. When an element of function is allocated to an element of form, a complete sentence is formed consisting of a noun-subject, its active transitive verb, and its operand. The sentence as a whole represents a physical *capability* in the system.

SysML consists of several graphical diagrams that represent an engineering system [68, 69]. Of these, the block diagram primarily represents system form while the activity diagram represents system function. Because there is no dedicated diagram for system concept, it appears in both diagrams. Consider the SysML block diagram in Figure 3.2. It shows a model of a smart city's form consisting of several types of resources (as form elements) which are capable of completing several processes (as function elements). This leads to sentence constructions like "Water pipeline transports water." Now consider the activity diagram with swim lanes in Figure 3.3. It shows a model of a smart city's function consisting of a logical flow of processes which have been allocated to several resources in vertical columns called "swim lanes." In each case, the system concept is represented equivalently. Hetero-functional graph theory now represents this SysML description of function allocated to form mathematically [2, 6, 14, 17, 20, 98].

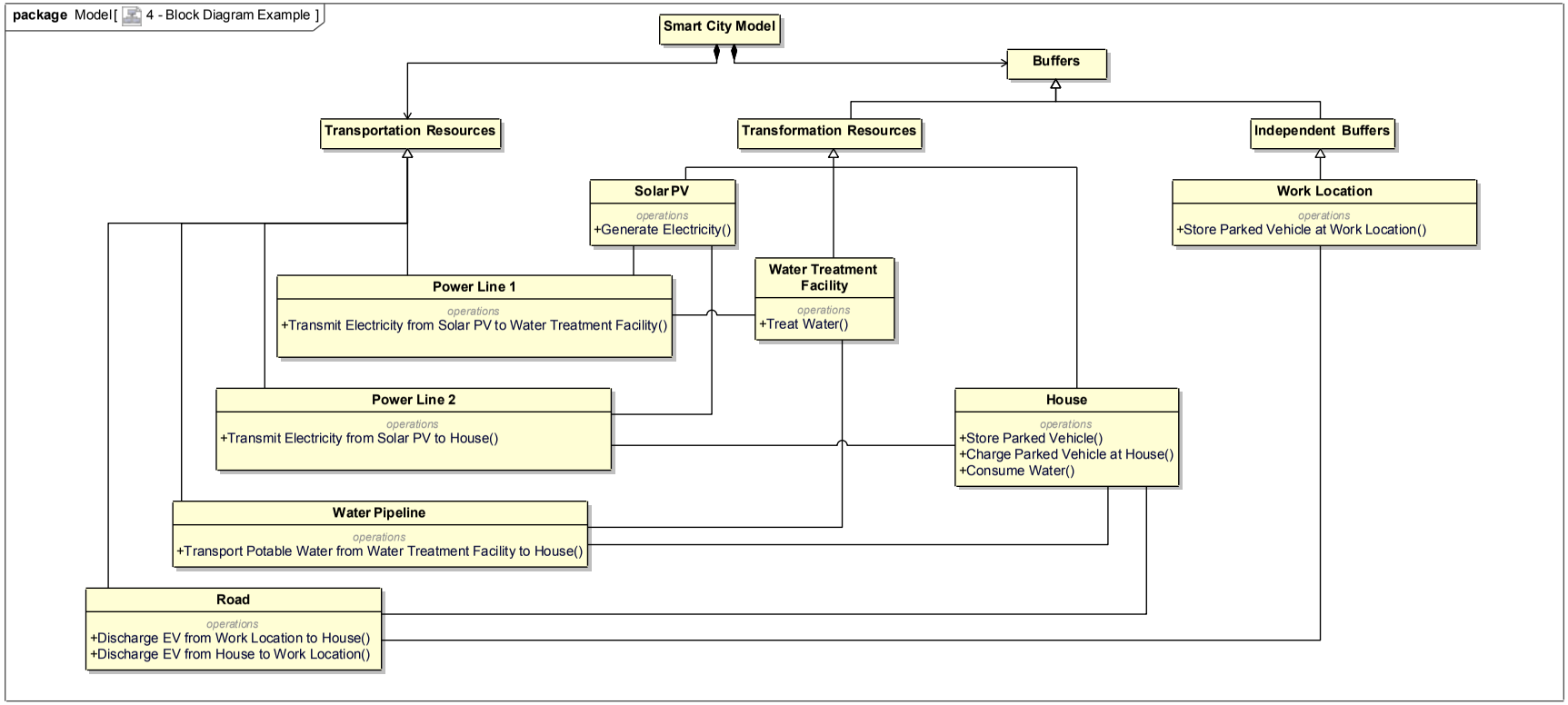


Fig. 3.2: A SysML Block Diagram: A representation of Figure 3.1 using the SysML. The 4-node smart city network consists of transportation, electricity, and water infrastructure.

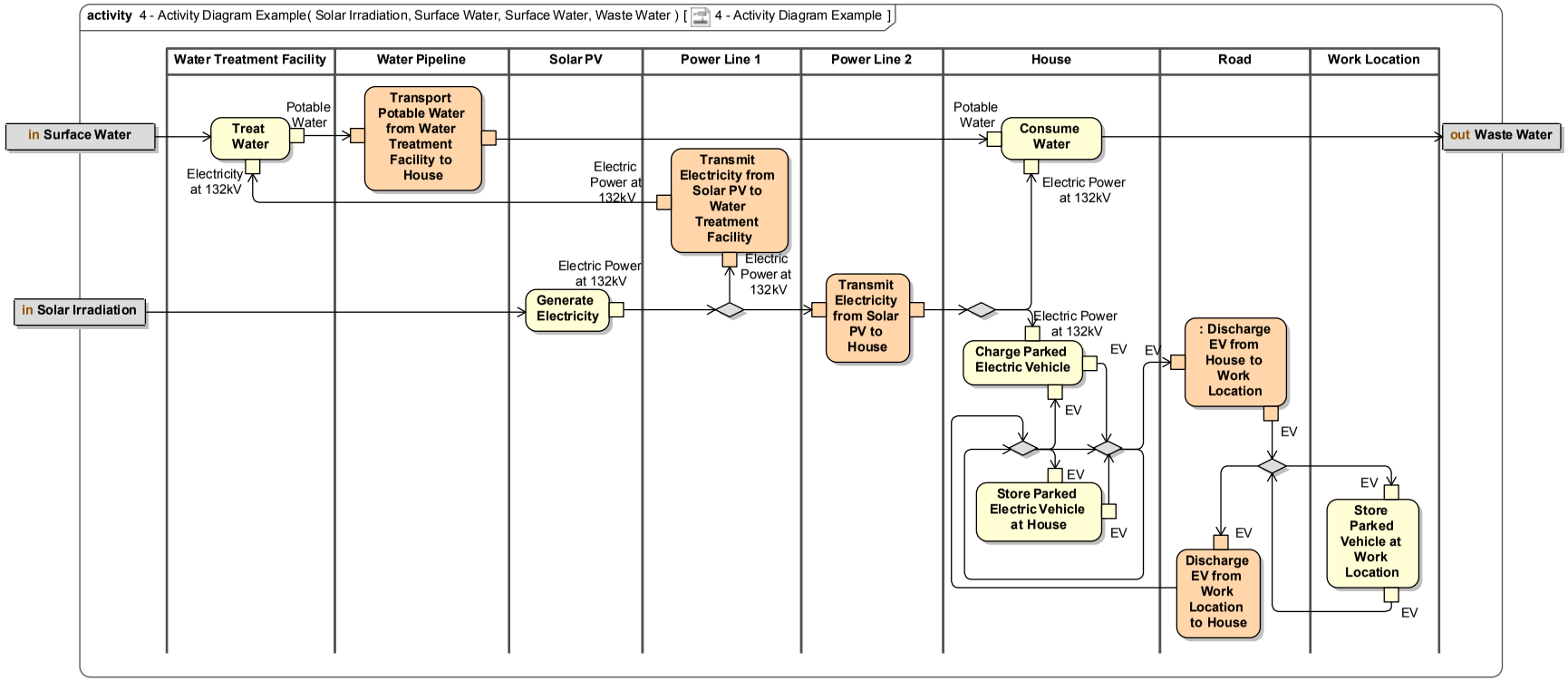


Fig. 3.3: A SysML Activity Diagram: Swim lanes allocate function to form for the 4-node smart city network as presented in Figure 3.1. The network consists of transportation, electricity, and water infrastructure.

3.1.1 System Form

Hetero-functional graph theory defines a set of system resources R as form elements. These are then classified into three types. $R = M \cup B \cup H$, where M are the transformation resources (or machines), B are the independent buffers, and H are the transporting resources[2].

Definition 3.1 – Transformation Resource: A resource $r \in R$ is a transformation resource $m \in M$ iff it is capable of one or more transformation processes on one or more operands and it exists at a unique location in space. ■

Definition 3.2 – Independent Buffer: A resource $r \in R$ is an independent buffer $b \in B$ iff it is capable of storing one or more operands, is not able to transform them or transport them to another location, and exists at a unique location in space. ■

Definition 3.3 – Buffer: A resource $r \in R$ is a buffer $b_s \in B_S$ iff it is capable of storing one or more operands at a unique location in space. $B_S = M \cup B$. ■

Definition 3.4 – Transportation Resource: A resource $r \in R$ is a transportation resource $h \in H$ iff it is capable of transporting one or more operands between an origin and a distinct destination, without transforming these operands. ■

Ontological Remark 3.1. *The classification of system resources into three different types originates from the field of production systems where each type of resource is conceptualized to have its own type of value. Transformation resources are often times referred to as "value adding," transportation resources gain value from logistical necessity, and independent buffers are often explicitly minimized. The distinction between these different types of resources serves to enhance the ontological* lucidity *of hetero-functional graph theory with respect to the conceptualization of value and avoid construct overload. Furthermore, the use of iff conditionals in the definition of these types of resources guarantees adherence to all four ontological properties.* ■

[2]The resource definitions presented above were mildly revised from those found in [2, 6, 14, 17, 20, 98] so as to maintain the integrity of the ontological argument presented throughout the thesis.

3.1.2 System Function

Next, hetero-functional graph theory defines a set of physical system processes P as functional elements.

Definition 3.5 – System Process [104, 239]: An activity that transforms a predefined set of inputs into a predefined set of outputs. ■

The system processes are classified into two types. $P = P_\mu \cup P_\eta$, where P_μ are the transformation processes[3], and P_η are the transportation processes between buffers $B_S = M \cup B$[4].

Definition 3.6 – Transformation Process: A process is a transformation process $p_{\mu j} \in P_\mu$ iff it is capable of transforming one or more properties of a set of operands into a distinct set of output properties in place. It's syntax is:

$$\{\text{transitive verb, operands}\} \rightarrow \{\text{outputs}\} \tag{3.1}$$

■

Definition 3.7 – Transportation Process: A process is a transportation process $p_{\eta u} \in P_\eta$ iff it is capable of transporting one or more operands between an origin buffer $b_{sy_1} \in B_S$ to a destination buffer $b_{sy_2} \in B_S$ according to the following convention of indices [2,6,14,17,20][5]:

$$u = \sigma(B_S)(y_1 - 1) + y_2 \tag{3.2}$$

Its syntax is:

$$\{\text{transport, operands,origin,destination}\} \rightarrow \{\text{outputs,destination}\} \tag{3.3}$$

■

[3]The transformation processes are assumed to transform the operand in place.

[4]The system process definitions presented above were mildly revised from those found in [2,6,14,17,20,98] so as to maintain the integrity of the ontological argument presented throughout the thesis.

[5]Note that a "storage process" is merely a transportation process with the same origin and destination.

Ontological Remark 3.2. *The classification of system processes into two different types originates from the field of production systems where each type of processes is conceptualized to have its own type of value. Transformation processes are often times referred to as "value adding," while transportation processes gain value from logistical necessity. The distinction between these different types of processes serves to enhance the ontological* lucidity *of hetero-functional graph theory with respect to the conceptualization of value and avoid construct overload. Furthermore, the use of iff conditionals in the definition of these types of processes guarantees adherence to all four ontological properties.* ■

The Independence Axiom (2.1) requires the mutual exclusivity of system processes.

Theorem 3.1 – Mutual Exclusivity of System Processes: A lucid representation of system processes as a domain conceptualization distinguishes between two system processes as modeling primitives with different sets of inputs and outputs. ■

Proof 3.1. *Assume that the inputs or outputs of a system process conceptualization are distinct from the inputs or outputs of another system process conceptualization. By Definition 3.5, these two system process conceptualizations are distinct. If a single system process modeling primitive is used to represent both of these two system process conceptualizations, then it is overloaded and by definition lucidity is violated. Consequently, two system process modeling primitives are required.* ■

The above defined set of transportation processes (Defn. 3.7), however, does not distinguish the multiple ways in which an operand is transported between origin and destination. Therefore, at times, it is necessary to introduce a set of *refined* transportation processes $P_{\bar{\eta}}$ which are defined individually as a feasible combination of a transportation process $p_\eta \in P_\eta$ and a holding process $p_\gamma \in P_\gamma$.

Definition 3.8 – Holding Process: A process is a holding process $p_{\gamma g} \in P_\gamma$ iff it holds one or more operands during the transportation from one buffer to another. ■

In order to maintain the independence axiom and the mutual exclusivity of the system

processes (Theorem 3.1), holding processes are specified so as to distinguish between transportation processes that:

- Have *different* operands,
- Hold a given operand in a *given way*, or
- Change the *state* of the operand.

Note that the last requirement for holding processes give them a potentially transformative nature[6]. Furthermore, in differentiating transportation processes by holding process, it is important to retain one holding process that has no transformative effect on its operand.

The set of refined transportation processes can be expressed as a combination of the set of transportation processes and the set of holding processes:

$$P_{\bar{\eta}} = P_{\gamma} \bigtimes P_{\eta} \tag{3.4}$$

where $\bigtimes$ is the Cartesian Product [240].

Definition 3.9 – Refined Transportation Process: A process is a refined transportation process $p_{\bar{\eta}\varphi} \in P_{\bar{\eta}}$ iff it is capable of transporting one or more operands between an origin buffer $b_{s\gamma_1} \in B_S$ to a destination buffer $b_{s\gamma_2} \in B_S$ while it is realizing holding process $p_{\gamma g} \in P_{\gamma}$. Its syntax is:

$$\{\text{transport, operands, origin, destination,while transitive verb}\} \rightarrow \{\text{outputs, destination}\} \tag{3.5}$$

■

In systems where holding processes are required, the complete set of system processes becomes: $P = P_{\mu} \cup P_{\bar{\eta}}$.

[6]While holding processes have been a part of hetero-functional graph theory since its inception, this work is the first to allow for the holding processes to change the state of the operand and have a potentially transformative nature. As later chapters will demonstrate, this expansion of the meaning of holding processes is necessitated by interdependent smart city infrastructure applications where a given operand changes its state as it moves from one location to another.

The introduction of holding processes of a transformative nature does require special attention. The use of hetero-functional graph theory must now take care to not double count transformation processes and refined transportation processes with a transformative nature that occur in a single location. This work uses transformation processes (rather than holding processes of a transformative nature) by default. It only uses transformative holding processes under two conditions: (1) they are required with transportation processes with a distinct origin and destination, or (2) the value-related operand [241] of the process (e.g. pumped water in a water distribution system) remains in place while a second operand is transformed (e.g. electricity). Here, the transformative nature of such a refined transportation process is modeled to occur at a single location in space. For example, while an electric pump may pump water from one distinct location to another, the transformative nature of consuming electricity occurs at a single point in the electric distribution system. Finally, resources that are capable of such refined transformation processes with a transformative nature would now be classified as transformation resources by virtue of Definition 3.1.

Based on the discussion above, one could derive that a separate set of transformation processes P_{μ} is no longer necessary as the holding processes can introduce transformative nature to a buffering transporting processes. However, transformation processes are *essential* to represent operands crossing the system boundary. When the operand enters the system, it seems that the operand appears from “nowhere." The operand is transported across the system boundary, and appears the moment it enters the system. The transportation process has an origin outside the system boundary, which consequently does not exist. As a result, these processes are necessarily represented by transformation processes, as if these operands were generated or consumed at the point of entry or departure. The refined transportation processes of a transformative nature convert the state of the transported operand, but the transported operand is always retained in the system. Heat losses or waste can be included as an output operand of the process[7], but can only leave the system via a transformation

[7]These outputs may also be neglected when they are not desired in the model.

Table. 3.2: System Processes & Resources

System Processes		System Resources	
P_μ	Transformation Processes	M	Transformation Resources
P_η	Transportation Processes	H	Transportation Resources
P_γ	Holding Processes	B	Independent Buffers
$P_{\bar{\eta}} = P_\gamma \bigtimes P_\eta$	Refined Transportation Processes	$B_S = M \cup B$	Buffers
$P = P_\mu \cup P_{\bar{\eta}}$	System Processes	$R = B_S \cup H$	System Resources

process.

In keeping with the ontological properties described in the previous chapter, hetero-functional graph theory requires that the set of system resources and processes be *mutually exclusive and collectively exhaustive* descriptions of system form and function respectively. The overview of system processes and resources is provided in Table 3.2.

Ontological Remark 3.3. *The first two sections (Sections 3.1.1 and 3.1.2) imply an underlying meta-architecture. In hetero-functional graph theory, the primitives of the meta-architecture of system form is a set of resources; classified as transportation resources H and buffers B_S which are in turn classified as transformation resources M and independent buffers B. This classification is depicted in Figure 3.4. Meanwhile, the meta-architecture of system function is a set of processes classified as transformation processes P_μ, transportation processes P_η, and holding processes P_γ. It is depicted in Figure 3.5 as a SysML activity*

diagram. Here, all potential sequences of processes are indicated because there is no a priori *reason at the meta-level to prevent one process from following another.* ■

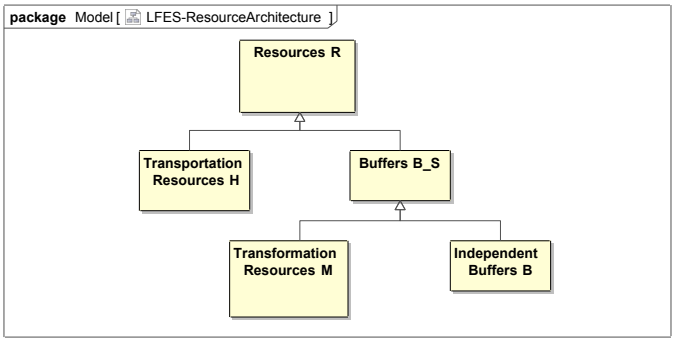


Fig. 3.4: A SysML Block Diagram: The meta-architecture of the system form of a LFES.

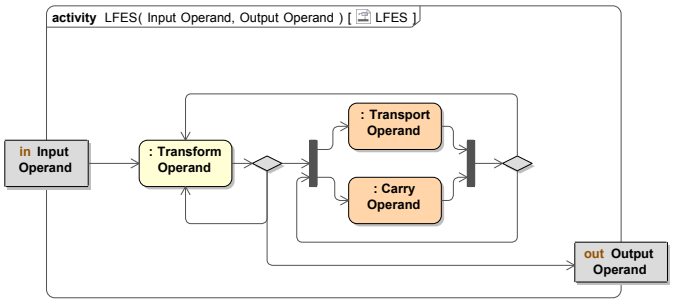


Fig. 3.5: A SysML Activity Diagram: The meta-architecture of the system function of a LFES.

Example 3.2: The SysML block diagram in Figure 3.2 represents a simplified model of a smart city infrastructure system and is now modeled using hetero-functional graph theory. The system resources R are M={Water Treatment Facility, Solar PV, House}, B={Work Location}, H={Water Pipeline, Power Line, Road}. Now, revisit the activity diagram with swim lanes in Figure 3.3. The activity diagram has 2 inputs, 1 output, and 11 activities allocated to 8 resources. First, the set of transformation processes is P_{μ} = {Treat water, consume water, generate electricity}. These three transformation processes either have an input as operand, or their output is also a system output. Second, the set of transportation processes P_{η} is defined as all transportation processes between the system buffers B_S. This example has four buffers ($B_S = M \cup B$) and consequently, the example has 16 transportation

processes (e.g. transport from M_1 to M_1, transport from M_1 to M_2, etc.). Third, the set of holding processes serves to distinguish transportation processes. The transportation processes in this example are distinguished for operand and transformative nature. The resulting set of holding processes is P_γ = {Carry potable water, Carry electricity at 132kV, Charge EV, Discharge EV, Carry EV}. Finally, the set of refined transportation processes is calculated using Equation 3.4. The resulting set has a size of 80; it contains all unique combinations of the elements in P_γ with the elements in P_η. Note that P_η and $P_{\bar{\eta}}$ thus contain all possible (refined) transportation processes. ■

3.1.3 Allocation of System Function onto System Form

In hetero-functional graph theory, the allocation of system processes to system resources is captured in the "design equation" [2]:

$$P = J_S \odot R \tag{3.6}$$

where J_S is the system knowledge base, and $\odot$ is "matrix boolean multiplication" [2, 242].

Definition 3.10 – System Knowledge Base [2, 6, 14, 17, 20, 98]: [8] A binary matrix J_S of size $\sigma(P) \times \sigma(R)$ whose element $J_S(w, v) \in \{0, 1\}$ is equal to one when action $e_{wv} \in \mathcal{E}$ (in the SysML sense) exists as a system process $p_w \in P$ being executed by a resource $r_v \in R$. The $\sigma()$ notation gives the size of a set. ■

In other words, the system knowledge base forms a bipartite graph [14] between the set of system processes and the set of system resources.

It is important to note that the system knowledge base views the set of system processes at the reference architecture level and the set of system resources at the instantiated architecture level [98]. As mentioned in Section 2.3, the mapping of the instantiated functional

[8] The system knowledge base was first introduced in 2006 [17] simply as a binary matrix and was later named as a knowledge base in [2, 6, 20, 98] in recognition of its use in the Axiomatic Design literature [3].

architecture to the instantiated physical architecture is *necessarily* one-to-one in large flexible engineering systems. In using the system processes defined at the reference architecture, the system knowledge base provides further insight into process redundancy [2, 38, 98].

$$\mathcal{R}_w = \sum_{v}^{\sigma(R)} J_S(w, v) \tag{3.7}$$

Furthermore, it is possible to aggregate resources R into aggregated resources $\overline{R}$ [2, 6, 20, 38, 98]:

$$\overline{R} = \Xi \circledast R \tag{3.8}$$

where Ξ is an aggregation matrix and $\circledast$ is the aggregation operator.

Definition 3.11 – Aggregation Operator $\circledast$ [2, 6, 20, 38, 98]:

$$C(i,k) = \bigcup_{j} a(i,j) \odot b(j,k) = A \circledast B \tag{3.9}$$

■

Some aggregated resources $\overline{R}$ can be considered *flexible* in that they are capable of executing more than one system process [2, 38, 98].

$$\mathcal{F}_v = \sum_{w}^{\sigma(P)} J_S(w, v) \tag{3.10}$$

Process redundancy $\mathcal{R}$ and resource flexibility $\mathcal{F}$ are valuable measures that directly impact a system's reconfigurability and resilience [14, 15].

The design equation is also applicable to the different different types of processes and

resources [2, 6, 14, 17, 20]:

$$P_\mu = J_M \odot M \tag{3.11}$$

$$P_\eta = J_H \odot R \tag{3.12}$$

$$P_\gamma = J_\gamma \odot R \tag{3.13}$$

where J_M is the transformation knowledge base, J_H is the transportation knowledge base, and J_γ is the holding knowledge base. The refined transportation knowledge base $J_{\bar{H}}$ is constructed as follows [2, 8, 14, 22, 30, 31][9]:

$$J_{\bar{H}} = \left[J_\gamma \otimes \mathbb{1}^{\sigma(P_\eta)}\right] \cdot \left[\mathbb{1}^{\sigma(P_\gamma)} \otimes J_H\right] \tag{3.14}$$

where $\otimes$ is the Kronecker product, $\cdot$ is the hadamard product, and $\mathbb{1}^n$ is a ones-vector of length n.

Note that the mathematical structure of $J_{\bar{H}}$ effectively stratifies several copies of J_H so that system transportation processes are differentiated by the holding processes in P_γ. These holding processes are introduced in either of the three scenarios, as mentioned previously regarding the set of refined transportation processes.

In previous two-operand system literature, a more practical approach was used, such that the operand specific refined transportation knowledge bases were simply concatenated. The resources that appear in both systems were merged after concatenation. This approach is equally correct, potentially quicker for two-operand systems, but not practical for multi-operand systems. For example, the concatenation of transportation knowledge bases was used in electrified transportation systems [30–32] and microgrid-enabled production systems (see Chapter 5 of this dissertation) [35]. In greater detail, in [31, 32], the holding knowledge base differentiated between several different modes of travel within an electrified transporta-

[9] While more recent references represent $J_{\bar{H}}$ in matrix form, the original formula in [2] based upon scalars remains useful in large-scale computational applications where memory is limited.

tion infrastructure. The transportation system knowledge base for a system that transports *multiple* operands is a concatenation of the transportation knowledge bases for each holding process.

$$J_{\bar{H}} = \begin{bmatrix} J_{\bar{H}1} \\ \vdots \\ J_{\bar{H}n} \end{bmatrix} \tag{3.15}$$

In either case, the system knowledge base J_S is [2, 6, 14, 17, 20]:

$$J_S = \left[\begin{array}{c|c} J_M & \mathbf{0} \\ \hline & J_{\bar{H}} \end{array}\right] \tag{3.16}$$

Hetero-functional graph theory also differentiates between the *existence* and the *availability* of physical capabilities in the system [2, 22]. While the former is described by the system knowledge base, the latter is captured by the system constraints matrix.

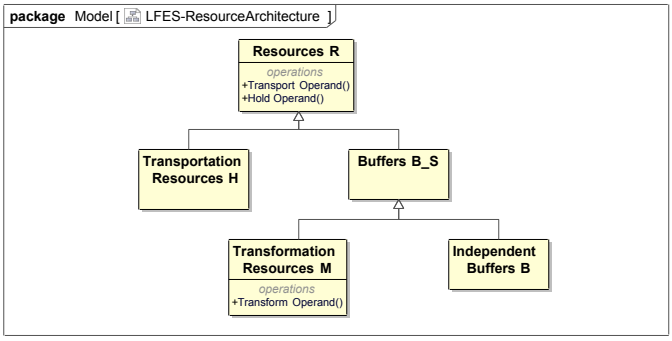


Fig. 3.6: A SysML Block Diagram: The meta-architecture of the allocated architecture of a LFES from a system form perspective.

Ontological Remark 3.4. *Building on the meta-architecture of the system processes and system resources, the mapping of system function to system form at the meta-level is indicated by the operations in Figure 3.6 or by the swim lanes in Figure 3.7. Transformation resources are able to execute transformation processes while all resources are able to execute transportation processes.* ■

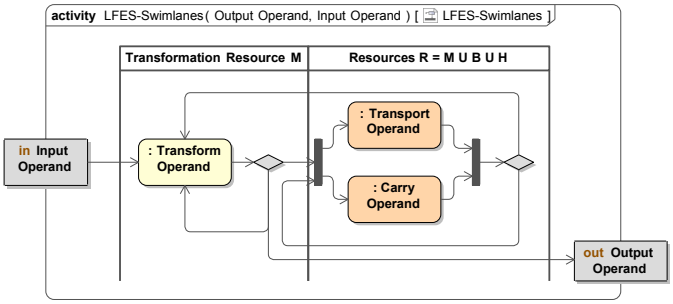


Fig. 3.7: A SysML Activity Diagram: The meta-architecture of the allocated architecture of a LFES from a system function perspective.

Definition 3.12 – System Constraints Matrix [2, 6, 14, 17, 20, 98]: [10] A binary matrix K_S of size $\sigma(P) \times \sigma(R)$ whose element $K_S(w,v) \in \{0,1\}$ is equal to one when a constraint eliminates event e_{wv} from the event set. ■

The system constraints matrix is constructed analogously to the system knowledge base [2, 6, 14, 17, 20].

$$K_S = \left[\begin{array}{c|c} K_M & \mathbf{0} \\ \hline & K_{\bar{H}} \end{array}\right] \tag{3.17}$$

In this regard, the system constraints matrix has a similar meaning to graph percolation [90, 243] and temporal networks [244].

A system's physical capabilities are quantified as structural degrees of freedom.

Definition 3.13 – Structural Degrees of Freedom [2, 6, 14, 17, 20, 98]: [11] The set of independent actions $\mathcal{E}_S$ that completely defines the available processes in a large flexible engineering system. The number of degrees of freedom is given by:

[10] The system constraints matrix was first introduced in [17] simply as a binary matrix and was later named in [2, 6, 14, 20, 98]. In some hetero-functional graph theory references, it is called the scleronomic constraints matrix or the sequence-independent constraints matrix to indicate that it is applied one capability at a time.

[11] The concept of structural degrees of freedom was first introduced in [17]. In some hetero-functional graph theory references, it is called scleronomic or sequence-independent degrees of freedom to indicate that they address capabilities one at a time.

$$DOF_S = \sigma(\mathcal{E}_S) = \sum_{w}^{\sigma(P)} \sum_{v}^{\sigma(R)} [J_S \ominus K_S](w,v) \tag{3.18}$$

$$= \sum_{w}^{\sigma(P)} \sum_{v}^{\sigma(R)} A_S(w,v) \tag{3.19}$$

$$= \langle J_S, \bar{K}_S \rangle_F \tag{3.20}$$

where $\ominus$ is boolean subtraction. The system constraints matrix limits the availability of degrees of freedom in the system knowledge base to create the system concept A_S. Note that the transformation degrees of freedom DOF_M and the refined transportation degrees of freedom DOF_H are calculated similarly [2, 6, 17, 20]:

$$DOF_M = \sum_{j}^{\sigma(P_\mu)} \sum_{m}^{\sigma(M)} [J_M \ominus K_M](j,m) \tag{3.21}$$

$$DOF_H = \sum_{u}^{\sigma(P_{\bar{\eta}})} \sum_{v}^{\sigma(R)} [J_{\bar{H}} \ominus K_{\bar{H}}](u,v) \tag{3.22}$$

■

Example 3.3: The term structural degrees of freedom is best viewed as a generalization of kinematic degrees of freedom (or generalized coordinates) [2, 6, 10, 14, 17, 20]. The number of degrees of freedom is given by [245]:

$$DOF = n_d * n_l - n_k \tag{3.23}$$

where n_l is the number of links, $n_d = 6$ is the number of primitive coordinates, and n_k is the number of applied scleronomic (i.e time independent) constraints that confine motion. For example, consider the two-bar linkage shown in Figure 3.8. $n_l = 2$ and $n_k = 2 * 5 = 10$.

Consequently $DOF = 2$. If any individual constraint is assumed to affect only a single combination of link and coordinate, then Equation 3.23 can be written as [2, 6, 14, 17, 20, 98]:

$$DOF = \sum_{i}^{n_d} \sum_{j}^{n_l} [J_S \ominus K_S](i,j) = \sum_{i}^{n_d} \sum_{j}^{n_l} A_S(i,j) \tag{3.24}$$

Consequently, in hetero-functional graph theory, mechanical links as elements of form are analogous to system resources and primitive coordinates as descriptors of function are analogous to system processes [2, 6, 10, 14, 17, 20]. ■

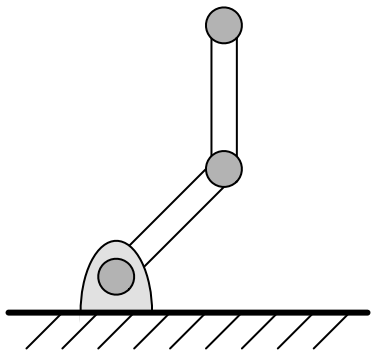


Fig. 3.8: A Two Bar Linkage System.

Example 3.4: Building off of Example 3.2 and returning to smart city applications, Figures 3.2 and 3.3 can be quantified for their system knowledge base J_S and system constraints matrix K_S. Figure 3.9 provides a visual overview of the system capabilities, presented at the location of their physical resources. Figure 3.10 provides the matrix representation of each of the knowledge bases.

The *transformation knowledge base* J_M maps the transformation processes to the transformation resources. The transformation knowledge base consequently has size $\sigma(P_\mu) \times \sigma(M) = 3 \times 3$, with three filled elements representing the transformation capabilities.

The *transportation knowledge base* J_H maps the transportation processes to the system resources, resulting in a matrix of size $\sigma(P_\eta) \times \sigma(R) = 16 \times 8$. Matrix J_H contains seven

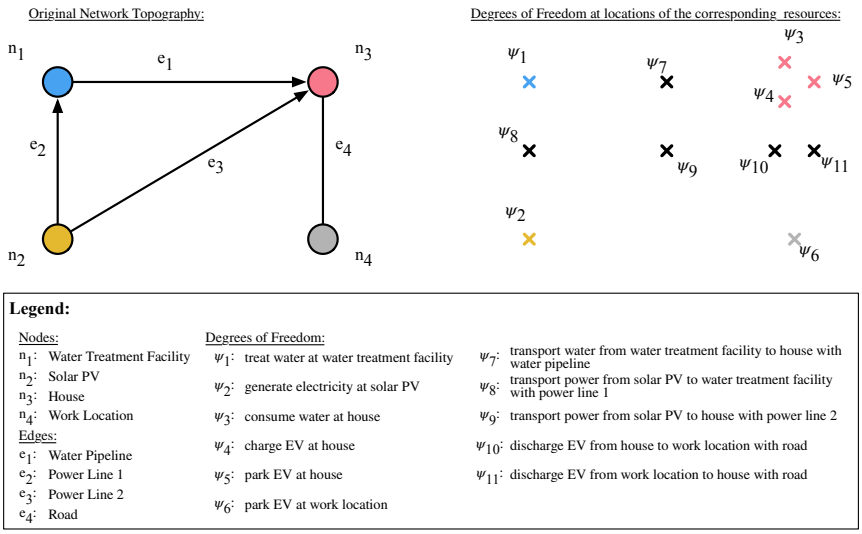


Fig. 3.9: Degrees of Freedom in the Example Network: A visual comparison of the original network topography on the left, and the system degrees of freedom on the right.

filled elements, representing the transportation capabilities.

The *holding knowledge base* J_γ serves to refine the transportation processes. It distinguishes the operands, the way of holding operands, and the way of holding operands that could potentially change the state of the operand. The size of J_γ is $\sigma(P_\gamma) \times \sigma(R) = 5 \times 8$, with nine filled elements.

The *refined transportation knowledge base* $J_{\bar{H}}$ is calculated using Equation 3.14. Its size is $\sigma(P_{\bar{\eta}}) \times \sigma(R) = 80 \times 8$, and it contains eight filled elements.

The *system knowledge base* J_S then follows as a concatenation of J_M and $J_{\bar{H}}$, following Equation 3.16. The system knowledge base consequently has the size $\sigma(P) \times \sigma(R) = 83 \times 8$, with 11 filled elements that represent the system capabilities.

In this example, all capabilities are available and K_S is thus an all-zeros matrix with the same size as J_S. The number of available capabilities (i.e. degrees of freedom) is calculated to be 11 using Equation 3.18. In this case, the number of capabilities is equal to the number of actions in the activity diagram in Figure 3.3. ■

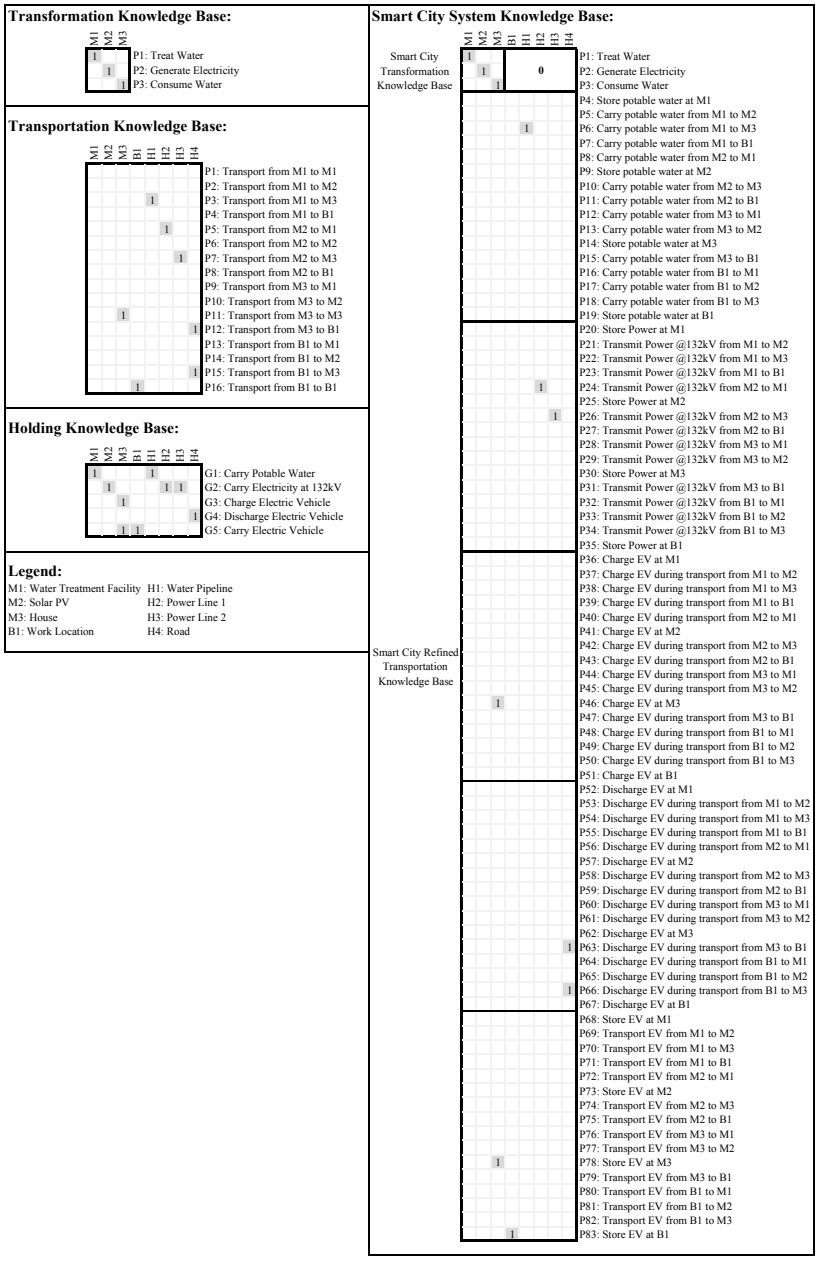


Fig. 3.10: Transformation, Transportation, Holding, and System Knowledge Bases Corresponding to Figures 3.2 and 3.3.

3.2 Hetero-functional Adjacency Matrix

In the previous subsection, system capabilities are described as sentences. In this subsection, the capabilities are connected into serial and parallel arrangements. This is similar to when sentences are connected together to form paragraphs and stories. The hetero-functional

adjacency matrix represents the logical order of physical capabilities in a system [14].

In SysML, both the block definition diagram and the activity diagram with swim lanes have the ability to represent the sequence of capabilities [68, 69]. Sequence in the block definition diagram is represented at the meta-architecture level, as physical continuity applies to all engineering systems. Figure 3.11 represents the physical sequence constraints with associations in green. Sequence in the activity diagram uses the functional interactions between the actions to link the capabilities together. For example, Figure 3.3 is interpreted as: “The solar PV panel generates electricity from solar irradiation. Power line 2 transmits electricity to the house."

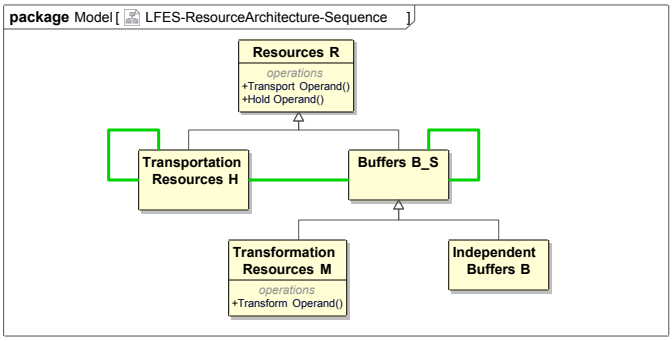


Fig. 3.11: A SysML Block Diagram: System sequence associations are added to the meta-architecture of the allocated architecture of a LFES from a system form perspective.

In hetero-functional graph theory, the hetero-functional adjacency matrix A_ρ is introduced to represent the sequence of physical capabilities [14, 25, 27, 31, 35]. Much like the system concept A_S, the hetero-functional adjacency matrix A_ρ arises from a Boolean difference [14, 25, 27, 31, 35].

$$A_\rho = J_\rho \ominus K_\rho \tag{3.25}$$

where J_ρ is the system sequence knowledge base and K_ρ is the system sequence constraints matrix.

Definition 3.14 – System Sequence Knowledge Base [14, 25, 27, 31, 35]: [12] A square

[12] One important change to hetero-functional graph theory over the years has been in the system sequence knowledge base. Originally, it was called the rheonomic knowledge base and defined as a binary matrix of size $\sigma^2(P) \times \sigma^2(R)$ to mirror its sequence-independent counterpart [2, 6, 17, 20]. Later works have adopted the definition presented above as a methodological development.

binary matrix J_ρ of size $\sigma(R)\sigma(P) \times \sigma(R)\sigma(P)$ whose element $J_\rho(\chi_1, \chi_2) \in \{0, 1\}$ is equal to one when string z_{χ_1,χ_2} exists, where index $\chi_i \in [1, \ldots, \sigma(R)\sigma(P)]$. It is calculated as:

$$J_\rho = A_S^V A_S^{VT} \tag{3.26}$$

$$J_\rho = \left[J_S \cdot \bar{K}_S\right]^V \left[J_S \cdot \bar{K}_S\right]^{VT} \tag{3.27}$$

where $()^V$ is shorthand for vectorization (i.e. vec()). ■

Definition 3.15 – System Sequence Constraints Matrix [14, 25, 27, 31, 35]: [13] A square binary constraints matrix K_ρ of size $\sigma(R)\sigma(P) \times \sigma(R)\sigma(P)$ whose elements $K(\chi_1, \chi_2) \in \{0, 1\}$ are equal to one when string $z_{\chi_1\chi_2} = e_{w_1 v_1} e_{w_2 v_2} \in \mathcal{Z}$ is eliminated. ■

Note that J_ρ creates all potential sequences of the capabilities in A_S. However, there are only certain pairs of capabilities that are feasible. The system sequence constraints matrix K_ρ serves to eliminate the infeasible pairs.

Sequence dependent constraints in K_ρ may be imposed on either functional interactions and/or physical interfaces. Consequently, there are several considerations when calculating K_ρ. In the case of functional interactions within an instantiated functional architecture, the *absence* of an arrow between any two allocated actions in a SysML activity diagram imposes a set of functional constraints within K_ρ. Similarly, in the case of physical interfaces within an instantiated physical architecture, the *absence* of an association link in a SysML block diagram imposes a set of physical constraints within K_ρ (without regard for system function). *Domain-specific* sequence dependent constraints in K_ρ may be imposed directly at the reference architecture level, be they from arrows in the SysML activity diagram or from association links in the SysML block diagram. Such design pattern constraints effectively impose topological exchanges of matter, energy, information, people, or money [1][14].

[13] Similarly, the system sequence constraints matrix was first called the rheonomic knowledge base and defined as a binary matrix of size $\sigma^2(P) \times \sigma^2(R)$. Later works have adopted the definition presented above as a methodological development.

[14] In [241], Crawley distinguishes between topological connections and spatial relationships as two types of interactions. Because spatial relationships are effectively a type of requirement and are not typically addressed in the network sciences, hetero-functional graph theory neglects their treatment here as well.

Table. 3.3: Types of Sequence-Dependent Production Degree of Freedom Measures [8, 14, 25]

Type	Measure	Process	Resource	Sequence-Dependent Knowledge Base	Constraint Matrix	Perpetual Constraint	Measure Function
I	$DOF_{MM\rho}$	$P_\mu P_\mu$	M, M	$J_{MM\rho} = \left[J_M \cdot \bar{K}_M\right]^V \left[J_M \cdot \bar{K}_M\right]^{VT}$	$K_{MM\rho}$	$m_1 = m_2$	$\langle J_{MM\rho}, \bar{K}_{MM\rho} \rangle_F$
II	$DOF_{MH\rho}$	$P_\mu P_\eta$	M, R	$J_{MH\rho} = \left[J_M \cdot \bar{K}_M\right]^V \left[J_H \cdot \bar{K}_H\right]^{VT}$	$K_{MH\rho}$	$m_1 - 1 = (u_1 - 1)/\sigma(B_S)$	$\langle J_{MH\rho}, \bar{K}_{MH\rho} \rangle_F$
III	$DOF_{HM\rho}$	$P_\eta P_\mu$	R, M	$J_{HM\rho} = \left[J_H \cdot \bar{K}_H\right]^V \left[J_M \cdot \bar{K}_M\right]^{VT}$	$K_{HM\rho}$	$m_1 - 1 = (u_1 - 1)\&\sigma(B_S)$	$\langle J_{HM\rho}, \bar{K}_{HM\rho} \rangle_F$
IV	$DOF_{HH\rho}$	$P_\eta P_\eta$	R, R	$J_{HH\rho} = \left[J_H \cdot \bar{K}_H\right]^V \left[J_H \cdot \bar{K}_H\right]^{VT}$	$K_{HH\rho}$	$(u_1 - 1)\%\sigma(B_S) = (u_2 - 1)/\sigma(B_S)$	$\langle J_{HH\rho}, \bar{K}_{HH\rho} \rangle_F$
ALL	DOF_ρ	PP	R, R	$J_\rho = \left[J_S \cdot \bar{K}_S\right]^V \left[J_S \cdot \bar{K}_S\right]^{VT}$	K_ρ	All of the Above	$\langle J_{S\rho}, \bar{K}_{S\rho} \rangle_F$

Finally, in the absence of either an explicitly stated physical or functional architecture be it at the instantiated system or reference level, the system sequence constraints matrix K_ρ must still address continuity as a meta-level property of physical architectures (in general). In other words, given two capabilities, the output of the first must occur at the location of the input of the second. These constraints are described quantitatively in Table 3.3 [14]. In such a case, the construction of the system sequence constraints matrix requires the tracking of all four constraints for each of the pairs of capabilities in the system. A straightforward way of calculating this matrix is a scalar implementation using FOR loops while adhering to the following relationships of indices $\chi = \sigma(P)(v-1) + w$. For a transformation process p_w, $w = j \; \forall j = [1 \ldots \sigma(P_\mu)]$. For a transportation process p_w, $w = [\sigma(P_\eta)(g-1) + u] + \sigma(P_\mu)$ $\forall g = [1 \ldots \sigma(P_\gamma)], \forall u = [1 \ldots \sigma(P_\eta)]$ [8, 14, 25].

A system's sequences are quantified as sequence-dependent degrees of freedom.

Definition 3.16 – Sequence-Dependent Degrees of Freedom [14, 25, 27, 31, 35]: [15] The set of independent pairs of actions $z_{\chi_1\chi_2} = e_{w_1 v_1} e_{w_2 v_2} \in \mathcal{Z}$ of length 2 that completely

[15] Similarly, in earlier works, the sequence-dependent degrees of freedom measure was first called the rheonomic degrees of freedom and defined over a binary matrix of size $\sigma^2(P) \times \sigma^2(R)$. Despite this methodological difference, it is easy to show that the number of sequence dependent degrees of freedom is the same regardless of the choice of calculation method.

describe the system language. The number is given by:

$$DOF_\rho = \sigma(\mathcal{Z}) = \sum_{\chi_1}^{\sigma(R)\sigma(P)} \sum_{\chi_2}^{\sigma(R)\sigma(P)} [J_\rho \ominus K_\rho](\chi_1, \chi_2) \tag{3.28}$$

$$= \sum_{\chi_1}^{\sigma(R)\sigma(P)} \sum_{\chi_2}^{\sigma(R)\sigma(P)} [A_\rho](\chi_1, \chi_2) \tag{3.29}$$

■

For systems of substantial size, the size of the hetero-functional adjacency matrix may be challenging to process computationally. However, the matrix is generally very sparse. Therefore, projection operators are used to eliminate sparsity by projecting the matrix onto a ones vector [31, 35]. This is demonstrated below for J_S^V and A_ρ:

$$\mathbb{P}_S J_S^V = \mathbb{1}^{\sigma(\mathcal{E}_S)} \tag{3.30}$$

$$\mathbb{P}_S A_\rho \mathbb{P}_S^T = \widetilde{A}_\rho \tag{3.31}$$

where $\mathbb{P}_S$ is a (non-unique) projection matrix for the vectorized system knowledge base and the hetero-functional adjacency matrix [31, 35].

The number of sequence dependent degrees of freedom for the projected hetero-functional adjacency matrix can be calculated as:

$$DOF_\rho = \sigma(\mathcal{Z}) = \sum_{\psi_1}^{\sigma(\mathcal{E}_S)} \sum_{\psi_2}^{\sigma(\mathcal{E}_S)} [\widetilde{A}_\rho](\psi_1, \psi_2) \tag{3.32}$$

where $\psi \in [1, \ldots, \sigma(\mathcal{E}_S)]$.

Example 3.5: This example continues where Examples 3.2 and 3.4 left off. Recall that Example 3.2 derived the sets of the 4-node example network, and that Example 3.4 continued by calculating the knowledge bases. This example completes the structural model for the

4-node network by calculating the hetero-functional adjacency matrix A_ρ. First, the system sequence knowledge base J_ρ is calculated using Equation 3.26. Recall that no constraints were imposed on the knowledge base of the 4-node network, and consequently, size of matrix A_S is $\sigma(P) \times \sigma(R) = 83 \times 8$, with 11 filled elements. The resulting size of matrix J_ρ is $\sigma(R)\sigma(P) \times \sigma(R)\sigma(P) = 664 \times 664$, with $11^2 = 121$ filled elements.

Second, each of the elements in the system sequence knowledge base can be checked for its adherence to the physical sequence constraints as defined in Table 3.3. Additionally, the sequences must adhere to the functional sequences as defined in the activity diagram in Figure 3.3 on Page 63. By directly checking the sequences for adherence to the constraints, the calculation of the full constraints matrix K_ρ is avoided, and the hetero-functional adjacency matrix A_ρ is calculated directly. Matrix A_ρ has size $\sigma(R)\sigma(P) \times \sigma(R)\sigma(P) = 664 \times 664$, with 20 filled elements.

Last, the matrix A_ρ is projected and visualized. The projection operation eliminates the row and column sparsity so as to reduce the size of the hetero-functional adjacency matrix (using Equation 3.31). This is especially important for more complex systems. The projected hetero-functional adjacency matrix $\widetilde{A}_\rho$ now has size $\sigma(\mathcal{E}_S) \times \sigma(\mathcal{E}_S) = 11 \times 11$, still with 20 filled elements. The matrix is presented in Figure 3.12, and presented as a network in Figure 3.13. ■

	1	2	3	4	5	6	7	8	9	10	11
1							1				
2								1	1		
3											
4				1	1					1	
5				1	1					1	
6						1					1
7			1								
8	1										
9			1	1							
10						1					1
11				1	1					1	

Legend:

1: treat water at water treatment facility

2: generate electricity at solar PV

3: consume water at house

4: charge EV at house

5: park EV at house

6: park EV at work location

7: transport water from water treatment facility to house with water pipeline

8: transport power from solar PV to water treatment facility with power line 1

9: transport power from solar PV to house with power line 2

10: discharge EV from house to work location with road

11: discharge EV from work location to house with road

Fig. 3.12: Projected Hetero-functional Adjacency Matrix $\tilde{A}_\rho$ for Example 3.5. (Row and column sparsity have been eliminated.)

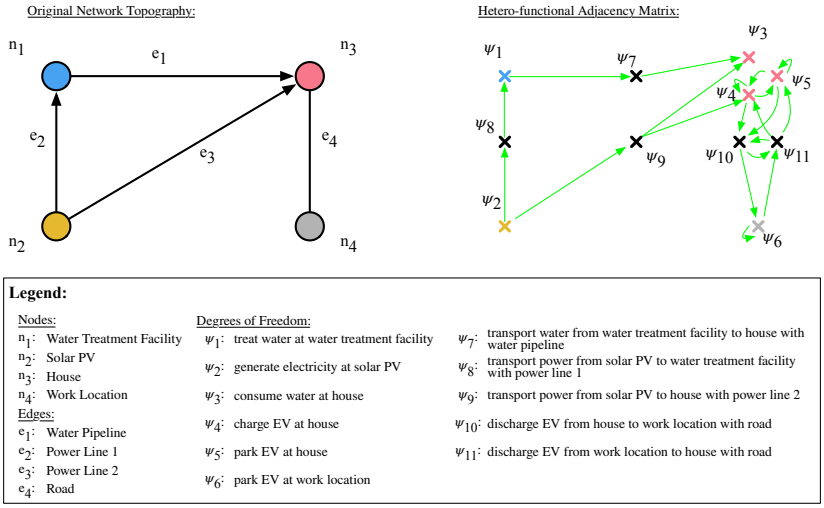


Fig. 3.13: Degrees of Freedom in the Example Network: A visual comparison of the original network topography on the left, and the hetero-functional adjacency matrix on the right.

Ontological Remark 3.5. *It is important to recognize that the hetero-functional adjacency matrix is entirely distinct from the (traditional) adjacency matrices often used in graph theory. While the latter often represents the nodes as some element of form (i.e. a resource) and edges as the flows of matter, energy, or information between them, the hetero-functional adjacency matrix has structural degrees of freedom (or capabilities) as nodes and edges to represent feasible logical sequences. This is an essential difference. While a traditional adjacency matrix assigns only one node to a resource regardless of its functionality, a hetero-functional adjacency matrix assigns a node to each capability that each resource can perform. A traditional adjacency matrix uses edges to represent transportation resources and implicitly assumes a one-to-one relationship with the transportation processes that they execute. A hetero-functional adjacency matrix, in contrast, assigns nodes to transportation capabilities. This directly facilitates an understanding of the redundancy of each transportation process. The explicit treatment of both the functional and physical architectures and their embedded process redundancy and resource flexibility become an integral part of understanding the structure and behavior of integrated smart city infrastructures.* ■

3.3 Controller Agency Matrix

The controller agency matrix serves to differentiate between two systems of equivalent capabilities but different control structure [2, 6, 7, 20]. Consider Figure 3.14 [2]. Both systems have equivalent capabilities; they are able to do the same processes on the same resources. The system on the right, however, requires a centralized controller to perform these capabilities; while the system on the left can do so without the need for such a controller. Ontologically speaking, the controller agency matrix is introduced to ensure the lucidity of the model. In the case of centralized controllers, the set of system resources R is assumed to now include cyber-resources Q. $R = M \cup B \cup H \cup Q$ [2, 6, 7, 20]. Furthermore, the previously identified resources M, B, and H constitute the system's physical resources $R_P = M \cup B \cup H$.

Definition 3.17 – Cyber-Resource [2, 6, 7, 20]: A resource $r \in R$ is a cyber-resource (e.g. controller or decision-making agent) $q \in Q = Q_D \cup Q_I$ iff it is capable of controlling how physical resources perform their system processes. *Dependent* cyber-resources Q_D are integral parts of physical resources (as in the case of embedded controllers). Independent cyber-resources Q_I are stand-alone and external to the physical resources (e.g. centralized controllers & decision-making entities). For the sake of simplicity, each cyber-resource is assumed to be able to execute a single control or decision algorithm $p_Q \in P_Q$. Cyber-resources that can execute multiple decisions are called aggregated cyber-resources $\overline{Q} = \Xi \circledast Q$[16]. ■

The control structure of an engineering system describes the scope of control or jurisdiction that a cyber-resource has over a physical resource. If a physical resource has one or more dependent cyber-resources, then it has its own control or decision-making agency and is usually able to behave autonomously or in coordination with other resources [22, 27]. Other physical resources must instead rely on independent cyber-resources in order to effectively

[16]The definition presented above is a generalization of the one found in earlier works [2, 6, 7, 20] where dependent cyber-resources were neglected and independent cyber-resources were called centralized controllers.

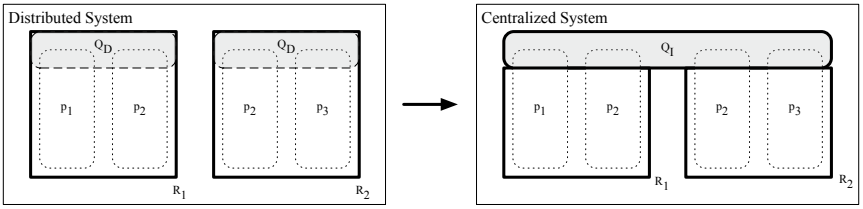


Fig. 3.14: Capabilities with Cyber-Resources. The distributed system on the left has embedded (dependent) controller Q_D, and the centralized system on the right has an independent controller Q_I [2].

control their capabilities.

Ontological Remark 3.6. *The above paragraphs imply a revision to the meta-architecture of a large flexible engineering system. The form primitives of hetero-functional graph theory now include a set of cyber-resources Q. Consequently, the classification depicted in Figure 3.4 is revised to the one depicted in Figure 3.15. Cyber-resources (i.e. controllers and decision-making agents) are added as a class. Dependent cyber-resources are linked to their associated physical resources with decomposition links. However, for modeling simplicity, this is rarely shown unless there is a need to decompose the resource into its physical and cyber parts. In such a case* $Q_D = \emptyset, Q = Q_I$. *Instead, physical resources with their own control or decision-making agency may be viewed as having an association link to itself. Independent cyber-resources are linked to the physical resources that they control with association links. Meanwhile, the primitives of the meta-architecture of system function is a set of processes that now include control and decision algorithms* P_Q. *Consequently, Figure 3.5 is revised to the one shown in Figure 3.16.* ■

In hetero-functional graph theory, the controller agency matrix A_Q defines the control structure of a system.

Definition 3.18 – Controller Agency Matrix [2, 6, 7, 20]: A binary matrix A_Q of size $\sigma(R_P) \times \sigma(R)$, whose element $A_Q(v_1, v_2)$ is equal to one when the resource $r_{v_2} \in R$ has control jurisdiction over the physical resource $r_{v_1} \in R_P$. When $A_Q(v_1, v_1) = 1$, a physical resource $r_{v_1} \in R_P$ is said to control itself (e.g by an embedded controller). When

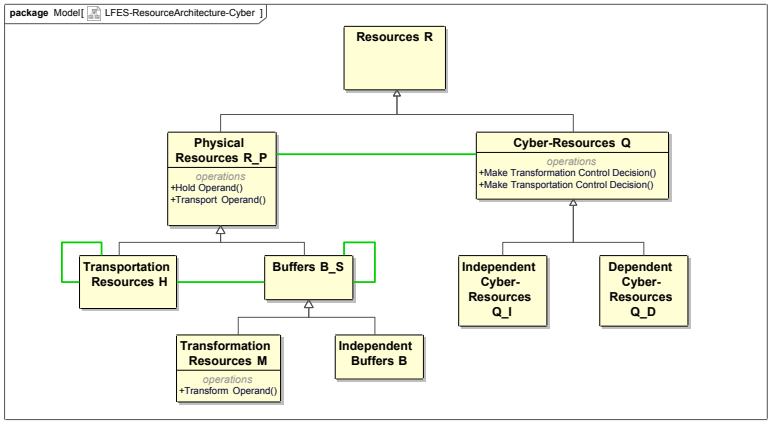


Fig. 3.15: A SysML Block Diagram: The meta-architecture of the system form of a LFES with cyber-resources.

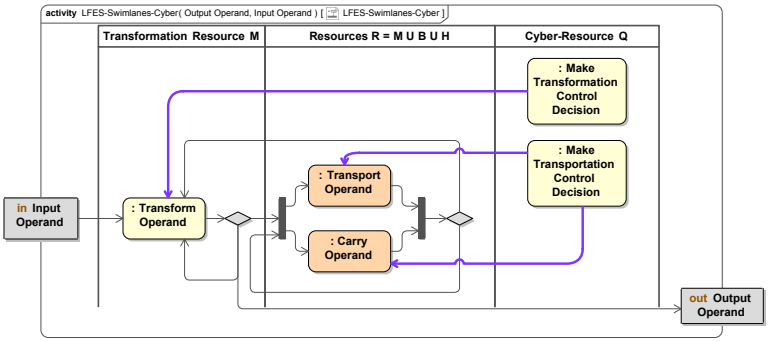


Fig. 3.16: A SysML Activity Diagram: The meta-architecture of the system function of a LFES with control and decision-making algorithms.

$A_Q(v_1, v_2) = 1$ and $v_1 \neq v_2$ then the independent cyber-resource $r_{v_2} \in Q$ is required (i.e. as a centralized controller) to control the physical resource r_{v_1}. In the absence of independent cyber-resources, $Q = \emptyset$, $A_Q = I^{\sigma(R_P)}$. In an engineering system with a *perfectly* centralized control system (i.e. one with a single centralized controller controlling multiple physical resources), $A_Q = [I^{\sigma(R_P)}, \mathbb{1}^{\sigma(R_P)}]$. In an engineering system with only *physical agents*, (i.e. one agent for every physical resource), $A_Q = [I^{\sigma(R_P)}, I^{\sigma(R_P)}]$. In such a case, the physical agents are often treated as *dependent cyber-resources*, and the cyber-physical resources $\overline{R}_P = A_Q \circledast R$ are used in place of the physical resources R_P. In all other cases, $A_Q = [I^{\sigma(R_P)}, \overline{A}_Q]$ where $\overline{A}_Q$ is the independent controller agency matrix. $\overline{A}_Q$ is a bipartite graph that shows the jurisdiction of independent cyber-resources Q over physical resources

R_P. ■

Ontological Remark 3.7. *Hetero-functional graph theory introduces the controller agency matrix in order to introduce ontological lucidity; recognizing that cyber and physical interactions are fundamentally different [128, 246, 247]. Flows of power are fundamentally two way; power can not be transferred from one entity to another without affecting the state of both entities. Flows of matter are either one way or two way, depending on whether the matter-flow is modeled to have an associated power flow. Meanwhile, in most engineering applications information is assumed to be one-way, aphysical, and of negligible power transfer. When information is passed from one entity to another, only the recipient but not the sender is affected. In quantum information processing, however, the information regains its physical characteristics. The recipient and sender are affected by the transfer of information [248]. The controller agency matrix shows* informatic *interactions between the resources R and the physical resources R_P. This is contrasted to the hetero-functional adjacency matrix which shows* physical *interactions between physical capabilities (i.e. structural degrees of freedom).*

When the engineering system has independent cyber-resources, the controller agency matrix is necessary to describe the allocation of system processes to system resources. The "design equation" in Equation 3.6 is then generalized to become:

$$P = J_S \odot (A_Q \circledast R) \tag{3.33}$$

where $R = R_P \cup Q$ and $\circledast$ is the aggregation operator.

In this regard, hetero-functional graph theory recognizes that physical and cyber resources are fundamentally different in nature. While the former has one or more associated capabilities, the latter is only able to affect how these system capabilities perform. The interactions described in the controller agency matrix are fundamentally different from those found in the system concept A_S and the hetero-functional adjacency matrix A_ρ. Mathemat-

ically speaking, A_Q specifies a relationship between two resources (rather than between processes and resources in the case of A_S, or between capabilities in the case of A_ρ). Conceptually, A_Q specifies control jurisdiction whereas A_S represents allocation of function to form and A_ρ describes the flow of matter and energy between capabilities. These distinctions make hetero-functional graph theory inherently cyber-physical. ■

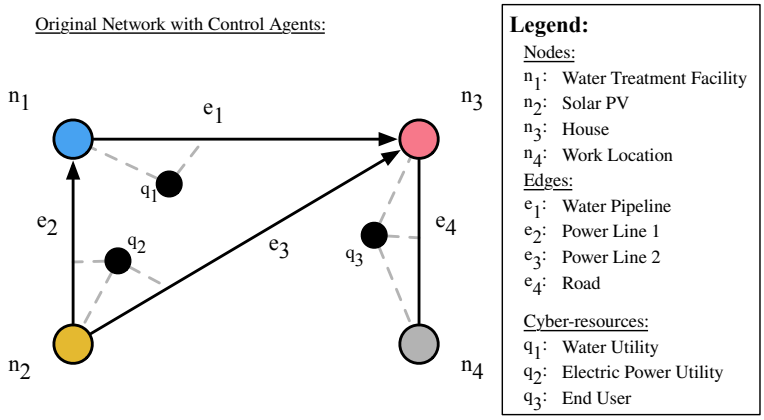


Fig. 3.17: Cyber-Resources in the Example Network: Independent Cyber Resources have jurisdiction over physical resources. Cyber-physical interfaces are indicated with grey dashed edges.

Example 3.6: Building off Example 3.2, the network in Figure 3.1 can be enhanced to include independent cyber-resources as shown in Figure 3.17. The SysML block diagram in Figure 3.2 can be enhanced similarly, as shown in Figure 3.18. The electric power utility, the water utility, and the end users are all considered cyber-resources for their ability to take control actions or make decisions. Associations are then introduced between cyber-resources and their associated (physical) resources to visualize the control structure. SysML uses associations to represent both physical and informatic interfaces whereas hetero-functional graph theory differentiates between the two. All resources without an independent cyber-resource are assumed to have a dependent cyber-resource which has not been shown for modeling simplicity. $Q_D = \emptyset, Q = Q_I$. Similarly, the self-associations reflecting a physical resource's ability to control itself have not been shown to maintain the graphical aesthetic. These associations are used to construct the controller agency matrix in Figure 3.19. The

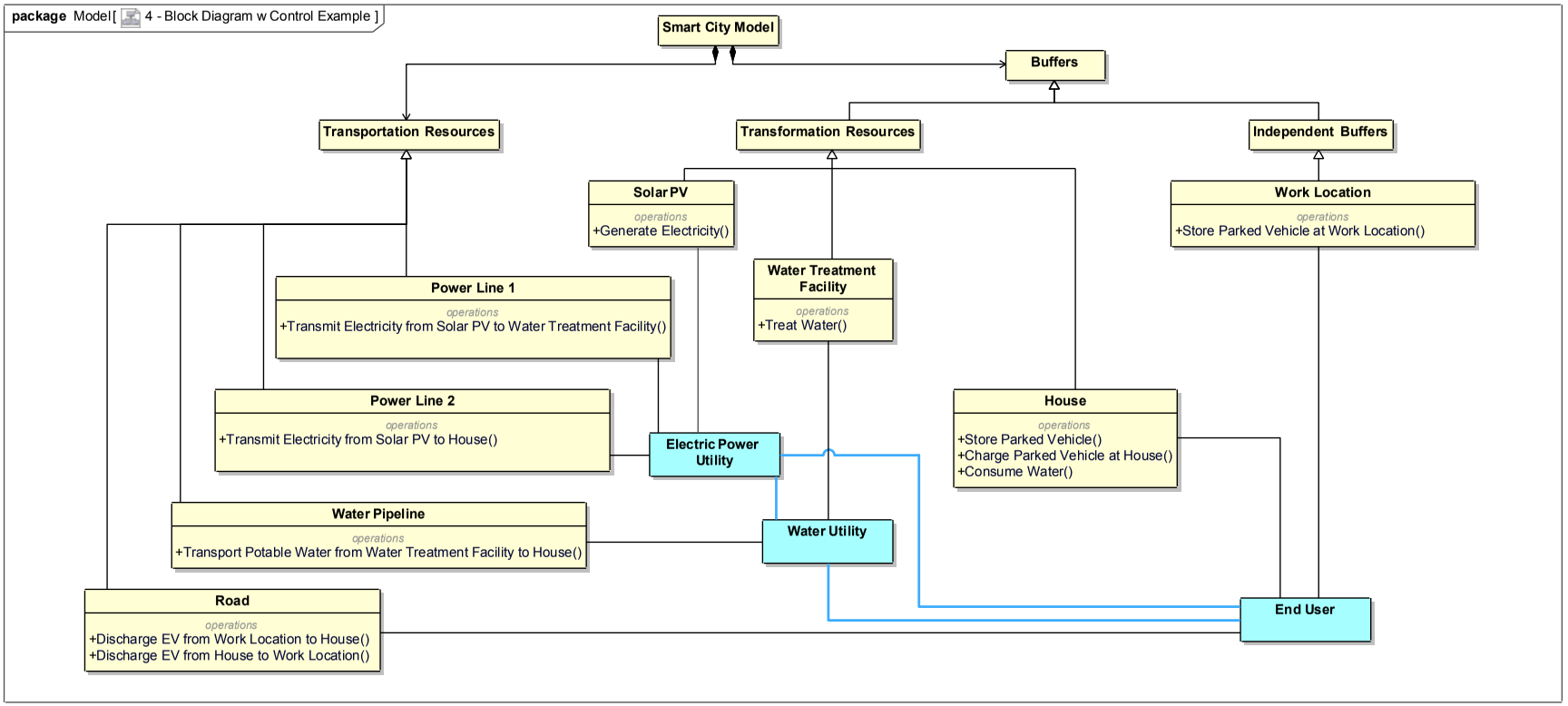


Fig. 3.18: SysML Block Definition Diagram for Example 3.6. This block diagram extends the block diagram from Figure 3.2 to include the Control Agents, who have control authority over resources via (colored) associations.

physical resource's self-associations appear as an identity matrix block. The associations between the physical and independent cyber-resources appear as filled elements in the remaining columns. ■

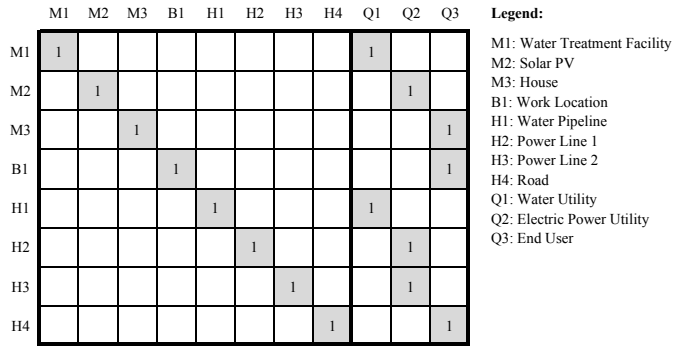


Fig. 3.19: Controller agency matrix for Example 3.6. The block form matrix contains two blocks: (1) The left side: the identity matrix of size $\sigma(R_P) \times \sigma(R_P)$. (2) The right side: the independent controller agency matrix of size $\sigma(R_P) \times \sigma(Q)$.

3.4 Controller Adjacency Matrix

The controller adjacency matrix serves to express the interactions between cyber-resources. Smart city infrastructure systems are controlled by a diverse set of actors, each of whom have jurisdiction over parts of the smart city (e.g. an operand-layer in a multi-operand infrastructure system). Operating the smart city requires these actors to interact.

Again, the controller adjacency matrix is introduced in order to maintain ontological lucidity; recognizing that informatic and physical interactions are fundamentally different. In essence, hetero-functional graph theory identifies three types of interfaces [2, 7]:

1. Type I: physical interactions between physical capabilities as described by the hetero-functional adjacency matrix.

2. Type II: cyber-physical interfaces between a physical resource and all resources cyber-resource as described by the controller agency adjacency matrix.

3. Type III: cyber-interfaces between cyber-resources as described by the controller adjacency matrix.

The three types of interfaces are visually presented in Figure 3.20 [2, 7]. From a reconfiguration perspective, the third type of interaction is often the most complex and requires the most effort to rearrange.

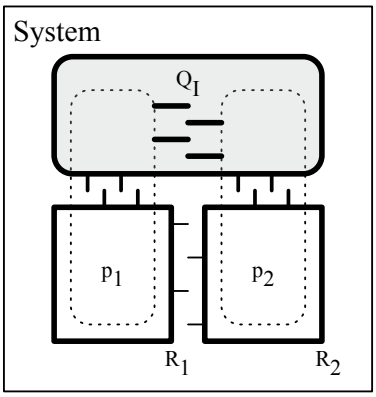


Fig. 3.20: Three Types of Interfaces between Physical and Cyber Resources. Type I is between two physical resources. Type II is between a physical and a cyber-resource. Type III is between two cyber-resources. (Line thickness represents the complexity of interaction and separation.) [2, 7]

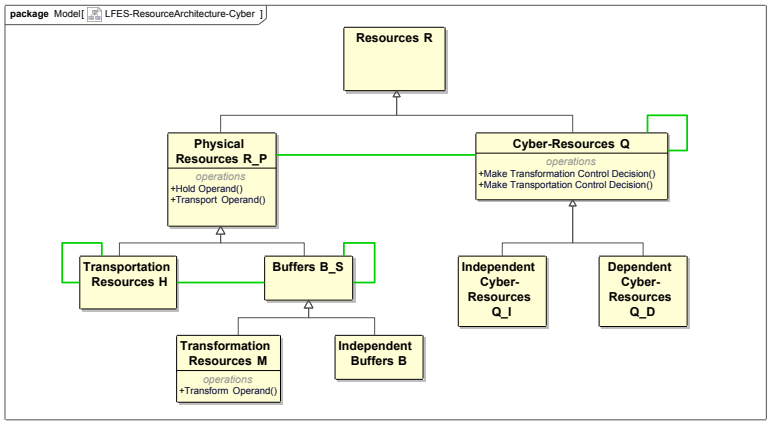


Fig. 3.21: A SysML Block Diagram: The meta-architecture of the system form of a LFES with cyber-resources and their adjacency.

In SysML, all three types of interfaces are represented in the class diagram as shown in Figure 3.21. The actors (or cyber-resources) appear as classes. The associations between

the physical resources represent the Type I interfaces (physical interactions). The Type II interfaces (cyber-physical interfaces) are represented by the association between the physical resources and the cyber-resources. The controller adjacency matrix describes the Type III interactions (cyber-interfaces) and is represented with a self-association link.

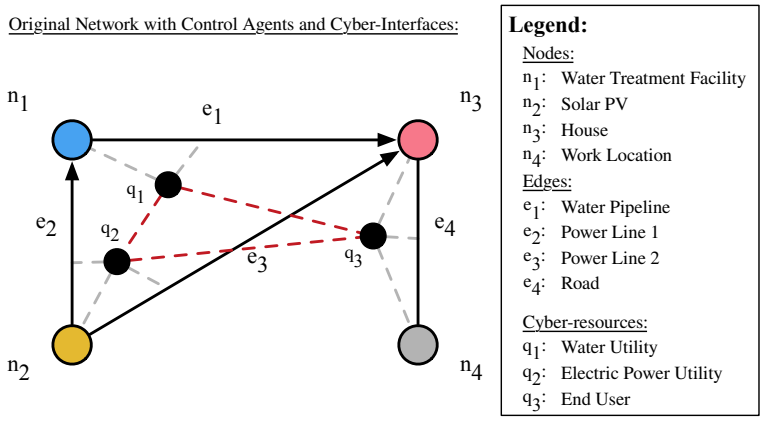


Fig. 3.22: Cyber-Resources in the Example Network: Independent Cyber Resources have jurisdiction over physical resources. Cyber-interfaces indicated with red dashed edges. Cyber-physical interfaces indicated with grey dashed edges.

In hetero-functional graph theory, the associations and interactions between cyber-resources (Type III interactions) are defined by the controller adjacency matrix.

Definition 3.19 – Controller Adjacency Matrix: The controller adjacency matrix A_C is a binary matrix of size $\sigma(Q) \times \sigma(Q)$, whose element $A_C(v_1, v_2)$ is equal to one when the cyber-resource $q_{v_1} \in Q$ pass information to cyber-resource $q_{v_2} \in Q$. ■

The number of connections in the controller adjacency matrix provides insight into the complexity of the network and thus its operational challenges. The cross-layer interactions of physical resources (such as electric vehicle charging stations) often require cyber-resources to coordinate their operational activities as their actions may cause positive outcomes in one layer while causing undesired consequences in another.

Example 3.7: In Example 3.6, both the cyber-resources and the cyber-interfaces were added to the SysML block diagram, with the cyber-interfaces displayed as associations in blue. Figure 3.22 now expands the original 4-node example network to include the cyber-interfaces

as well. From the figures it is clear that all three cyber-resources exchange information. The electric power utility supplies power to both the water treatment facility and the end user. Additionally, the end user consumes water delivered by the water utility. The resulting matrix A_C in Figure 3.23 has size $\sigma(Q) \times \sigma(Q) = 3 \times 3$, with all 9 elements filled. ■

	Q1	Q2	Q3	**Legend:**
Q1	1	1	1	Q1: Water Utility
Q2	1	1	1	Q2: Electric Power Utility
Q3	1	1	1	Q3: End User

Fig. 3.23: Controller Adjacency Matrix for Example 3.7 [2, 7].

3.5 Service as Operand Behavior

The previous subsections focused on a system's capabilities as sentences that describe what an engineering system does. It identified those capabilities (in Section 3.1), connected them into parallel and serial arrangements (in Section 3.2), and described their control structure (in Sections 3.3 and 3.4). Each of these capabilities act on a physical operand that, when delivered across the engineering system boundary, constitutes a service. This section recognizes that as these capabilities act on the operand, they sometimes change its state. Therefore, it is useful in these cases to track these state changes as an operand behavior.

In the SysML modeling language, service delivery is achieved by fulfilling the sequence of processes from input to output in the activity diagram. As soon as the final output is generated, the system service is considered delivered. As mentioned, the state of operands changes as a result of the processes in the activity diagram. In order to describe the behavior of operands in the system, SysML defines a state machine that tracks the state and state transitions of operands or (sub-)systems. The state transitions in the state machine are directly related to the processes in the activity diagram [68, 69]. The state machines for

operands *water*, *electric power*, and *electric vehicle* in the 4-node example network are presented in Figures 3.24, 3.25, and 3.26.

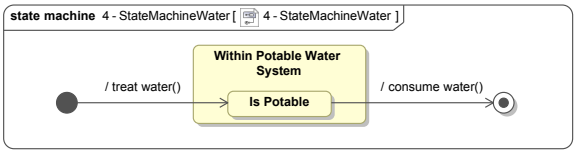


Fig. 3.24: State Machine for the service *deliver water*.

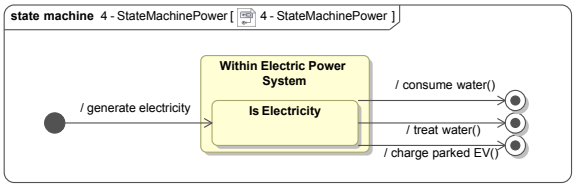


Fig. 3.25: State Machine for the service *deliver electric power*.

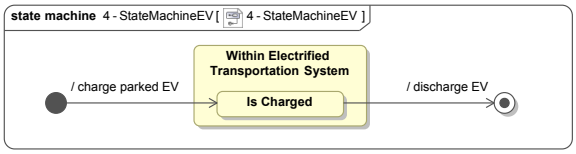


Fig. 3.26: State Machine for the service *deliver EV*.

In order to maintain ontological lucidity, hetero-functional graph theory recognizes that the structure of the operand is distinct from the structure of the engineering system that acts upon it. Furthermore, the state of the engineering system is also distinct from the state of the system's operands. As the operand transitions to its final state, it is considered to have delivered a service. Naturally, the state of the engineering system and its operands are coupled as discussed later in Section 3.6.

3.5.1 Service Delivery as Service Net

An engineering system delivers one or more services; each with its associated operand. The set of services is $L = \{l_1, \ldots, l_{\sigma(L)}\}$, where each service l_i has its associated set of service activities $e_{xl_i} \in \mathcal{E}_{l_i}$ [2, 6, 8].

Definition 3.20 – Service Activity [10, 11, 14, 35, 44]: [17,18] A specific transformation process that may be applied as a part of a larger service. ■

Table 3.4 shows examples of service activities for a broad range of engineering systems.

Table. 3.4: Examples of System Services in LFESs [10, 11, 14]

Transportation:	{Enter passenger at the origin station, Exit the passenger at the destination}
Power Grid:	{Generate electricity at the origin, Consume the electricity at the destination}
Water Distribution:	{Treat water, Consume water}
Production:	{Enter the part to an input buffer, Mill the part, Drill a hole in the part, Polish the part, Exit the part from an output buffer}

Relatively simple services may be described as a sequence of service activities [8]:

$$z_{l_i} = e_{x_1 l_i} e_{x_2 l_i} \dots e_{x_{\sigma(\mathcal{E}_{l_i})} l_i} \tag{3.34}$$

This definition, however, is insufficient for complex services such as those found in healthcare [39, 41, 44, 45] and manufacturing systems [2, 6, 34, 35] where multiple pathways and parallelism is often necessary. Service nets allow service activities to be connected into arbitrary parallel and serial arrangements.

Definition 3.21 – Service Petri Net [10, 11, 14, 35, 44]: [19] Given service l_i, a service net is

[17] In earlier works where hetero-functional graph theory was applied to production systems [2, 6], services represented manufactured products. The transformation processes necessary to evolve the production of a product, were called product events. As the theory found new application domains, however, the more generic term of service was adopted to replace the term products. Furthermore, the term service activity was adopted to align with the SysML language. While the delivery of services and products do have their differences, hetero-functional graph theory assumes that the delivery of a product constitutes a service and hence treats the two concepts equally.

[18] The term “service" within hetero-functional graph theory may be counter-intuitive to some audiences. For example, some might say that services are associated with the execution of the engineering system’s capabilities. Such a view, however, is not consistent with the production system literature. Products and services are delivered only after a ***value-adding*** process has happened. It is not sufficient, for example, to simply hold or move an operand within an engineering system. Consequently, it is necessary to track the intermediate states of a product as it gains value prior to delivery across the system boundary.

[19] In earlier works where hetero-functional graph theory was applied to production systems [2, 6], service nets were called product nets. In more recent work on healthcare delivery systems, service nets are called health nets and represent the health of an individual (patient).

described as a tuple:

$$N_{l_i} = \{S_{l_i}, \mathcal{E}_{l_i}, M_{l_i}, W_{l_i}, Q_{l_i}\} \tag{3.35}$$

where

- N_{l_i} is the service net.
- S_{l_i} is the set of places describing a set of service states.
- $\mathcal{E}_{l_i}$ is the set of transitions describing service activities.
- $\mathbf{M}_{l_i} \subseteq (S_{l_i} \times \mathcal{E}_{l_i}) \cup (\mathcal{E}_{l_i} \times S_{l_i})$ is the set of arcs describing the relations of (service states to service activities) and (service activities to service states). The associated incidence matrix is $M_{l_i} = M^+_{l_i} - M^-_{l_i}$ where the positive incidence matrix has element $M^+_{l_i}(s_{\zeta l_i}, e) \in \{0, 1\}$ and the negative incidence matrix has element $M^-_{l_i}(s_{\zeta l_i}, e) \in \{0, 1\}$ for all $(s_{\zeta l_i}, e) \in S_{l_i} \times \mathcal{E}_{l_i}$.
- $W_{l_i} : \mathbf{M}_{l_i} \rightarrow [0 \dots 1]$ is the set of weights on the arcs describing the service transition probabilities for the arcs.
- Q_{l_i} is the Petri net marking representing the set of service states.

■

The service Petri net structure implies the following discrete-event dynamics:

Definition 3.22 – Timed Petri Net (Discrete-Event Dynamics [249]): Given a binary input firing vector $U^+_{l_i}[k]$ and a binary output firing vector $U^-_{l_i}[k]$ both of size $\sigma(\mathcal{E}_{l_i}) \times 1$, and the positive and negative components $M^+_{l_i}$ and $M^-_{l_i}$ of the Petri net incidence matrix of size $\sigma(S_{l_i}) \times \sigma(\mathcal{E}_{l_i})$, the evolution of the marking vector Q_{l_i} is given by the state transition function $\Phi_T(Q_{l_i}[k], U^-_{l_i}[k], U^+_{l_i}[k])$:

$$Q_{l_i}[k+1] = \Phi_T(Q_{l_i}[k], U^-_{l_i}[k], U^+_{l_i}[k]) \tag{3.36}$$

where $Q_{l_i} = [Q_{Sl_i}; Q_{El_i}]$ and

$$Q_{Sl_i}[k+1] = Q_{Sl_i}[k] + M_{l_i}^{+} U_{l_i}^{+}[k] - M_{l_i}^{-} U_{l_i}^{-}[k] \tag{3.37}$$

$$Q_{El_i}[k+1] = Q_{El_i}[k] - U_{l_i}^{+}[k] + U_{l_i}^{-}[k] \tag{3.38}$$

■

Example 3.8: Example 3.2 introduced a simple infrastructure system in a SysML block diagram and a SysML activity diagram with swim lanes. The system delivers three services: deliver potable water, deliver electric power, and deliver EV. These services were introduced in Figures 3.24, 3.25, and 3.26 as SysML state machine diagrams. Hetero-functional graph theory uses service nets (as defined in Definition 3.21) to describe the service delivery behavior. In order to define the service net, each state of the operands is assigned a place. A transition is assigned to the service activities that evolve the state of the operand. The arcs between the places and transitions are based on the inputs and outputs of each of the transitions. Furthermore, an additional "maintain operand state" transition is connected with two one-way arcs for each of the places in the service net[20].

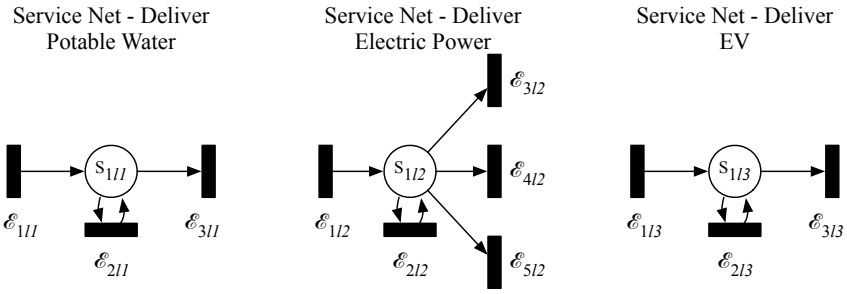


Fig. 3.27: Service Nets: three service nets in the 4-node example network. Operands from left to right: (a) Water, (b) Power, and (c) Electric Vehicle.

The service *deliver water* can, therefore, be described using the service net in Figure 3.27.a. The service net has one place: {Is Potable}, and three transitions: {treat water(),

[20]The addition of a single "maintain operand state" transition for each place is absolutely necessary once holding processes of a transformative nature are added to the model.

maintain potable water(), consume water()}. The first inputs the operand into the system, the second maintains its existence until the third outputs the operand from the system. The non-potable water states are not of concern in the 4-node example network and are considered outside the system boundary.

The service *deliver electricity* is described as a service net in Figure 3.27.b. The service net has one place: {Is Electricity}, and five transitions: {generate electricity(), maintain electric power(), treat water(), consume water(), charge parked EV()}. These five transitions highlight the interdependent nature of a city's infrastructure systems, as the delivery of electric power is critical to delivering other services in the city.

The service *deliver electric vehicle* is described as a service net in Figure 3.27.c. The service net has one place: {Is Charged}, and three transitions: {charge parked EV(), maintain state-of-charge EV(), discharge EV()}. Note that this is a discretization of the continuous state of charge of an electric vehicle[21]. ■

3.5.2 Service Delivery as Service Graph

The service net introduced above can now be translated to a service graph that defines the adjacency of service activities. The translation of a Petri net into a graph is usually achieved by translating the places into nodes, and the transitions into edges. The dual-adjacency matrix of the translated graph calculates the adjacency of the directed edges, and thus the service activities [2, 6]. The service graph is, therefore, the dual-adjacency matrix of the service net. It is calculated as:

$$A_{l_i} = M_{l_i}^{+T} M_{l_i}^{-} \tag{3.39}$$

Where $M_{l_i}^{+}$ is the positive incidence matrix of the service net with size $\sigma(S_{l_i}) \times \sigma(\mathcal{E}_{l_i})$, and $M_{l_i}^{-}$ is the negative incidence matrix of the service net with size $\sigma(S_{l_i}) \times \sigma(\mathcal{E}_{l_i})$. The resulting matrix A_{l_i} has size $\sigma(\mathcal{E}_{l_i}) \times \sigma(\mathcal{E}_{l_i})$, and shows the adjacency of service activities in the service net. This adjacency matrix is then represented as a graph. The transitions are its nodes, and

[21] For a more in depth look into continuous Petri nets, the reader is referred to [250].

the directed arcs represent their adjacency [2, 6].

Example 3.9: In order to convert the service nets into service graphs, the incidence matrices of each of the service nets serve as an input to Equation 3.39. The dual-adjacency matrices are:

$$A_{l_1} = M^{+T}_{\text{water}} M^{-}_{\text{water}} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.40}$$

$$A_{l_2} = M^{+T}_{\text{power}} M^{-}_{\text{power}} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.41}$$

$$A_{l_3} = M^{+T}_{\text{EV}} M^{-}_{\text{EV}} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.42}$$

These adjacency matrices are visualized in Figure 3.28. ■

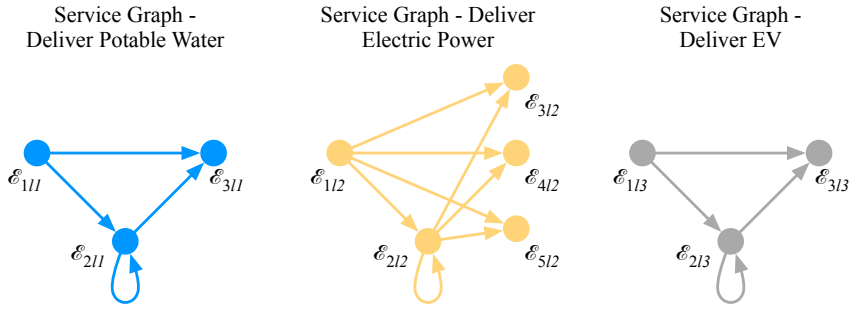


Fig. 3.28: Service Graphs: three service graphs in the 4-node example network. Operands from left to right: (a) Water, (b) Power, and (c) Electric Vehicle.

3.6 Service Feasibility Matrix

The service feasibility matrix couples the structure of the engineering system to the structure of its services. As the state of the engineering system's operands evolve, the state of the engineering system itself must also evolve. Therefore, these two structures are inherently coupled.

In the SysML modeling language, state machines are inherently coupled to activity diagrams. The triggers between states in the state machine correspond to the actions in the activity diagram. The state machines in Figures 3.24, 3.25, and 3.26 can be traced back to the activity diagram in Figure 3.29 (repetition of Figure 3.3). Furthermore, Figure 3.29 shows that the operand states reflected in the state machine diagram are also reflected in the activity diagram as operand or output statements. Additionally, the service model is also integrated in the SysML block diagram as presented in Figure 3.30. The physical resources perform services and are coupled to services in the block diagram by association. Furthermore, services potentially interface with themselves which is represented by self-association [22].

In hetero-functional graph theory, the coupling of the engineering system structure with the operand's behavior as a service is defined by service feasibility matrices [2, 6, 8, 14]. In addition to the coupling of the operand states to the system state, the service feasibility matrix also allows for the calculation of measures of customization and redundancy [2, 15]. This section, first, introduces three feasibility matrices that establish the coupling in Section 3.6.1. It, then, introduces the service selector matrices that allow for the development of service-oriented measures of customization and redundancy in Section 3.6.2.

3.6.1 Service Feasibility Matrix Definitions

By their nature, transformation capabilities, when executed, have the ability to change the operand's state, either in place, or by injecting them across the system boundary. The service transformation feasibility matrix links service transitions to the transformation capabilities

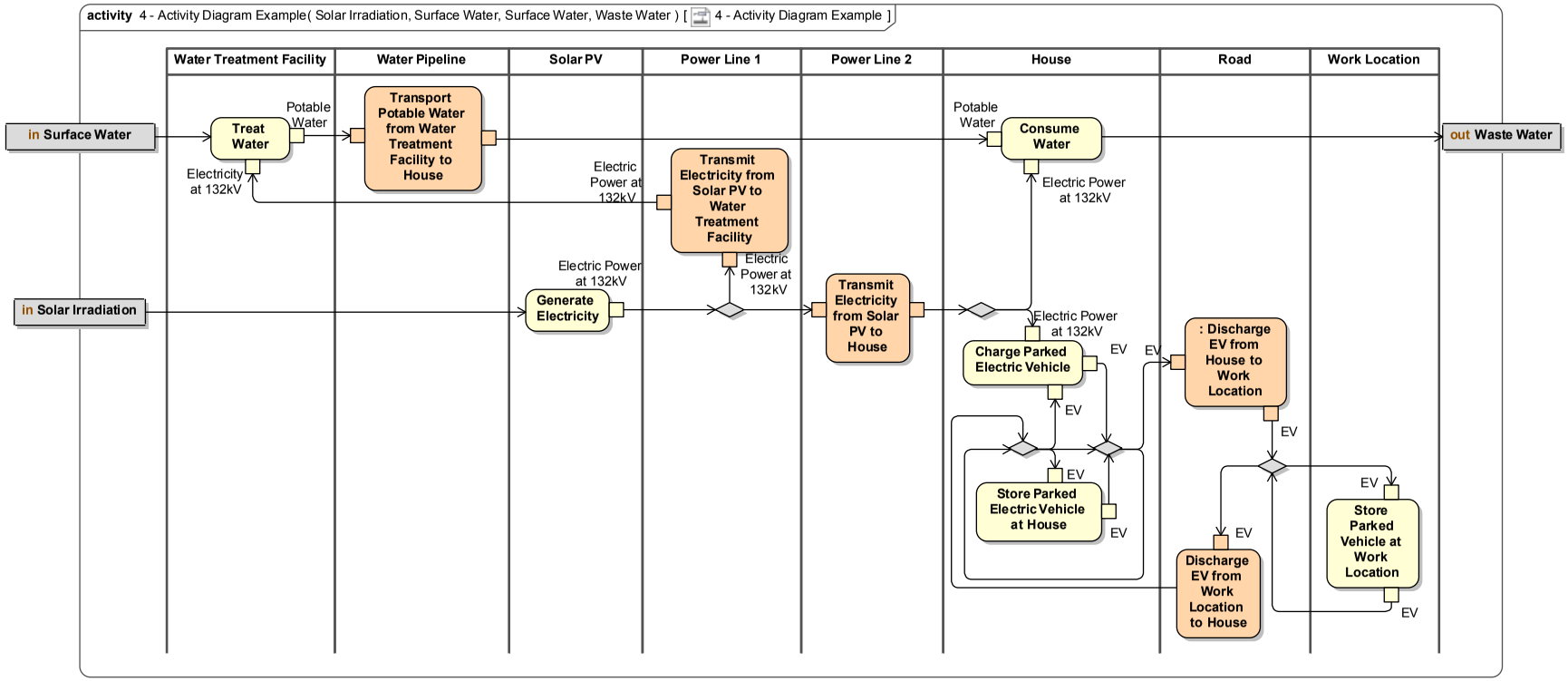


Fig. 3.29: A SysML Activity Diagram: Swim lanes allocate function to form for the 4-node smart city network as presented in Figure 3.1. The network consists of transportation, electricity, and water infrastructure.

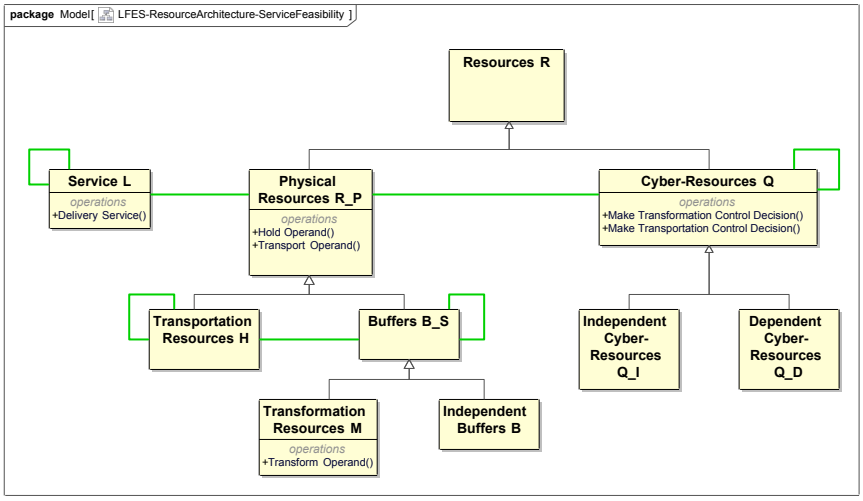


Fig. 3.30: A SysML Block Diagram: The meta-architecture of the system form of a LFES with cyber-resources and the service model.

that can execute them.

Definition 3.23 – Service Transformation Feasibility Matrix [2, 8, 10, 14]: For a given service l_i, a binary matrix of size $\sigma(\mathcal{E}_{l_i}) \times \sigma(P_\mu)$ whose value $\Lambda_{\mu i}(x, j) = 1$ if e_{xl_i} realizes transformation process $p_{\mu j}$. ■

While transportation processes normally do not change the state of their operands, the holding processes of a transformative nature, as introduced in Section 3.1, have the potential to do so. Consequently, the service transportation feasibility matrix couples these holding processes to the service transitions.

Definition 3.24 – Service Transportation Feasibility Matrix [2, 8, 10, 14]: [22,23] For a given service l_i, a binary matrix of size $\sigma(\mathcal{E}_{l_i}) \times \sigma(P_\gamma)$ whose value $\Lambda_{\gamma i}(x, g) = 1$ if e_{xl_i} realizes holding process $p_{\gamma g}$. ■

Note that the service transportation feasibility matrix couples the service transitions to the holding processes, rather than the refined transportation processes.

[22] Originally, $\Lambda_{\gamma i}$ was defined as a binary matrix of size $1 \times \sigma(P_\gamma)$. However, it has since been expanded to address the potentially transformative nature of holding processes.

[23] The introduction of the "maintain operand state()" transition in the service net mentioned on Page 99 now require a link to a single holding process of a non-transformative nature so as to indicate that a holding process is associated with the operand that it holds.

In order for each service activity to be realized by the physical engineering system, there must exist exactly one associated transformation or holding process [2, 6].

$$\sum_{j}^{\sigma(P_\mu)} \Lambda_{\mu i}(x, j) + \sum_{g}^{\sigma(P_\gamma)} \Lambda_{\gamma i}(x, g) = 1 \tag{3.43}$$

Having zero would mean that the service activity would not occur. Having two would mean that the set of transformation and holding processes are not mutually exclusive and an absence of ontological laconicity.

Together, the service transformation and transportation feasibility matrices can be integrated to provide a system-wide sense of the coupling between system processes and service transitions.

Definition 3.25 – Service Feasibility Matrix: For a given service l_i, a binary matrix of size $\sigma(\mathcal{E}_{l_i}) \times \sigma(P)$ whose value $\Lambda_i(x, w) = 1$ if e_{xl_i} realizes process p_w.

$$\Lambda_i = \left[\Lambda_{\mu i} \quad | \quad \Lambda_{\gamma i} \otimes \mathbb{1}^{\sigma(P_\eta)T} \right] \tag{3.44}$$

■

Example 3.10: This example couples the service behavior from Example 3.8 to the structural model from Example 3.2. To that end, it uses the service feasibility matrices. First, the *service transformation feasibility matrix* couples the system transformation processes P_μ to the service activities $\mathcal{E}_{l_i}$. Recall that the set of transformation processes consists of: {treat water, generate electricity, consume water}.

Deliver Potable Water: The set of service activities for the operand water $\mathcal{E}_{l_1}$ contains: {treat water, maintain potable water, consume water}. The service transformation feasibility matrix for the operand water $\Lambda_{\mu l_1}$, therefore, has a size of $\sigma(\mathcal{E}_{l_1}) \times \sigma(P_\mu) = 3 \times 3$, with two filled

elements:

$$\Lambda_{\mu l_1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.45}$$

Deliver Electric Power: The set of service activities for the operand electricity $\mathcal{E}_{l_2}$ contains: {generate electricity, maintain electric power, treat water, consume water, charge parked EV}. The service transformation feasibility matrix for the operand water $\Lambda_{\mu l_2}$, therefore, has a size of $\sigma(\mathcal{E}_{l_2}) \times \sigma(P_\mu) = 5 \times 3$, with three filled elements:

$$\Lambda_{\mu l_2} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.46}$$

Deliver Electric Vehicle: The set of service activities for the operand EV $\mathcal{E}_{l_3}$ contains: {charge parked EV, maintain state-of-charge EV, discharge EV}. The service transformation feasibility matrix for the operand EV $\Lambda_{\mu l_3}$, therefore, has size $\sigma(\mathcal{E}_{l_3}) \times \sigma(P_\mu) = 3 \times 3$, with zero filled elements. None of the activities are realized by transformation processes.

$$\Lambda_{\mu l_3} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.47}$$

Second, the *service transportation feasibility matrix* couples the system holding processes P_γ to the service activities $\mathcal{E}_{l_i}$. Recall that the set of holding processes consists of: {carry potable water, carry electricity, charge electric vehicle, discharge electric vehicle, carry electric vehicle}.

Deliver Potable Water: The service transportation feasibility matrix for the operand water

$\Lambda_{\gamma l_1}$ has size $\sigma(\mathcal{E}_{l_1}) \times \sigma(P_\gamma) = 3 \times 5$. The matrix contains one filled element that maps the holding process *carry water* to the service activity *maintain potable water*:

$$\Lambda_{\gamma l_1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.48}$$

Deliver Electricity: The service transportation feasibility matrix for the operand electricity $\Lambda_{\gamma l_2}$ has size $\sigma(\mathcal{E}_{l_2}) \times \sigma(P_\gamma) = 5 \times 5$. The matrix contains two filled elements. The first couples the holding process *charge parked EV* to its service activity. The second couples the service activity *maintain electric power* to the holding process *carry electricity*:

$$\Lambda_{\gamma l_2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{3.49}$$

Deliver Electric Vehicle: The service transportation feasibility matrix for the operand EV $\Lambda_{\gamma l_3}$ has size $\sigma(\mathcal{E}_{l_3}) \times \sigma(P_\gamma) = 3 \times 5$. The matrix contains three filled elements. The first two elements couple the holding processes *charge parked EV* and *discharge EV* to the service activities. The third element couples *park EV* to the service activity *maintain state-of-charge EV*:

$$\Lambda_{\gamma l_3} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.50}$$

Finally, the *service feasibility matrices* are calculated based on the two previous matrices, using Equation 3.44:

- The service feasibility matrix for the service deliver potable water is calculated as

follows:

$$\Lambda_1 = \left[\Lambda_{\mu 1} \quad | \quad \Lambda_{\gamma 1} \otimes \mathbb{1}^{\sigma(P_\eta)T} \right] \tag{3.51}$$

where Λ_1 has size $\sigma(\mathcal{E}_{l_1}) \times \sigma(P) = 3 \times 83$, with 18 filled elements.

- The service feasibility matrix for the service deliver electricity is calculated as follows:

$$\Lambda_2 = \left[\Lambda_{\mu 2} \quad | \quad \Lambda_{\gamma 2} \otimes \mathbb{1}^{\sigma(P_\eta)T} \right] \tag{3.52}$$

where Λ_2 has size $\sigma(\mathcal{E}_{l_2}) \times \sigma(P) = 5 \times 83$, with 35 filled elements.

- The service feasibility matrix for the service deliver electric vehicle is calculated as follows:

$$\Lambda_3 = \left[\Lambda_{\mu 3} \quad | \quad \Lambda_{\gamma 3} \otimes \mathbb{1}^{\sigma(P_\eta)T} \right] \tag{3.53}$$

where Λ_3 has size $\sigma(\mathcal{E}_{l_3}) \times \sigma(P) = 3 \times 83$, with 48 filled elements.

■

3.6.2 Service Degrees of Freedom

The service feasibility matrices described above are now used to calculate how many system capabilities apply to a service activity, to a service as a whole, or the full line (or set) of services. When enumerated these are called service degrees of freedom. Table 3.5 provides service selector matrices that are essential to the calculation of the service degrees of freedom.

Definition 3.26 – Service Transformation Degrees of Freedom [6,8,14]: The set of independent service events $\mathcal{E}_{LM}$ that completely define the available combinations of transformation process and resource that are required by the delivery of the service. ■

Definition 3.27 – Service Transportation Degrees of Freedom [6,8,14]: The set of independent service events $\mathcal{E}_{LH}$ that completely define the available combinations of transporta-

Table. 3.5: Types of Service Selector Matrices [6, 8, 14]

	Symbol	Formula	Scope
Transformation	Λ_{Mxi}	$\left[e_x^T \Lambda_{\mu i}\right]^T \mathbb{1}^{\sigma(M)T}$	Service Activity
	Λ_{Mi}	$\left[\mathbb{1}^{\sigma(\mathcal{E}_{l_i})T} \odot \Lambda_{\mu i}\right]^T \mathbb{1}^{\sigma(M)T}$	Service
	Λ_{ML}	$\bigvee_i^{\sigma(L)} \Lambda_{Mi}$	Service Line
Transportation	Λ_{Hxi}	$\left[\left[e_x^T \Lambda_{\gamma i}\right] \otimes \mathbb{1}^{\sigma(P_\eta)T}\right]^T \mathbb{1}^{\sigma(R)T}$	Service
	Λ_{Hi}	$\left[\mathbb{1}^{\sigma(\mathcal{E}_{l_i})T} \odot \Lambda_{\gamma i} \otimes \mathbb{1}^{\sigma(P_\eta)T}\right]^T \mathbb{1}^{\sigma(R)T}$	Service
	Λ_{HL}	$\bigvee_i^{\sigma(L)} \Lambda_{Hi}$	Service Line
Transformation & Transportation	Λ_{Sxi}	$\left[e_x^T \Lambda_i\right]^T \mathbb{1}^{\sigma(R)T}$	Service Activity
	Λ_{SMxi}	$\left[\begin{array}{c\|c} \Lambda_{Mxi} & \mathbf{1} \\ \hline \multicolumn{2}{c}{\mathbf{0}} \end{array}\right]$	Service Activity
	Λ_{SHxi}	$\left[\begin{array}{c\|c} \mathbf{0} & \mathbf{0} \\ \hline \multicolumn{2}{c}{\Lambda_{Hxi}} \end{array}\right]$	Service Activity
	Λ_{Si}	$\left[\begin{array}{c\|c} \Lambda_{Mi} & \mathbf{1} \\ \hline \multicolumn{2}{c}{\Lambda_{Hi}} \end{array}\right] = \left[\mathbb{1}^{\sigma(\mathcal{E}_{l_i})T} \odot \Lambda_i\right]^T \mathbb{1}^{\sigma(R)T}$	Service
	Λ_{SHi}	$\left[\begin{array}{c\|c} \mathbf{0} & \mathbf{0} \\ \hline \multicolumn{2}{c}{\Lambda_{Hi}} \end{array}\right]$	Service
	Λ_{SL}	$\left[\begin{array}{c\|c} \Lambda_{ML} & \mathbf{1} \\ \hline \multicolumn{2}{c}{\Lambda_{HL}} \end{array}\right] = \bigvee_i^{\sigma(L)} \Lambda_{Si}$	Service Line

tion process and resource that can be utilized by the delivery of the service. ■

Definition 3.28 – Service Degrees of Freedom [6, 8, 14]: The set of independent service

events $\mathcal{E}_{LS}$ that completely define the available combinations of transformation or transportation process and resource that can be utilized by the delivery of the service. ■

In order to complete the calculation, a number of service selector matrices are introduced as an intermediate step. They are all of equal size to the corresponding knowledge base. Table 3.5 summarizes their calculation. For example, Equations 3.54 through 3.56 calculate the number of transformation and transportation capabilities utilized by the full set of services delivered by the engineering system [6, 8, 14].

$$DOF_{LM} = \langle \Lambda_{ML} \cdot J_M, \bar{K}_M \rangle_F \tag{3.54}$$

$$DOF_{LH} = \langle \Lambda_{HL} \cdot J_{\bar{H}}, \bar{K}_{\bar{H}} \rangle_F \tag{3.55}$$

$$DOF_{LS} = \langle \Lambda_{SL} \cdot J_S, \bar{K}_S \rangle_F \tag{3.56}$$

These values are important to understand how capable the engineering system is relative to the services it needs to deliver [2, 15]. Excessively redundant capabilities can be potentially decommissioned to reduce costs without any degradation in the quality of service.

Example 3.11: This example demonstrates the calculation of all three service degree of freedom measures for the 4-node example network (Example 3.2). It builds off the service feasibility matrices calculated in Example 3.10.

First, the service transformation degrees of freedom DOF_{LM} are calculated using Equation 3.54. The transformation knowledge base J_M was calculated in Example 3.4, and Λ_{ML} can be calculated using the service selector matrices in Table 3.5.

$$\Lambda_{ML} = \bigvee_{i}^{\sigma(L)} \Lambda_{Mi} \tag{3.57}$$

where:

$$\Lambda_{Mi} = \left[\mathbb{1}^{\sigma(\mathcal{E}_{l_i})T} \odot \Lambda_{\mu i}\right]^T \mathbb{1}^{\sigma(M)T} \tag{3.58}$$

$$\Lambda_{M1} = \left[\mathbb{1}^{\sigma(\mathcal{E}_{l_1})T} \odot \Lambda_{\mu 1}\right]^T \mathbb{1}^{\sigma(M)T} = \left[\begin{bmatrix}1 & 1 & 1\end{bmatrix} \odot \begin{bmatrix}1 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 1\end{bmatrix}\right]^T \begin{bmatrix}1 & 1 & 1\end{bmatrix} = \begin{bmatrix}1 & 1 & 1\\ 0 & 0 & 0\\ 1 & 1 & 1\end{bmatrix} \tag{3.59}$$

$$\Lambda_{M2} = \left[\mathbb{1}^{\sigma(\mathcal{E}_{l_2})T} \odot \Lambda_{\mu 2}\right]^T \mathbb{1}^{\sigma(M)T} = \left[\begin{bmatrix}1 & 1 & 1 & 1 & 1\end{bmatrix} \odot \begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 0\\ 1 & 0 & 0\\ 0 & 0 & 1\\ 0 & 0 & 0\end{bmatrix}\right]^T \begin{bmatrix}1 & 1 & 1\end{bmatrix} = \begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \tag{3.60}$$

$$\Lambda_{M3} = \left[\mathbb{1}^{\sigma(\mathcal{E}_{l_3})T} \odot \Lambda_{\mu 3}\right]^T \mathbb{1}^{\sigma(M)T} = \left[\begin{bmatrix}1 & 1 & 1\end{bmatrix} \odot \begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\right]^T \begin{bmatrix}1 & 1 & 1\end{bmatrix} = \begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix} \tag{3.61}$$

Therefore:

$$\Lambda_{ML} = \begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \tag{3.62}$$

Now, the service transformation degrees of freedom are calculated based on J_S, Λ_{ML}, and the transformation constraints matrix K_M (all-zeroes, 3×3):

$$DOF_{LM} = \langle \Lambda_{ML} \cdot J_M, \bar{K}_M \rangle_F = \langle \begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \cdot \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}, \begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \rangle_F = 3 \tag{3.63}$$

In conclusion, the number of service transformation degrees of freedom is three.

The number of service transportation degrees of freedom is calculated with Equation 3.55. Matrix Λ_{HL} has size 80×8 with all filled elements. Consequently, the number of service transportation degrees of freedom is eight.

Given the calculations for the service transformation and transportation degrees of freedom, the service degrees of freedom follow straightforwardly:

$$DOF_{LS} = \langle \Lambda_{SL} \cdot J_S, \bar{K}_S \rangle_F \tag{3.64}$$

where

$$\Lambda_{SL} = \left[\begin{array}{c|c} \Lambda_{ML} & \mathbf{1} \\ \hline & \Lambda_{HL} \end{array}\right] \tag{3.65}$$

The total number of service degrees of freedom is, therefore, 11. ■

3.7 The System Adjacency Matrix: An Integrated View of Hetero-functional Graph Theory

The previous sections introduced the fundamental mathematical models in hetero-functional graph theory. This section integrates these models into a single hetero-functional graph represented as a System Adjacency Matrix $\mathbb{A}$. The System Adjacency Matrix provides a holistic representation of the cyber-physical engineering system. As shown in Equation 3.66, the matrix is organized in a matrix block form derived from the six mathematical models discussed previously.

$$\mathbb{A} = \begin{bmatrix} \mathbb{A}_L & \mathbb{A}_{L\rho} & \mathbf{0} \\ \mathbb{A}_{\rho L} & A_\rho & \mathbb{A}_{\rho C} \\ \mathbf{0} & \mathbb{A}_{C\rho} & A_C \end{bmatrix} \tag{3.66}$$

Alternatively, a projected system adjacency matrix can be calculated so as to significantly eliminate sparsity.

$$\widetilde{\mathbb{A}} = \begin{bmatrix} \mathbb{A}_L & \widetilde{\mathbb{A}}_{L\rho} & \mathbf{0} \\ \widetilde{\mathbb{A}}_{\rho L} & \widetilde{A}_\rho & \widetilde{\mathbb{A}}_{\rho C} \\ \mathbf{0} & \widetilde{\mathbb{A}}_{C\rho} & A_C \end{bmatrix} \tag{3.67}$$

This section discusses the unprojected and projected forms of each of these block matrix forms in detail.

In essence, a hetero-functional graph represents an interconnected model of an engineering system including its capabilities, its service model, and its control model. Equation 3.66 shows that the central matrix block of the system adjacency matrix $\mathbb{A}$ is the hetero-functional adjacency matrix A_ρ (discussed in Section 3.2 on Page 79). It represents the logical connections between the system's capabilities. Equation 3.66 also shows that the bottom right matrix block of $\mathbb{A}$ is controller adjacency matrix, A_C, (discussed in Section 3.4 on Page 92). It represents the informatic connections between the controller agents in the engineering system. The upper left matrix block $\mathbb{A}_L$ represents the *collection* of service models in the engineering system (discussed in Section 3.5 on Page 95). It includes the connections between the service activities in each of the service models (Petri nets). Consequently, the block matrices $\mathbb{A}_{L\rho}$ and $\mathbb{A}_{\rho L}$ represent the logical coupling of service activities to the engineering system's capabilities (i.e. structural degrees of freedom). They are derived from the service feasibility matrices (discussed in Section 3.6 on Page 102). Similarly, the block matrices $\mathbb{A}_{\rho C}$ and $\mathbb{A}_{C\rho}$ represent the logical coupling of controller agents to the engineering system's capabilities. They are derived from the controller agency matrix (discussed in Section 3.3 on Page 86). Each of these block matrices are now discussed in the order presented above.

Block A_ρ: The hetero-functional adjacency matrix A_ρ is the core of the System Adjacency Matrix $\mathbb{A}$. A_ρ constitutes the first two models of hetero-functional graph theory, as System Concept A_S is necessary for its calculation. It represents the structure of the physical engineering system in terms of structural degrees of freedom as nodes and system-sequence degrees of freedom as edges. It has a size of $\sigma(R_P)\sigma(P) \times \sigma(R_P)\sigma(P)$. The projected hetero-functional adjacency matrix $\widetilde{A}_\rho$ is a projection of A_ρ, as calculated in Equation 3.31. Its size is $\sigma(\mathcal{E}_S) \times \sigma(\mathcal{E}_S)$.

Block A_C: The controller adjacency matrix A_C is the bottom right block of $\mathbb{A}$. It

constitutes the third model of hetero-functional graph theory. It represents the interfaces between the cyber-resources. It has a size of $\sigma(Q) \times \sigma(Q)$.

Block $\mathbb{A}_L$: $\mathbb{A}_L$ is the upper left matrix block of $\mathbb{A}$. As it represents the collection of service models in the engineering system, it constitutes the fourth model of hetero-functional graph theory. It takes service activities as nodes and shows the logical coupling between them. The block diagonal features terms of the form $\mathcal{M}_{li}^{+T}\mathcal{M}_{li}^{-}$, as demonstrated in Equation 3.68. Recall that the postive and negative components of the service Petri net incidence matrix for a given service $\mathcal{M}_{li}^{+}$ and $\mathcal{M}_{li}^{-}$ create a bipartite graph between service activities and places. The product $\mathcal{M}_{li}^{+T}\mathcal{M}_{li}^{-}$ creates an adjacency matrix between the service activities. The block diagonal form shows that activities in one service are only coupled within the same service. Distinct services are entirely uncoupled.

$$\mathbb{A}_L = \begin{bmatrix} \mathcal{M}_{l1}^{+T}\mathcal{M}_{l1}^{-} & 0 & \dots & 0 \\ 0 & \mathcal{M}_{l2}^{+T}\mathcal{M}_{l2}^{-} & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & \mathcal{M}_{\sigma(L)}^{+T}\mathcal{M}_{\sigma(L)}^{-} \end{bmatrix} \tag{3.68}$$

where $\mathbb{A}_L$ has size $\sum_i \sigma(\mathcal{E}_{l_i}) \times \sum_i \sigma(\mathcal{E}_{l_i})$.

Block $\mathbb{A}_{\rho C}$ and $\mathbb{A}_{C\rho}$: These two block matrices couple the structural degrees of freedom in A_ρ and the independent cyber-resources Q. In this regard, the controller agency matrix as the third model of hetero-functional graph theory proves useful. Rather than using A_Q which couples the set of resources R to the set of physical resources R_P, it is more useful to use the sub-matrix $\overline{A}_Q$ which only shows the jurisdiction of independent cyber-resources Q

over the physical resources R_P. The calculation of $\mathbb{A}_{\rho C}$ then follows straightforwardly:

$$\mathbb{A}_{\rho C} = \overline{A}_Q \otimes \mathbb{1}^{\sigma(P)} \tag{3.69}$$

where the term $\mathbb{1}^{\sigma(P)}$ is introduced to create a coupling to all the processes that pertain to the physical resources R_P. The couplings captured in $\mathbb{A}_{\rho C}$ represent the *sensed* information coming from the structural degrees of freedom to the cyber-resources, while the couplings captured in $\mathbb{A}_{C\rho}$ represent the *actuation* signals sent in the opposite direction. Consequently, the size of $\mathbb{A}_{\rho C}$ is $\sigma(R_P)\sigma(P) \times \sigma(Q)$ and the size of $\mathbb{A}_{C\rho}$ is its transpose. Their projected form is calculated as follows:

$$\widetilde{\mathbb{A}}_{\rho C} = \mathbb{P}_S \mathbb{A}_{\rho C} \tag{3.70}$$

$$\widetilde{\mathbb{A}}_{C\rho} = \mathbb{A}_{C\rho} \mathbb{P}_S^T \tag{3.71}$$

Blocks $\mathbb{A}_{L\rho}$ and $\mathbb{A}_{\rho L}$: These two block matrices couple the transitions in the service Petri nets $\mathcal{E}_{l_i}$ to the structural degrees of freedom in A_ρ. In this regard, the service feasibility matrices as the sixth model of hetero-functional graph theory proves useful. The service feasibility matrix $\mathbb{A}_{L\rho}$ is calculated as:

$$\mathbb{A}_{L\rho} = \mathbb{A}_{\rho L}^T = \begin{bmatrix} \widehat{\Lambda}_{l_1} \\ \widehat{\Lambda}_{l_2} \\ \vdots \\ \widehat{\Lambda}_{\sigma(L)} \end{bmatrix} \tag{3.72}$$

where

$$\widehat{\Lambda}_{l_i} = \mathbb{1}^{\sigma(R_P)T} \otimes \Lambda_{l_i} \tag{3.73}$$

which has a size of $\sigma(\mathcal{E}_{l_i}) \times \sigma(R_P)\sigma(P)$. The size of matrix $\mathbb{A}_{L\rho}$ is consequently $\sum_i \sigma(\mathcal{E}_{l_i}) \times$

$\sigma(R_P)\sigma(P)$. Their projected form is calculated as follows:

$$\widetilde{\mathbb{A}}_{\rho L} = \mathbb{P}_S \mathbb{A}_{\rho L} \tag{3.74}$$

$$\widetilde{\mathbb{A}}_{L\rho} = \mathbb{A}_{L\rho} \mathbb{P}_S^T \tag{3.75}$$

Example 3.12: This example integrates all examples in the chapter related to the 4-node example network to create the system adjacency matrix. Given the size of the matrices, even for a small 4-node network, the example constructs the projected system adjacency matrix. As a conclusion, the projected system adjacency matrix is visualized in Figure 3.31.

The projected system adjacency matrix consists of seven matrix blocks.

- The *first block*, the projected hetero-functional adjacency matrix $\widetilde{A}_\rho$ is directly drawn from Example 3.5:

$$\widetilde{A}_\rho = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.76}$$

- The *second block*, the controller adjacency matrix A_C is directly drawn from Example 3.7:

$$A_C = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{3.77}$$

- The *third block*, matrix $\mathbb{A}_L$ is constructed using the three service adjacency matrices from Example 3.9:

$$\mathbb{A}_L = \begin{bmatrix} A_{l_1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & A_{l_2} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & A_{l_3} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} & & \mathbf{0} & & \mathbf{0} \\ & \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} & & \mathbf{0} & \\ \mathbf{0} & & \mathbf{0} & & \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \end{bmatrix} \tag{3.78}$$

- The *fourth and fifth* blocks are matrices $\widetilde{\mathbb{A}}_{\rho C}$ and $\widetilde{\mathbb{A}}_{C\rho}$. These are calculated using the controller agency matrix ($\overline{A}_Q$) from Example 3.6:

$$\widetilde{\mathbb{A}}_{\rho C} = \mathbb{P}_S \left[\overline{A}_Q \otimes \mathbb{1}^{\sigma(P_\eta)} \right] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.79}$$

- The *sixth and seventh* blocks are matrices $\widetilde{\Lambda}_{L\rho}$ and $\widetilde{\Lambda}_{\rho L}$. These are calculated as follows:

$$\widetilde{\mathbb{A}}_{L\rho} = \widetilde{\mathbb{A}}_{\rho L}^{T} = \begin{bmatrix} \mathbb{1}^{\sigma(R)T} \otimes \Lambda_{l_1} \\ \mathbb{1}^{\sigma(R)T} \otimes \Lambda_{l_2} \\ \mathbb{1}^{\sigma(R)T} \otimes \Lambda_{l_3} \end{bmatrix} \mathbb{P}_{S}^{T} = \left[\begin{array}{cccccccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right] \tag{3.80}$$

After the derivation of each of the seven block matrices in the system adjacency matrix, the full system adjacency matrix is calculated using Equation 3.67:

$$\mathbb{A} = \left[\begin{array}{cc} \left[\begin{array}{ccccccccccc} 0 & 1 & 1 & & & & & & & & \\ 0 & 1 & 1 & & & & \mathbf{0} & & & & \mathbf{0} \\ 0 & 0 & 0 & & & & & & & & \\ & & & 0 & 1 & 1 & 1 & 1 & & & \\ & & & 0 & 1 & 1 & 1 & 1 & & & \\ & \mathbf{0} & & 0 & 0 & 0 & 0 & 0 & & & \mathbf{0} \\ & & & 0 & 0 & 0 & 0 & 0 & & & \\ & & & 0 & 0 & 0 & 0 & 0 & & & \\ & & & & & & & & 0 & 1 & 1 \\ & \mathbf{0} & & & & & \mathbf{0} & & 0 & 1 & 1 \\ & & & & & & & & 0 & 0 & 0 \end{array}\right] & \left[\begin{array}{ccccccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right] \\ \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] & \left[\begin{array}{ccccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}\right] & \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{array}\right] \\ & \left[\begin{array}{ccccccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{array}\right] & \left[\begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array}\right] \end{array}\right] \tag{3.81}$$

Where the final size of the system adjacency matrix is:

$$\left[\sum_{i}^{\sigma(L)} \mathcal{E}_{l_i} + \sigma(R)\sigma(P) + \sigma(Q)\right] \times \left[\sum_{i}^{\sigma(L)} \mathcal{E}_{l_i} + \sigma(R)\sigma(P) + \sigma(Q)\right] = 25 \times 25 \tag{3.82}$$

Finally, the system adjacency matrix is visualized in Figure 3.31. In this figure, a distinction is made between three separate parts of the system adjacency matrix to maintain readability. The original 4-node example network is presented in Figure 3.31.(a). The core of the system adjacency matrix is the hetero-functional adjacency matrix, presented in Figure 3.31.(b). Figure 3.31.(c) presents the visualization of matrices $\widetilde{\mathbb{A}}_{\rho C}$, $\widetilde{\mathbb{A}}_{C\rho}$, and A_C, mapped to the system capabilities. Finally, Figure 3.31.(d) visualizes the service delivery matrices $\widetilde{\mathbb{A}}_{L\rho}$, $\widetilde{\mathbb{A}}_{\rho L}$, and $\mathbb{A}_L$, mapped to the system capabilities. The system capabilities are shared across all three figures and are the interface when the models are superimposed. ■

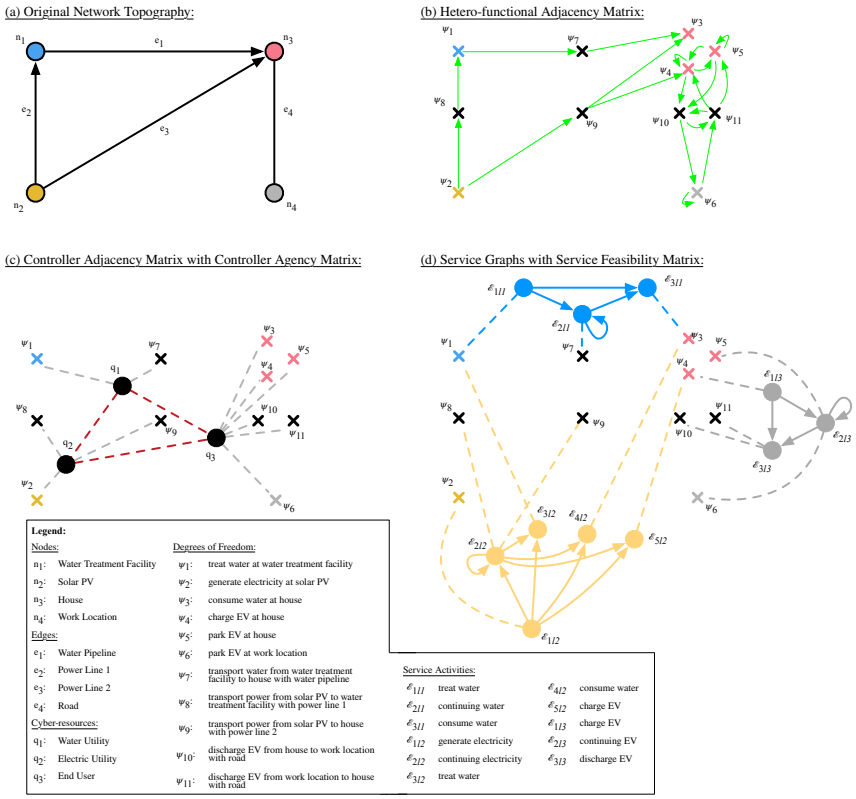


Fig. 3.31: System Adjacency Matrix: A comparison of the original 4-node example network in (a) with the hetero-functional adjacency matrix in (b), the controller model coupled to the capabilities in (c), and the service model coupled to the capabilities in (d). Graphs (b), (c), and (d) are three distinct representations of subsets of the system adjacency matrix.

The System Adjacency Matrix provides a holistic representation of the structure of a cyber-physical engineering system. It, explicitly, addresses both system function and

system form. At its heart, the hetero-functional adjacency matrix A_ρ represents the edges between structural degrees of freedom (i.e. capabilities) as nodes. Similarly, the nodes of the controller adjacency matrix A_C are the controllers Q; each of which is only able to execute a single control or decision algorithm. Finally, the nodes of the $\mathbb{A}_L$ matrix represents the service Petri transitions of each service. In all of these cases, system function and system form are inextricably tied. In this regard, hetero-functional graph theory offers greater ontological lucidity than a graph theory based exclusively on form or function.

3.8 Conclusion

In conclusion, relative to other quantitative structural models, hetero-functional graph theory differentiates itself in that it centers itself on the allocated architecture as it addresses system function and system form simultaneously. This implicitly requires mutually exclusive and collectively exhaustive sets of system processes and resources. It also quantifies a system's capabilities in terms of structural degrees of freedom. The hetero-functional adjacency matrix then couples these capabilities into a network that respects both functional interactions and physical interfaces. ***Because there is an explicit differentiation of system processes, hetero-functional graph theory provides a basis upon which networks with unlike function can be combined into a single mathematical model of system structure.*** Hetero-functional graph theory is also explicitly cyber-physical in that it differentiates between systems of equivalent physical capability but different control structure. The controller adjacency matrix admits a spectrum of control architectures from entirely centralized to entirely distributed. Finally, hetero-functional graph theory recognizes that the execution of the engineering system's capabilities can change the state of the associated operands. The service Petri net describes this state evolution and the service feasibility matrices couples it back to the capabilities of the engineering system. These six matrices describe entirely different phenomena in an engineering system. Their systematic integration results in the

system adjacency matrix as a block diagonal form that addresses the full structure of a cyber-physical engineering system. At the beginning of this chapter, Table 3.1 provided an overview of its discussion. Table 3.6 now provides an overview of the elements and the mathematical notation of hetero-functional graph theory.

Given the articulation of these seven graph theoretical models in hetero-functional graph theory, the discussion can return to the motivational example in Chapter 2.2.2. Each of these seven matrices has been calculated and worked in Section 4.11. Whereas “multi-layer networks" all exhibited a constraint that prevented a complete mathematical description of this motivational example, hetero-functional graph theory straightforwardly provides a mathematical model. Furthermore, this result shows that hetero-functional graph theory does not suffer from any of the previously identified modeling constraints. Rather than continue discussion on this relatively small example, the next chapter, (Chapter 4), demonstrates hetero-functional graph theory on a hypothetical smart city infrastructure system with features that more closely resemble the complexity of real-life city infrastructure.

Chapter Summary:

This chapter presented hetero-functional graph theory as the first internally consistent quantitative structural modeling approach to describing engineering systems. It leverages seven mathematical models: 1. the System Concept, 2. the Hetero-functional Adjacency Matrix, 3. The Controller Agency Matrix, 4. the Controller Adjacency Matrix, 5. the Service as Operand Behavior, 6. the Service Feasibility Matrix, and 7. the System Adjacency Matrix. This chapter has demonstrated the use of hetero-functional graph theory for a small example for illustration purposes. The next chapter, Chapter 4, demonstrates the use of hetero-functional graph theory for two much larger and more complex test cases. ■

Table. 3.6: Summary of Hetero-functional Graph Theory

Conceptual Elements	Mathematical Elements	Mathematical Notation
(A) System Concept	System Form: Physical Resources	$R_P = B_S \cup H,\ B_S = M \cup H$
	System Function: Processes	$P = P_\mu \cup \left(P_\gamma \bigtimes P_\eta\right)$
	System Knowledge Base	$J_S = \left[\begin{array}{c\|c} J_M & \mathbf{0} \\ \leftarrow \quad J_{\vec{H}} & \rightarrow \end{array}\right]$, where $J_{\vec{H}} = \left[J_\gamma \otimes \mathbb{1}^{\sigma(P_\eta)}\right] \cdot \left[\mathbb{1}^{\sigma(P_\gamma)} \otimes J_H\right]$
	System Constraints Matrix	$K_S = \left[\begin{array}{c\|c} K_M & \mathbf{0} \\ \leftarrow \quad K_{\vec{H}} & \rightarrow \end{array}\right]$
	System Degrees of Freedom	$DOF_S = \sum_{v}^{\sigma(R)\sigma(P)} \sum_{w}^{\sigma(R)\sigma(P)} \left[J_S \ominus K_S\right](w,v) = \sum_{v}^{\sigma(R)\sigma(P)} \sum_{w}^{\sigma(R)\sigma(P)} A_S(w,v)$
(B) Hetero-functional Adjacency Matrix	System Sequence Knowledge Base	$J_\rho = A_S^V A_S^{VT} = \left[J_S \cdot \tilde{K}_S\right]^V \left[J_S \cdot \tilde{K}_S\right]^{VT}$
	System Sequence Constraints Matrix	K_ρ. Please refer to Table 3.3.
	Hetero-functional Adjacency Matrix	$A_\rho = J_\rho \ominus K_\rho$, or when projecting: $\tilde{A}_\rho = \mathbb{P}_S A_\rho \mathbb{P}_S^T$
	System Sequence Degrees of Freedom	$DOF_\rho = \sum_{\chi 1}^{\sigma(R)\sigma(P)\sigma(R)\sigma(P)} \sum_{\chi 2} \left[A_\rho\right](\chi_1,\chi_2) = \sum_{\psi_1}^{\sigma(\mathcal{E}_S)\sigma(\mathcal{E}_S)} \sum_{\psi_2} \left[\tilde{A}_\rho\right](\psi_1,\psi_2)$
(C) Controller Agency Matrix	Physical Resources	$R_P = B_S \cup H,\ B_S = M \cup H$
	(Cyber-) Resources	$R = R_P \cup Q$, where cyber-resources: $Q = Q_I \cup Q_D$
	Controller Agency Matrix	A_Q
(D) Controller Adjacency Matrix	Controller Adjacency Matrix	A_C
(E) Service as Operand Behavior	Services	$L = \{l_1, \ldots, l_{\sigma(L)}\}$
	Service Activities	$e_{xl_i} \in \mathcal{E}_{l_i}$
	Service String and Service Petri net	$z_{l_i} = e_{x_1 l_i} e_{x_2 l_i} \cdots e_{x_{\sigma(\mathcal{E}_{l_i})} l_i}$, and $N_{l_i} = \{S_{l_i}, \mathcal{E}_{l_i}, M_{l_i}, W_{l_i}, Q_{l_i}\}$
(F) Service Feasibility Matrix	Service Transformation Feasibility Matrix	$\Lambda_{ML} = \bigvee_{i}^{\sigma(L)} \left[\left[\mathbb{1}^{\sigma(\mathcal{E}_{l_i})T} \odot \Lambda_{\mu i}\right]^T \mathbb{1}^{\sigma(M)T}\right]$
	Service Transportation Feasibility Matrix	$\Lambda_{HL} = \bigvee_{i}^{\sigma(L)} \left[\left[\mathbb{1}^{\sigma(\mathcal{E}_{l_i})T} \odot \Lambda_{\gamma i} \otimes \mathbb{1}^{\sigma(P_\eta)T}\right]^T \mathbb{1}^{\sigma(R)T}\right]$
	Service Line Feasibility Matrix	$\Lambda_{SL} = \left[\begin{array}{c\|c} \Lambda_{ML} & \mathbf{1} \\ \Lambda_{HL} & \end{array}\right] = \bigvee_{i}^{\sigma(L)} \Lambda_{Si}$
	Service Degrees of Freedom	$DOF_{LM} = \langle \Lambda_{ML} \cdot J_M, \tilde{K}_M \rangle_F$ $DOF_{LH} = \langle \Lambda_{HL} \cdot J_{\vec{H}}, \tilde{K}_{\vec{H}} \rangle_F$ $DOF_{LS} = \langle \Lambda_{SL} \cdot J_S, \tilde{K}_S \rangle_F$

Chapter 4

Modeling Interdependent Smart City Infrastructures with Hetero-functional Graph Theory

Chapter Abstract:

This chapter serves to demonstrate the development of a hetero-functional graph theory structural model of an engineering system. The chapter also demonstrates that hetero-functional graph theory does not impose the ontological and modeling limitations introduced by the multilayer networks literature. The chapter provides two examples, the first is an interdependent smart city infrastructure test case entitled *"Trimetrica"*. The second models the example network from Section 2.2.2.

The structural model of the *"Trimetrica"* test case is presented in Sections 4.1 through 4.10. This work is directly adopted from Chapter 5, called *"Modeling Interdependent Smart City Infrastructure Systems with HFGT"*, in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5]. One feature of *"Trimetrica"* is its significant heterogeneity of function. It integrates a water distribution system, a power grid, and an (electrified) transportation network. The modeling approach is incremental:

each of the models in hetero-functional graph theory are discussed and connected to the other models to construct the system adjacency matrix piece-by-piece. Sections 4.3 through 4.8 develop the six mathematical models of hetero-functional graph theory leading up to the definition of the system adjacency matrix in Section 4.9. Additionally, Section 4.10 shows that hetero-functional graph theory overcomes the ontological and modeling constraints found in the multilayer networks literature.

The second demonstration is presented in Section 4.11 and is directly adopted from Appendix A, called *"Representing a Four Layer Network in Hetero-functional Graph Theory"*, in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5]. This demonstration further reinforces that hetero-functional graph theory overcomes the ontological and modeling constraints found in the multilayer network literature. This demonstration explicitly shows that hetero-functional graph theory can describe engineering systems that cannot be described with multilayer networks.

This dissertation develops a Hetero-functional Graph Theory for structural modeling of engineering systems. The previous chapter provided an exposition of hetero-functional graph theory in terms of its constituent mathematical models. This chapter now applies the theory to an interdependent smart city infrastructure test case called *"Trimetrica"*, which serves as an instance of an engineering system. First, the role of test cases in smart city development is discussed in Section 4.1. Thereafter, the Trimetrica test case is introduced in Section 4.2. Then Sections 4.3 through 4.9 subsequently discuss the seven elements of hetero-functional graph theory. Section 4.10 provides a discussion of the results, referring back to the premise of Section 2.2.2 and it specifically establishes that hetero-functional graph theory does not impose the same ontological and modeling constraints as the multilayer networks literature. Finally, Section 4.11 explicitly develops a hetero-functional graph theory model of the example network presented in Section 2.2.2, which could *not* be modeled by the multilayer

networks approaches.

4.1 The Role of Test Cases in Smart City Development

Test cases have served an invaluable role in the development of infrastructure systems. While data collected from real-life cities is invaluable for drawing conclusions about certain smart city instances, it is often unclear how modeling, analysis, and simulation methods used in one case study can be applied to other cities. Any one smart city may not be representative of the class of smart cities. Indeed, the process of data collection itself comes with inherent ontological assumptions that are often overlooked. Reconsider the discussion from Chapter 2.3. Figure 2.7 (on page 45) shows that relying on data-driven from a single smart city instance may violate the ontological properties of completeness and lucidity. Stated differently, the smart city data used in the model may not address all smart city abstractions (i.e. violation of completeness). For example, the footnote from Example 2.2 (on page 52) mentions that power flow analysis data fails to capture the data from the lead lines between generators and substations. The smart city data used in the model may also be applied to multiple smart city abstractions (i.e. violation of lucidity). For example, the footnote from Example 2.2 (on page 52) mentions that power flow analysis data uses the same data to refer to both power generation and energy storage facilities. Test cases resolve many[1] of these concerns because their meta-data implies an underlying reference architecture which may be ontologically critiqued for generality.

Many infrastructure engineering fields have recognized these concerns from a practical perspective. The power systems engineering field developed strong rationales for the usage of test cases as early as the 1970s [251] and has since developed several test case repositories [252, 253]. Water distribution engineers have used the famous “Anytown" network [254]. In

[1] While reference architectures overcome doubts about how representative a certain instantiated architecture may be, their meta-data may still violate the four ontological properties. The examples provided above indicate the degeneracy of power flow analysis data despite its widespread use.

both cases, these test cases allowed the development of rigorous numerical methods without compromising the physical security of sharing data about critical infrastructure [255–258]. Despite a general trend towards transportation system instances, some transportation system engineers have advocated the use of test cases to understand the fundamental properties of transportation dynamics [259]. Many of these developments arguments were presented in the development of “Symmetrica" as a test case for the development transportation-electrification research [50]. It is on this foundation that “Trimetrica" is developed as an interdependent smart city infrastructure.

4.2 Smart City Test Case: Trimetrica

Trimetrica consists of three infrastructure systems: a power system, a transportation system, and a water distribution system. The first two are drawn from the “Symmetrica" test case [50]. The third is derived from the “Anytown" water network [254]. These systems are chosen as three critical infrastructures within any smart city. They also include both physical as well as cyber-resources. Additionally, their dynamic behaviors are well-known. Therefore, this test case may, in the future, provide a basis for dynamic cyber-physical modeling of smart cities. The topological lay-out is presented in Figure 4.1 and the associated data is publicly available [47]. This section first discusses each of the individual infrastructure systems. Secondly, it details the connection between the infrastructure systems, and thirdly the section highlights the “architectural" assumptions in this test case.

The *electric power system* is based on the IEEE 201-bus test case [218, 219] and is depicted in Figure 4.1.b. The power system consists of a single power generation bus, 200 load buses, and 200 branches. The power system supplies electric power to the consumption nodes, which in many cases are connected to the other infrastructure systems. If a power node is connected to another system, it is assumed that it maintains a separate power consumption function. The power system provides the service “delivery of electric power

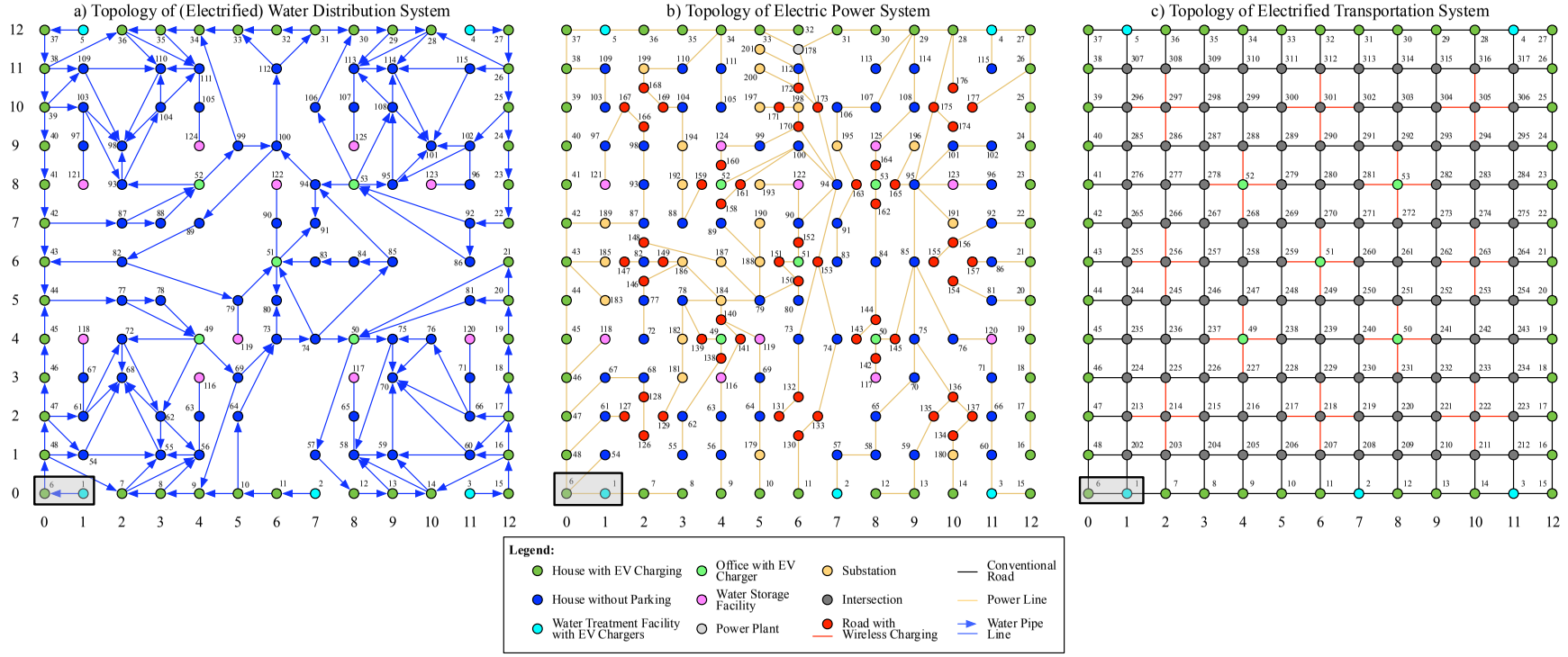


Fig. 4.1: Topological depiction of the Trimetrica Smart City Infrastructure Test Case: The networks are topologically superimposed.

at 132kV", which upon arrival may be used directly or to support the transformation and storage of other processes at the consumption node.

The *transportation system* in the Trimetrica test case consists of 169 nodes, structured in a grid-like shape, as depicted in Figure 4.1.c [50]. From the 169 nodes, 53 nodes are charging stations that charge electric vehicles while they are parked. These charging locations impose a load on the power system (and are thus a cross-layer connection). The nodes are connected by 624 roads, 104 of which are charging roads that charge electric vehicles as they drive over the road. The charging roads also impose a load on the power system. Note that each of the edges in Figure 4.1.c represents two unidirectional roads. The transportation system aims to provide the service “transport person from an origin to a destination", where the person uses a vehicle, and the vehicle remains a part of the transportation system at all times.

The *water distribution system* is developed based on the University of Exeter’s Anytown water distribution network test case [254]. The Trimetrica Water Distribution System was first introduced in [260], but has since been revised to better match the electric power system and transportation system architecture. The Trimetrica water distribution system scales the Anytown network by a factor 5, arriving at 5 water treatment facilities, 10 water storage facilities, and 110 water consumption nodes. These consumption nodes provide both warm and cold water to the consumers. The structure of the Anytown network has not been maintained identically, but has been revised to fit the structure of the Symmetrica test case. Consequently, the Trimetrica water distribution system consists of 186 water pipe lines, of which 19 are bidirectional and 167 are unidirectional[2]. The Trimetrica water distribution system is presented in Figure 4.1.a. The specific service of the water system is delivering potable water to the consumers in the city.

In a smart city, these three infrastructure system types share resources, rather than being physically separated. This coupling is mimicked in Trimetrica. For example, a Water

[2]The water pipes are assumed to be loss-less, without any elevation differences. The water system pressure is assumed to be maintained by the water treatment facilities and the water storage facilities, rather than the use of pumps in the pipe lines.

Treatment Facility appears in the water distribution system as a “water generating resource". However, from a power system perspective, the water treatment facility is a load bus, as it consumes power to treat the water. Additionally, the water treatment facility contains a parking lot with an EV charging facility for its employees. From a transportation system perspective, the water treatment facility is a buffering resource, as (electric) vehicles can be stored in the parking lot. This example shows that a single resource is shared across all three infrastructure systems. Figure, 4.1, therefore uses color to indicate the resources in Trimetrica.

The resources shared across infrastructure systems naturally have the same coordinates in each of the infrastructure topologies. However, it is important to note that two resources with the same coordinates are not necessarily the same resource. For example, the Water Storage Facility at (4,9) (Node 124), is the same resource in the electric power system (Node 124). However, the intersection in the transportation system, Node 288 at (4,9), is separate from the water storage facility and consequently not the same resource.

Some resources, however, have a different representation in the infrastructure topologies. The transportation system uses edges to represent roads between intersections. Each of these edges represents two separate roads in opposite directions. However, when these roads facilitate wireless charging of the vehicles, they share a single load bus in the electric power system. When the systems are discussed independently, they contain three separate resources. However, as a result of explicitly sharing resources across infrastructure systems, the three resources aggregate into a single resource. This resource is presented as an edge in the transportation system, and as a node in the electric power system. An example is Node 126 in the electric power system, which is also the edge from Node 203 to Node 214, which in turn represents both unidirectional roads between Nodes 203 and 214. Table 4.1 provides an overview of all system resources that appear in Trimetrica. The table also identifies the infrastructure system(s) in which the resources appear.

For reasons of clarity, the cyber-resources are not shown in Figure 4.1. Table 4.1 also

Table. 4.1: Resources in Trimetrica with associated infrastructure system and controller type.

#	Resources	Infrastructure Type			Controller Type			Count
		Transportation System	Electric Power System	Water Distribution System	End User	Water Utility	Power Utility	
1	Water treatment facility w/ EV charger	■	■	■		■		5
2	House w/ EV charger	■	■	■	■			43
3	Office w/ EV charger	■	■	■	■			5
4	House w/o parking		■	■	■			62
5	Water storage facility		■	■		■		10
6	Electrified road	■	■		■			52
7	Power plant		■				■	1
8	Substation		■				■	23
9	Intersection	■			■			116
10	Conventional road	■			■			520
11	Electric power line		■				■	200
12	Directed water pipeline			■		■		167
13	Undirected water pipeline			■		■		19

provides an overview of the cyber-resources, and their relationship to Trimetrica's resources. Note that the Electric Power System and the Water Distribution System generation and transportation resources are controlled centrally. In contrast, the transportation system is controlled solely by the end users.

The Trimetrica smart city infrastructure test case is designed to provide a platform to compare modeling methods of smart city infrastructure systems. As discussed before in Chapter 2.3, these infrastructure systems are Large Flexible Engineering Systems both separately and when combined. Hetero-functional Graph Theory can thus be used to model the Trimetrica test case, as demonstrated in the remainder of this Chapter.

Given its size, visualizing the Trimetrica test case is often challenging. Therefore, the book uses both full and partial visualizations to support the mathematics. The topological lay-out of the full model is depicted in Figure 4.1. The partial model is independently depicted in Figure 4.2.

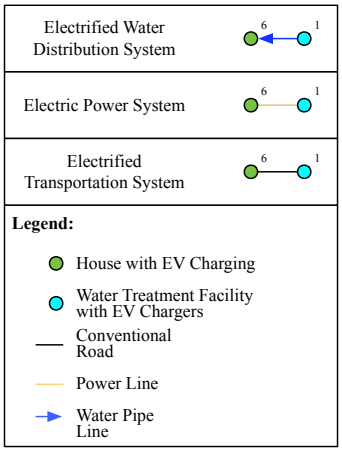


Fig. 4.2: Partial topological depiction of the Trimetrica Smart City Infrastructure Test Case: The networks are topologically superimposed.

After this introduction of the Trimetrica test case, the chapter now continues by modeling Trimetrica using Hetero-functional Graph Theory. The chapter discusses all seven elements of Hetero-functional Graph Theory sequentially to derive a Hetero-functional System Adjacency Matrix as a complete, mathematical model of Trimetrica.

4.3 System Concept

System concept is introduced in Section 3.1 as the mapping of system function onto system form. This section discusses system concept for the Trimetrica test case. Trimetrica consists of three interfacing infrastructure systems, as introduced in Section 4.2. It is now critical to define system concept accurately across the disciplinary systems to derive the correct System Adjacency Matrix in the end. The section first discusses the resources, followed by the processes. Finally, this section maps the processes to the resources to calculate the smart city knowledge base.

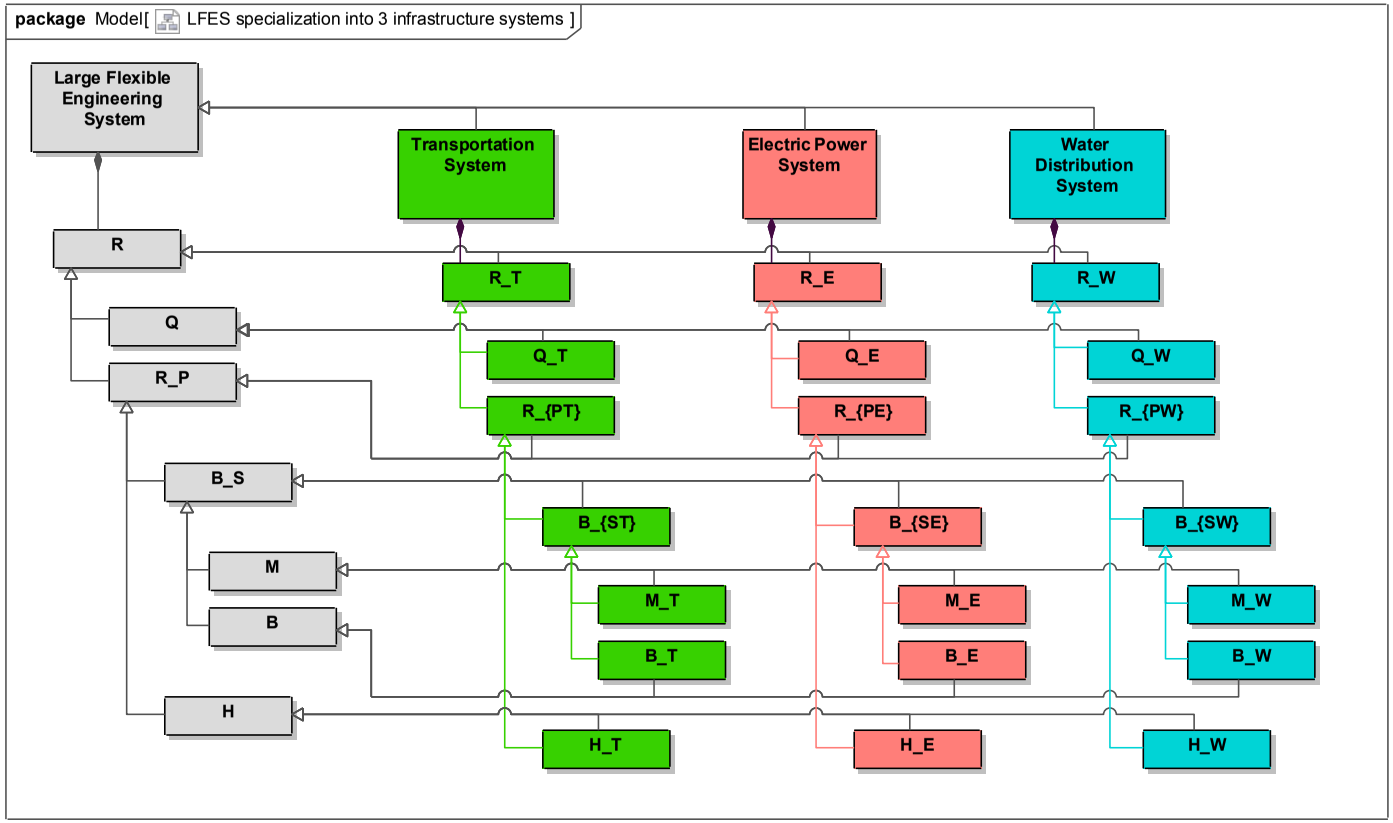


Fig. 4.3: SysML specialization of three infrastructure systems relative to the LFES meta-architecture.

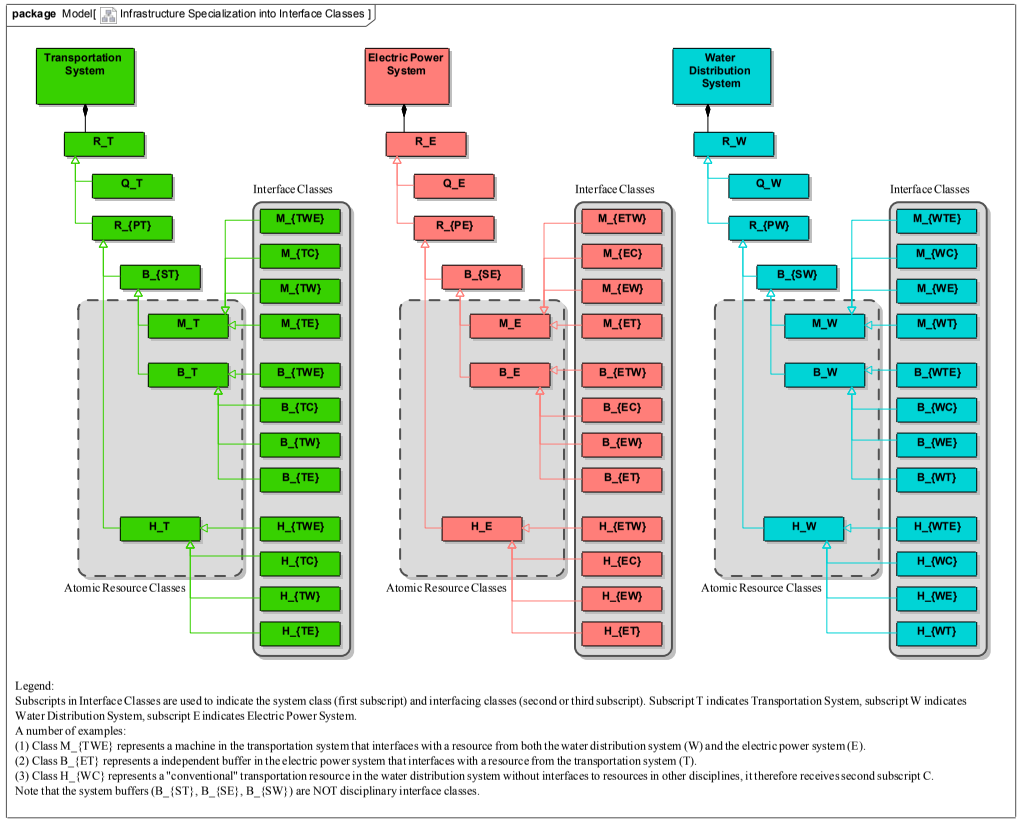


Fig. 4.4: Full SysML overview of the disciplinary system class structure with specialization of atomic disciplinary classes into the complete set of interface classes.

4.3.1 Smart City Resources

The three smart city infrastructures both independently and collectively adhere to the definition of a Large Flexible Engineering System (LFES). Therefore, to derive the smart city resources, this section relies on the resource architecture of Large Flexible Engineering Systems as discussed in Section 3.1, in Figure 3.4 on Page 70.

In the Trimetrica Case Study, the LFES resource structure is specialized into three types: a transportation, an electric power, and a water distribution resource structure. By virtue of this specialization, the resource structure of the three infrastructure systems equals the structure of the LFES. This is best explained using Figure 4.3, in which the SysML representation of the LFES resource meta-architecture is depicted aside the specialized resource architectures of the transportation system, electric power system, and the water distribution system. Each LFES consists of a set of resources R. Similar to the specialization of the LFES meta-architecture, the set of resources R is specialized into transportation system resources R_T, electric power system resources R_E, and water distribution system resources R_W. Mathematically, $R = R_T \cup R_E \cup R_W$.

Furthermore, Figure 4.3 shows the atomic class structure of the set of LFES resources: $R = M \cup B \cup H \cup Q$. Similarly, the atomic class structure of the infrastructure systems is:

- Transportation System: $R_T = M_T \cup B_T \cup H_T \cup Q_T$
- Electric Power System: $R_E = M_E \cup B_E \cup H_E \cup Q_E$
- Water Distribution System: $R_W = M_W \cup B_W \cup H_W \cup Q_W$

Note that these resource classes are also specializations of the LFES resources:

- Transformation Resources $M = M_T \cup M_E \cup M_W$
- Independent Buffers $B = B_T \cup B_E \cup B_W$
- Transportation Resources $H = H_T \cup H_E \cup H_W$

- Cyber Resources $Q = Q_T \cup Q_E \cup Q_W$

The set of atomic resource classes in Figure 4.3 specializes each of the infrastructure systems individually. This specialization does not specify if resources interface or interact across infrastructure systems. Therefore, *interface classes* are introduced as a specialization that distinguishes resource classes based on their interfaces or interactions with other infrastructure types. Figure 4.4 shows such a specialization. The atomic resource classes are specialized into four groups of interface classes.

- The first group is *conventional*, indicated by subscript c. The conventional resources do not interact across disciplinary boundaries. For example, H_{TC} is a conventional transportation resource in the transportation system.

- The second and third groups are resources that interact with one *other* infrastructure system. For example, M_{WE} is a water distribution system transformation resource that interfaces with the electric power system.

- The fourth group contains resources that interact with two *other* infrastructure systems. For example, M_{ETW} is an electric power system resource that interfaces with both the transportation and the water distribution system.

Note that each of the subscripts indicates the infrastructure interface type. The first subscript indicates its original infrastructure type, and the second and third subscript are arbitrarily sequenced.

The interface classes in Figure 4.4 are still defined within a specified infrastructure system. However, some resources are part of multiple infrastructure systems. Therefore, the interface classes must be specialized into multi-operand resource classes. These multi-operand resource classes inherit their properties from multiple interface classes in distinct infrastructure systems. The multi-operand resource classes are thus truly part of multiple infrastructure systems. Figure 4.5 presents an overview of the multi-operand resource classes

that appear in Trimetrica[3].

It is now important to note that if hetero-functional graph theory is applied to another test case or system, these specializations will be different. For example, if Trimetrica did not have three infrastructure systems but four infrastructure systems, the number of interface classes would be much larger. Each of the atomic resource classes is currently specialized into four possible combinations[4]. With four infrastructure systems the number of possible combinations becomes eight. Similarly, the number of potential multi-operand resource classes as combinations of interface classes will increase rapidly by virtue of an added layer.

However, Trimetrica uses only a subset of the available interface classes and multi-operand resource classes. Trimetrica contains 13 resource types, as specified in Table 4.1. These resource types are now matched to the interface and multi-operand resource classes:

1. “Water treatment facility with EV charger" – is part of all three infrastructure systems. From a transportation system perspective, this resource is a buffer as it stores an EV temporarily. From a water distribution system perspective, this resource is a transformation resource, as it adds water across the system boundary. From an electric power system perspective, this resource is a transformation resource, as it is a load on the electric power grid. As a result, it is classified as resource class: $(M_E\&M_W\&B_T)$. In Trimetrica there are 5 instances.

2. “House with EV charger" – is part of all three infrastructure systems. From a transportation system perspective, this resource is a buffer as it stores an EV temporarily. From a water distribution system perspective, this resource is a transformation resource, as it takes water across the system boundary. From an electric power system perspective, this resource is a transformation resource, as it is a load on the elec-

[3]The name of multi-operand resource classes uses the following convention: its name is a combination of each of its parent interface classes. The sequence of the combination is: first the transformation resource, then the buffer, and last the transportation resource. For example, $M_E\&B_W$ is a multi-operand resource that inherits from the interface class M_{EW} in the electric power system and B_{WE} in the water distribution system.

[4]Figure 4.4 presents 36 possible resource types. In all three systems, the transformation resources, independent buffers, and transportation resources are specialized as four interface classes.

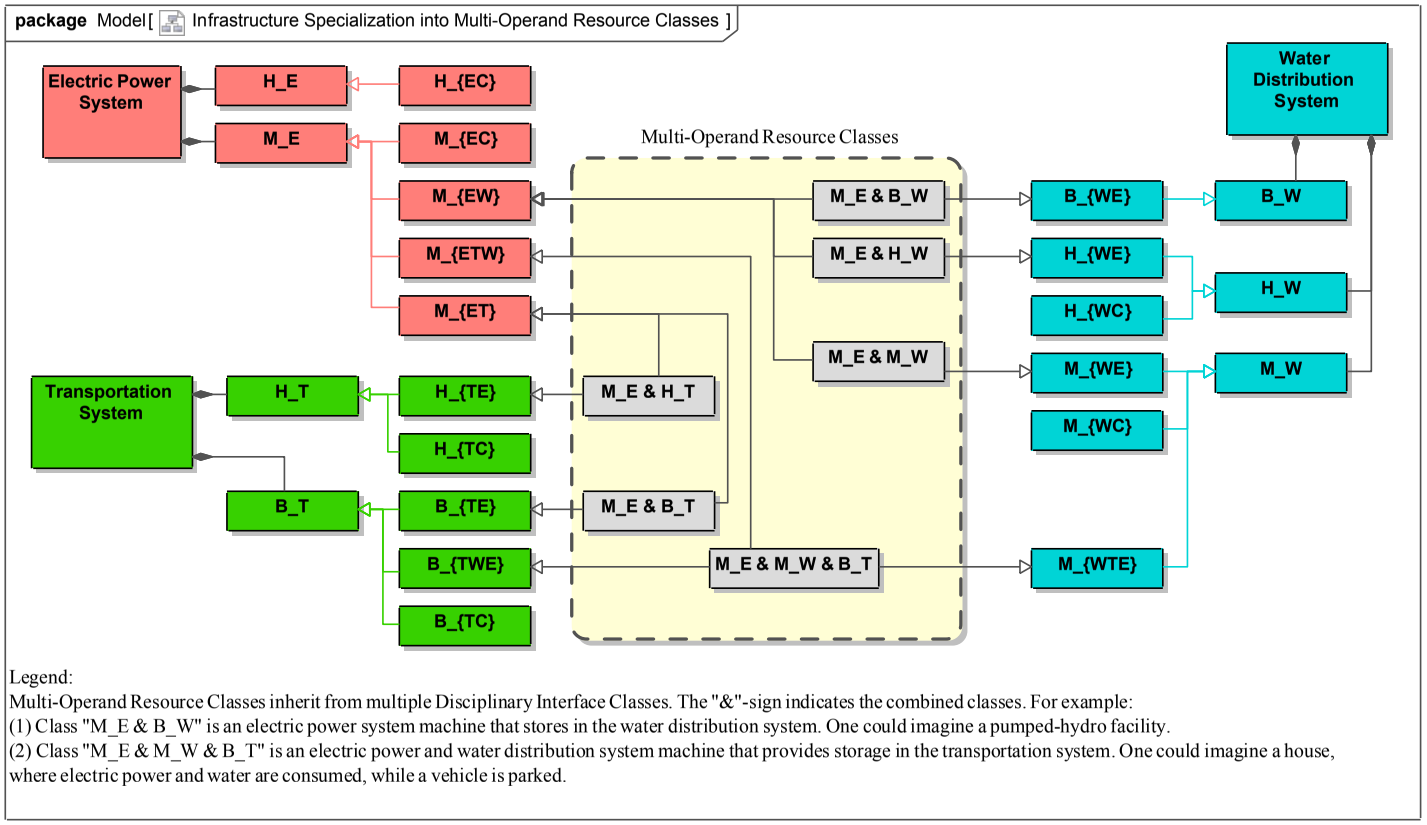


Fig. 4.5: SysML Block Definition Diagram of the Trimetrica infrastructure systems specialized to define the multi-operand resources that allow the disciplinary systems to interface.

tric power grid. As a result, it is classified as resource class: (M_E&M_W&B_T). In Trimetrica there are 43 instances.

3. “Office with EV charger" – is part of all three infrastructure systems. From a transportation system perspective, this resource is a buffer as it stores an EV temporarily. From a water distribution system perspective, this resource is a transformation resource, as it takes water across the system boundary. From an electric power system perspective, this resource is a transformation resource, as it is a load on the electric power grid. As a result, it is classified as resource class: (M_E&M_W&B_T). In Trimetrica there are 5 instances.

4. “House without parking" – is part of the water distribution system and the electric power system. From a water distribution system perspective, this resource is a transformation resource, as it takes water across the system boundary. From an electric power system perspective, this resource is a transformation resource, as it is a load on the electric power grid. As a result, it is classified as resource class: (M_E&M_W). In Trimetrica there are 62 instances.

5. “Water storage facility" – is part of the water distribution system and the electric power system. From a water distribution system perspective, this resource is an independent buffer, as it stores water. From an electric power system perspective, this resource is a transformation resource, as its pumps are a load on the electric power grid. As a result, it is classified as resource class: (M_E&B_W). In Trimetrica there are 10 instances.

6. “Electrified road" – is part of the transportation system and the electric power system. From a transportation system perspective, this resource is a transportation resource as it transports the vehicle from its origin to its destination. From an electric power system perspective, this resource is a transformation resource, as it is a load on the electric power grid. As a result, it is classified as resource class: (M_E&H_T). In Trimetrica there are 52 instances.

7. “Power plant" – is part of the electric power system. From an electric power system perspective, this resource is a transformation resource, as it generates power and adds it to the electric power grid. As a result, it is classified as resource class: M_{EC}. In Trimetrica there is one instance.

8. “Substation" – is part of the electric power system. From an electric power system perspective, this resource is a transformation resource, as all substations consume at least a minor amount of power. Additionally, substations are conventionally modeled as load buses in power systems engineering. As a result, it is classified as resource class: M_{EC}. In Trimetrica there are 23 instances.

9. “Intersection" – is part of the transportation system. From a transportation system perspective, this resource is a buffer as it stores a vehicle temporarily while it redirects from one road to another. As a result, it is classified as resource class: B_{TC}. In Trimetrica there are 116 instances.

10. “Conventional road" – is part of the transportation system. From a transportation system perspective, this resource is a transportation resource as it transports the vehicle from its origin to its destination. As a result, it is classified as resource class: H_{TC}. In Trimetrica there are 520 instances.

11. “Electric power line" – is part of the electric power system. From an electric power system perspective, this resource is a transportation resource, as it transports electric power from its origin to its destination. As a result, it is classified as resource class: H_{EC}. In Trimetrica there are 200 instances.

12. “Directed water pipeline" – is part of the water distribution system. From a water distribution system perspective, this resource is a transportation resource, as it transports water from its origin to its destination. As a result, it is classified as resource class: H_{WC}. In Trimetrica there are 167 instances.

13. “Undirected water pipeline" – is part of the water distribution system. From a water distribution system perspective, this resource is a transportation resource, as it transports water from its origin to its destination. As a result, it is classified as resource class: H_{WC}. In Trimetrica there are 19 instances.

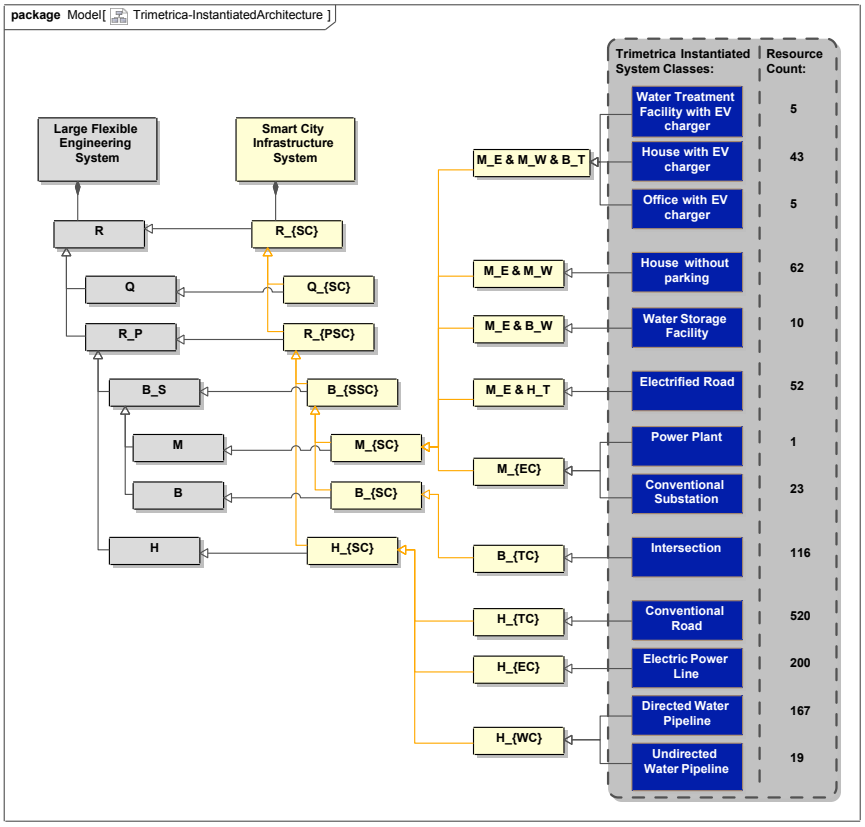


Fig. 4.6: SysML Block Definition Diagram of the Trimetrica smart city infrastructure system as a specialization of the LFES meta-architecture: This figure shows that Trimetrica’s smart city infrastructure system is a single system, rather than three separate systems. The three systems, each classified as an LFES in Figure 4.3, are reconciled into a single smart city infrastructure, of type LFES.

Figure 4.6 provides an overview of the resource structure in Trimetrica[5]. Additionally, the figure shows that the resource structure of the integrated smart city infrastructure system is a type of LFES. The set of Smart City Resources is: $R_{SC} = (M_E \& M_W \& B_T) \cup (M_E \& M_W) \cup (M_E \& B_W) \cup (M_E \& H_T) \cup M_{EC} \cup B_{TC} \cup H_{TC} \cup H_{EC} \cup H_{WC}$. These sets are

[5]Note that the cyber-resources Q_{SC} are discussed in Section 4.5 on Page 171

mutually exclusive and collectively exhaustive as they contain all resources, without overlap. The 9 resource sets have the following sizes:

- $\sigma(M_E \& M_W \& B_T) = 53$. The set contains 5 water treatment facilities, 5 offices, and 43 houses, all with EV chargers.
- $\sigma(M_E \& M_W) = 62$. The set contains 62 houses without parking or EV chargers.
- $\sigma(M_E \& B_W) = 10$. The set contains 10 water storage facilities, without parking or EV chargers.
- $\sigma(M_E \& H_T) = 52$. The set contains 52 electrified roads, able to transport vehicles, and charge vehicles in both directions[6].
- $\sigma(M_{EC}) = 24$. The set contains 23 conventional substations, that all impose an electric load on the power grid, and a power plant that generates electric power.
- $\sigma(B_{TC}) = 116$. The set contains 116 intersections.
- $\sigma(H_{TC}) = 520$. The set contains 520 conventional roads.
- $\sigma(H_{EC}) = 200$. The set contains 200 power lines.
- $\sigma(H_{WC}) = 186$. The set contains 167 directed water pipelines, and 19 undirected water pipelines.

The set of smart city resources R_{SC} has a size of 1,223. The set of smart city buffers B_{SSC} has a size of 317, the set of smart city transformation resources M_{SC} has a size of 201, and the set of smart city independent buffers B_{SC} has a size of 116. Lastly, the set of smart city transportation resources H_{SC} has a size of 906. The set of smart city system resources has been derived, and the next step is the derivation of the smart city system processes.

[6]Note that contrary to set H_{TC}, this set has one resources per edge with a bidirectional capability.

4.3.2 Smart City Processes

The definition of Trimetrica's System Processes follows an approach similar to the definition of the System Resources. First, the Large Flexible Engineering Systems' Meta-Architecture is introduced, to then discuss its specializations in the three infrastructure systems. Thereafter, the system processes are reconciled to define the final set of Trimetrica's system processes.

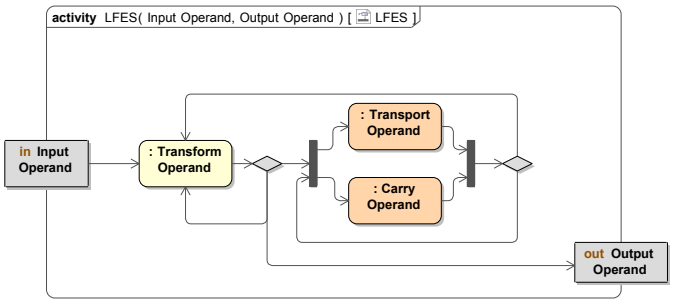


Fig. 4.7: Activity Diagram of the LFES Meta-Architecture: The diamonds represent exclusive decisions. For example, after "Transform Operand" one of three options must be chosen: (1) "Transform Operand", (2) "Transport Operand with Carry Operand", or (3) End the sequence by creating: "Output Operand".

The LFES meta-architecture for system processes is presented using a SysML Activity Diagram. Figure 4.7 restates the meta-architecture as introduced in Section 3.1.2 on Page 65. The functional meta-architecture is a set of processes, classified as transformation processes, transportation processes, and holding processes. Their sequence is unrestricted as a result of a lack of general, cross-disciplinary constraints. For the specialized reference architectures, however, process sequence is constrained by the limitations specific to the reference architecture. This section introduces the design pattern for each of Trimetrica's three infrastructure disciplines separately. Thereafter, the design patterns are reconciled to create a triple-infrastructure design pattern. Trimetrica is an instantiation of this triple-infrastructure design pattern. The set of Trimetrica's instantiated system processes is therefore derived from its design pattern.

The *water distribution system* reference architecture is presented as an activity diagram in

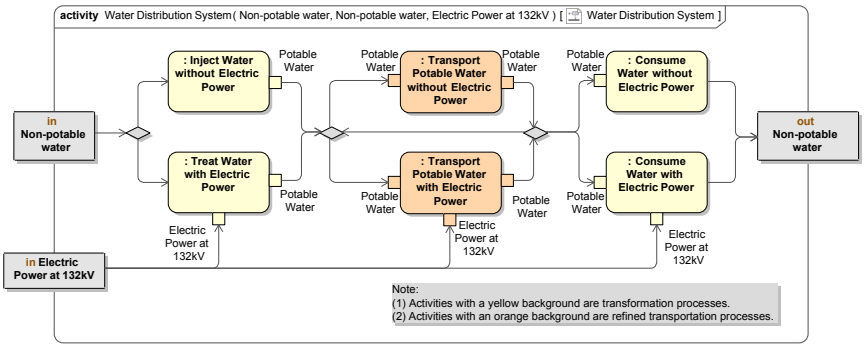


Fig. 4.8: Activity Diagram of the Water Distribution System Reference Architecture.

Figure 4.8. The operand crosses the system boundary when entering and leaving the system which requires transformation processes. A distinction is made between input and output transformations. The input and output transformations are classified as power consuming or non-power consuming. Examples of the former are (1) the extraction of ground water, and (2) hot water consumption. An example of the latter is a river that flows through a city. Note that the power consuming processes require the input of electric power which is transformed to perform work or generate heat.

The location of the inputs and outputs of the water distribution system are physically separated. Water treatment is usually centralized, whereas the consumption of water is highly distributed. Consequently, transportation processes are required to facilitate the distribution of water throughout the smart city. These transportation processes can be refined to differentiate power consuming from non-power consuming transportation processes. The refined transportation processes are presented in Figure 4.8. An example of a power consuming transportation process is a pump, whereas a non-power consuming transportation process is a regular water pipeline. In the Trimetrica test case, the water storage process is modeled as a refined transportation process that consumes electric power.

The *electric power system* reference architecture is presented as an activity diagram in Figure 4.9. Similar to the water distribution system, the electric power system operands enter and leave the system by virtue of transformation processes. These transformation processes are distinctive in type and location, and are therefore separated in the reference

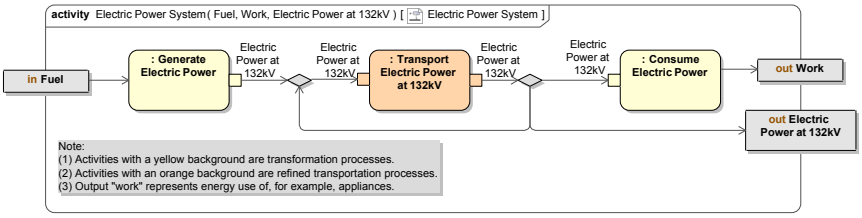


Fig. 4.9: Activity Diagram of the Electric Power System Reference Architecture.

architecture as “generate electric power" (electric power entering the system) and “consume electric power" (electric power leaving the system). Note that the activity diagram indicates that some power is crossing the system boundary as an output to perform work in the other infrastructure systems. This emphasizes the need for a holistic assessment of the water distribution system, electric power system, and the transportation system. Transportation processes necessarily occur between generation and consumption, as these processes are physically separated. Electric power is conventionally expressed as a “per unit" voltage relative to the designed line capacity. Consequently, there is no need to differentiate between the holding processes in the electric power system in this example.

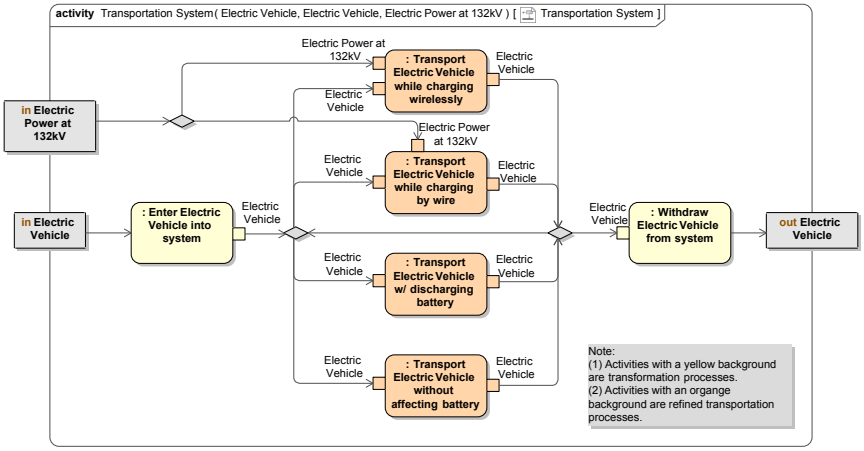


Fig. 4.10: Activity Diagram of the Electrified Transportation System Reference Architecture.

The *electrified transportation system* design pattern is presented as an activity diagram in Figure 4.10. The transportation system’s core function is to transport vehicles to locations in the smart city. However, the transportation system may also receive vehicles from other

locations, or accommodate vehicles that leave the city. The input and output processes are accommodated by transformation processes. The transportation processes are refined to represent the types of transportation with electric vehicles as the operand. Electric vehicles discharge while driving, but the impact of parking on the battery capacity is negligible. Additionally, the electric vehicles need to be charged by wire when they are stationary, or wirelessly if they are charged while driving. The power consumption of these charging processes is facilitated by the electric power system, and requires the input of electric power. Similar to the water distribution system, the charging processes transform the electric power into (portable) charge.

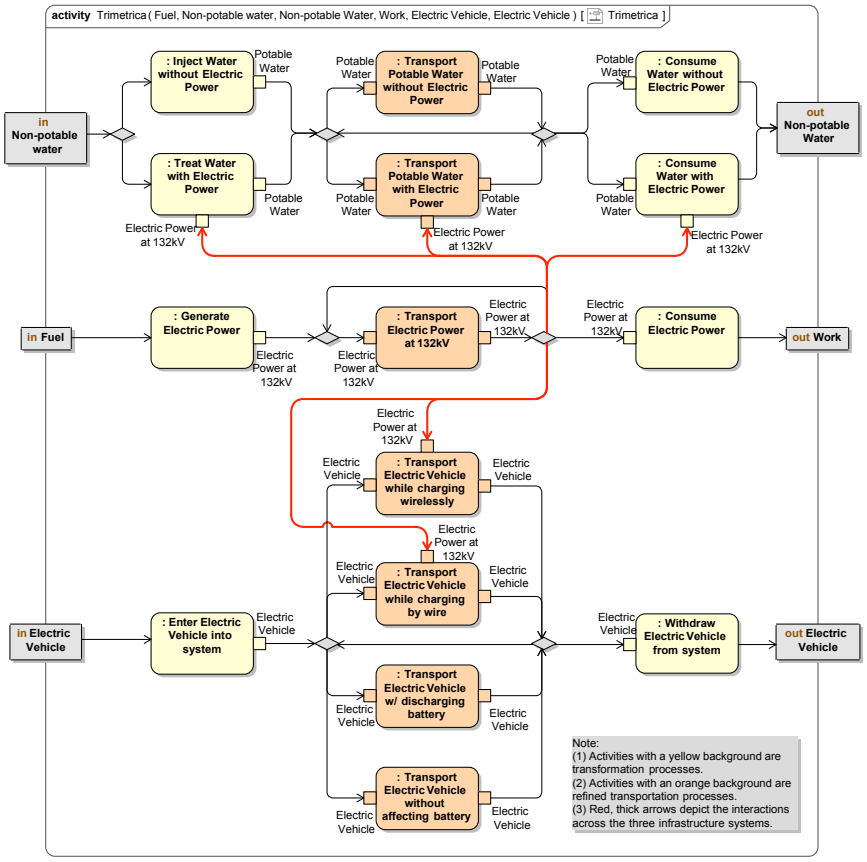


Fig. 4.11: Activity Diagram of a Triple Operand Smart City Infrastructure System Reference Architecture: The three operands are water, electric power, and electric vehicles.

The *triple-infrastructure smart city* design pattern is a combination of each of the infrastructure's individual design patterns. The design pattern is displayed as an activity

diagram in Figure 4.11. Note that electric power is consumed internally, whereas Figure 4.9 showed it as an output. Also note that the consumption of water by the electric power generator (for reasons such as cooling) has been omitted, because the water distribution system in Trimetrica is a *potable* water distribution system.

After the definition of the holistic reference architecture for the smart city infrastructure system in Figure 4.11, Trimetrica's instantiation can be derived. Trimetrica's system processes are classified as transformation, transportation, and holding processes, and each of these classes is now discussed in turn.

Trimetrica's *transformation processes* occur at the locations where operands flow into or out of the system. First, there are two input processes in Trimetrica. The water treatment facilities are the input sources of potable water, and use the process "treat water" to inject potable water across the system boundary. The input of electric power is performed by a single resource with a single "generate electric power" process. The transportation system is different in that it is assumed to be a closed system, without electric vehicles entering or leaving the system. As a result, no transformation processes are required for Trimetrica's transportation system.

The output of the system is determined by the consumption patterns of the distributed consumption resources. Water is consumed in houses and offices, with two different consumption types: consume hot water and consume cold water. The water consumption processes are, for simplicity, assumed to be equal across the locations, resulting in a total of 2 transformation processes. The electric power consumption processes are also assumed to be equal across all consumption locations, and is thus added as a single transformation process. In conclusion, Trimetrica's set of system transformation processes is defined as $P_{\mu SC}$ ={treat water, generate electric power, consume hot water, consume cold water, consume electric power}.

Trimetrica's *refined transportation processes* require a discussion of both the holding processes and the transportation processes. Trimetrica's holding processes are derived

following the three conditions that distinguish between transportation processes that:

- Have *different* operands,
- Hold a given operand in a *given* way.
- Change the *state* of the operand.

Trimetrica consists of three infrastructure systems with three separate operands. Consequently, the holding processes need to differentiate between those operands. The holding processes do not need to differentiate for the different ways to "hold" the operand in a given way. However, both the water distribution system and the transportation system differentiate the ways in which the state of the operand is changed. For the water distribution system, the water storage processes are assumed to consume electric power to, for example, pump the water to a water storage facility. Consequently, there are two holding processes for water: (1) carry potable water and (2) carry potable water while consuming electric power.

The transportation system's operands are electric vehicles. There are four different types of movement of electric vehicles in Trimetrica [31, 32, 50, 261]. First, the vehicles can be stationary; without anything really changing. This is generally referred to as "parking". Second, the vehicles can be driving; in which case the battery of the vehicle will discharge. Third, the vehicle can be charged while it is stationary. Usually, this is performed by wire for the fastest charging rates. Last, the vehicle can be charged while it is driving via induction (wireless) charging. Each of these four processes has a different impact on the state of the electric vehicle and should consequently be included as holding processes. In conclusion, Trimetrica differentiates seven holding processes $P_{\gamma SC} = \{$carry potable water, carry potable water while consuming electric power, carry electric power at 132kV, carry electric vehicle without affecting battery, carry electric vehicle while discharging, carry electric vehicle while charging by wire, carry electric vehicle while charging wirelessly$\}$.

The transportation processes in Trimetrica are derived from the number of buffers in the smart city infrastructure system (see Equation 3.2). The number of unique buffers $\sigma(B_{SSC})$

equals 317, and the total number of transportation processes $P_{\eta SC}$ equals $\sigma(B_{SSC})^2$ = 100,489. The refined transportation processes $P_{\bar{\eta}SC}$ can be derived rather straightforwardly as the combination of each of the holding processes with all transportation processes: $\sigma(P_{\gamma SC})\sigma(P_{\eta SC}) = 703,423$. In summary, the Trimetrica test case has the following sets of processes:

- The set of Transformation Processes: $P_{\mu SC}$ has a size of 5.
- The set of Transportation Processes: $P_{\eta SC}$ has a size of 100,489.
- The set of Holding Processes: $P_{\gamma SC}$ has a size of 7.
- The set of Refined Transportation Processes: $P_{\bar{\eta}SC}$ has a size of 703,423.
- The set of System Processes: P_{SC} has a size of 703,428.

4.3.3 Smart City Knowledge Base

After the discussion of system form and system function, the section now uses the system knowledge base to map system function onto system form. Figure 4.12 introduces the SysML representation of system concept, by mapping the functions onto the instantiated resource classes[7]. For the discussion of system concept, the text will, however, refer to the atomic resources sets rather than the instantiated resource classes.

In hetero-functional graph theory, the mapping of system function onto system form is conceived using the design equation, as introduced in Equation 3.6 on Page 71:

$$P = J_S \odot R \tag{4.1}$$

The Trimetrica knowledge base is calculated using the transformation knowledge base

[7]Note that Figure 4.12 includes the refined transportation processes rather than differentiating between regular transportation processes and holding processes.

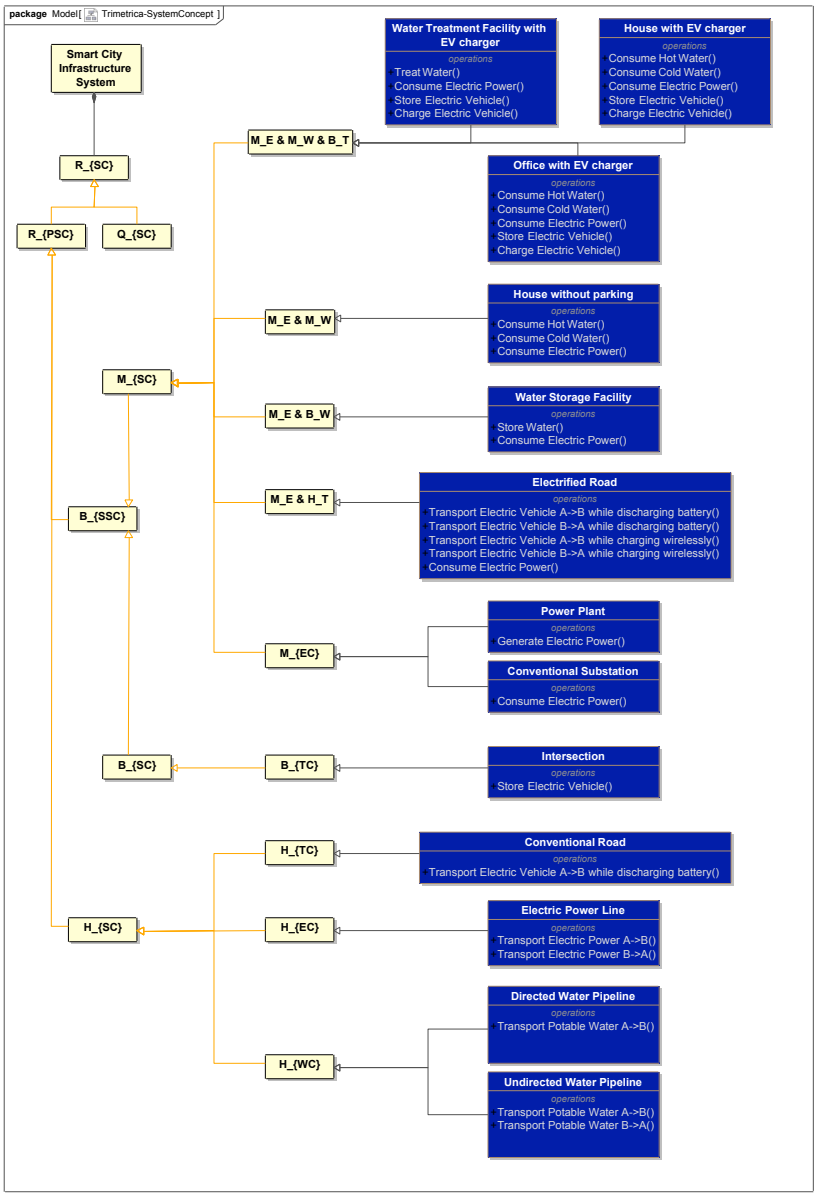


Fig. 4.12: SysML Representation of the System Concept for the Trimetrica Smart City Infrastructure system: This figure contains the unique set of capabilities for each of the resource classes in Trimetrica.

J_M, and the refined transportation knowledge base $J_{\tilde{H}}$, following Equation 3.16 on Page 74:

$$J_S = \left[\begin{array}{c|c} J_M & \mathbf{0} \\ \hline & J_{\tilde{H}} \end{array}\right] \tag{4.2}$$

First, this section derives the *Trimetrica Transformation Knowledge Base*. The system

transformation knowledge base was first introduced in Equation 3.11 on Page 73:

$$P_{\mu} = J_M \odot M \tag{4.3}$$

For Trimetrica, the set of transformation processes $P_{\mu SC}$ has a size of five. The set of transformation resources M_{SC} has a size of 201. The size of matrix J_{MSC} follows as 5×201. The transformation knowledge base contains a degree of freedom for each mapping of a transformation process and a transformation resource. For each of the five transformation processes, the number of degrees of freedom are determined:

- The transformation process “treat water" is mapped onto five water treatment facilities, creating five degrees of freedom.
- The transformation process “generate electric power" is mapped onto the power plant, creating one degree of freedom.
- The transformation process “consume hot water” is mapped onto three resource classes: (1) house with EV charger, (2) office with EV charger, and (3) house without parking, of which Trimetrica contains 43, 5, and 62 respectively. Consume hot water, therefore, maps 110 times on to resources.
- The transformation process “consume cold water" maps onto the same resources as “consume hot water" and also has 110 degrees of freedom associated with it.
- The transformation process “consume electric power" is mapped onto all nodes in the electric power system, except the power plant. The total number of degrees of freedom related to “consume electric power" is thus 200.

In conclusion, the number of transformation degrees of freedom is: $5+1+110+110+200 = 426$.

Trimetrica’s Refined Transportation Knowledge Base provides the mapping of the refined transportation processes $P_{\bar{\eta}}$ onto the system resources R_{SC}, using the design equation

(Equation 4.1). The refined transportation knowledge base can be calculated using the holding knowledge base $J_{\gamma SC}$ and the transportation knowledge base J_{η}, using Equation 3.14 on Page 73:

$$J_{\bar{H}SC} = \left[J_{\gamma SC} \otimes \mathbb{1}^{\sigma(P_{\eta SC})}\right] \cdot \left[\mathbb{1}^{\sigma(P_{\gamma SC})} \otimes J_{HSC}\right] \tag{4.4}$$

where the transportation knowledge base J_{HSC} is derived using Equation 3.12 on Page 73:

$$P_{\eta SC} = J_{HSC} \odot R_{SC} \tag{4.5}$$

and where the holding knowledge base $J_{\gamma SC}$ is derived using Equation 3.13 on Page 73:

$$P_{\gamma SC} = J_{\gamma SC} \odot R_{SC} \tag{4.6}$$

The transportation knowledge base J_{HSC} maps the transportation processes $P_{\eta SC}$ onto the system resources R_{SC}, and has size $\sigma(P_{\eta SC}) \times \sigma(R_{SC})$ (or 100,489 × 1,223). As a result, the transportation degrees of freedom can be divided in two sets: transportation degrees of freedom realized by transportation resources (H), and the transportation degrees of freedom realized by buffers (B_S). The former set of degrees of freedom is derived by matching the origin and destination of a transportation process to the physical origin and destination of the transportation resource. The elements in the set of conventional roads H_{TC} have a dedicated direction and perform a single transportation process. Their total number is 520, and consequently they enable 520 degrees of freedom. The 200 electric power lines (in set H_{EC}), however, allow for a flow of power in either direction, and contain two degrees of freedom per resource. The total number of degrees of freedom associated with the power lines is thus 400, as each power line allows for a transportation process from its origin to the destination and back. The set of water pipelines (H_{WC}) consists of 167 pipelines with direction, and 19 without. Consequently, their total number degrees of freedom is 167+2(19)=205. For all of Trimetrica's transportation resources H_{SC}, the total number of

transportation degrees of freedom is 520+400+205=1,125.

Trimetrica's set of buffers B_{SSC} consist of atomic resource classes that have transportation processes mapped to them. The majority of buffers in Trimetrica have a single location, and do not transport the operands between two different locations. Those sets are: $M_E\&M_W\&B_T$, $M_E\&M_W$, $M_E\&B_W$, M_{EC}, and B_{TC}. From these sets, only $M_E\&M_W$ and M_{EC} do not have the capability to store their respective operands, and therefore don't have a transportation process mapping to them. $M_E\&M_W\&B_T$, $M_E\&B_W$, and B_{TC} all have the ability to store an operand, or transport an operand from origin to destination where the origin equals the destination. For all elements in those three sets, the transportation knowledge base J_{HSC} contains a degree of freedom, which total 53+10+116=179.

There is, however, one set that does transport operands and is still considered a buffer: the set of electrified roads $M_E\&H_T$. This set has an origin and a destination, and allows one type of operand to be transported using the resource (electric vehicles). However, electric power is transformed into charge while the electric vehicles are transported. Consequently, the electrified road has a transformative nature and should be considered as a machine. After all, the classification "machine" supersedes the other classifications. The set of electrified roads (of length 52), therefore, adds 104 transportation degrees of freedom to the transportation knowledge base. In conclusion, the total number of Trimetrica's transportation degrees of freedom is 1,125+179+104=1,408.

The holding knowledge base $J_{\gamma SC}$ maps the holding processes $P_{\gamma SC}$ onto the system resources R_{SC}, and has size $\sigma(P_{\gamma SC}) \times \sigma(R_{SC})$ (or $7 \times 1{,}223$). As introduced previously, the holding processes are used to differentiate transportation processes in operand type, way of holding, or potential transformative nature. Each resource has at least one associated holding degree of freedom. The first set is: $M_E\&M_W\&B_T$. For each of the resources in this set, the resources have two transportation processes: charge and park electric vehicles. Consequently, this set has two degrees of freedom per element, which totals 53*2=106 holding degrees of freedom. The set $M_E\&M_W$ do not have associated transportation (or

storage) processes. Consequently, this set does not have any holding degrees of freedom. The set $M_E\&B_W$ stores water and consumes electric power. The associated holding process to the transportation process is only “store water." Consequently, each of the resources has a single associated holding degree of freedom: 10 in total for the set. The set $M_E\&H_T$ contains the electrified roads. These resources have two ways of transporting electric vehicles: transport electric vehicle while charging wirelessly or while discharging. Consequently, the resources have two associated holding degrees of freedom per element, which totals 52*2=104 holding degrees of freedom. The set M_{EC} does not have associated transportation of holding processes. The remaining sets B_{TC}, H_{TC}, H_{EC}, and H_{WC} all have a single specific way to transport their associated operands and their transportation processes do not change the state of the operand. Consequently, the resources in these sets have a single holding process, with a total of 116+520+200+167+19=1,022. Based on the holding degrees of freedom for all the atomic resource sets, the total number of Trimetrica's holding degrees of freedom is 106+10+104+1,022=1,242.

The refined transportation knowledge base $J_{\bar{H}SC}$ now combines the transportation knowledge base and the holding knowledge base using Equation 4.4. The resulting set of refined transportation processes are mapped onto the set of resources. Figure 4.12 presents the mapping of the refined transportation processes onto the Trimetrica's sets of resources. The resulting matrix has size $\sigma(P_{\bar{\eta}SC}) \times \sigma(R_{SC}) = 703,423 \times 1,223$. The total number of refined transportation degrees of freedom is calculated using Equation 3.22 on Page 76:

$$DOF_{HSC} = \sum_{u}^{\sigma(P_{\bar{\eta}SC})} \sum_{v}^{\sigma(R_{SC})} [J_{\bar{H}SC} \ominus K_{\bar{H}SC}](u,v) \tag{4.7}$$

where K_{HSC} is all zeros. The total number of refined transportation degrees of freedom DOF_{HSC} is 1,565.

In conclusion, the Trimetrica knowledge base J_{SSC}, constraints matrix K_{SSC}, and system concept A_{SSC} have size $703,428 \times 1223$. The total number of degrees of freedom can be

calculated using Equation 3.18 on Page 76, or adjusted for the Trimetrica test case:

$$DOF_{SSC} = \sigma(\mathcal{E}_S) = \sum_{w}^{\sigma(P_{SC})} \sum_{v}^{\sigma(R_{SC})} [J_{SSC} \ominus K_{SSC}](w, v) \tag{4.8}$$

which equals 1,991.

4.3.4 Visualizing Degrees of Freedom

In order to intuitively understand the meaning of Trimetrica's set of degrees of freedom, a visualization is required. The degrees of freedom are a mapping of system function onto system form, and therefore, a mapping of system processes onto system resources. Each of those mappings is a degree of freedom, or a capability of the interdependent infrastructure system. Logically, these system capabilities can be represented per resource while adopting the physical location of their associated resource.

Trimetrica is originally depicted in Figure 4.1 on Page 127. This section compares each of the three separate infrastructure systems to the degrees of freedom associated with these systems. Figure 4.13 provides a comparison between the Trimetrica water distribution system topology and its degrees of freedom. Three observations are made. First, the water treatment facilities have a single capability. This capability acts on two operands, because it consumes electric power to generate potable water. Second, all water consumption nodes, whether they are houses or offices, have two associated degrees of freedom. These represent both hot water consumption and cold water consumption; where the former has two operands, and the latter only consumes potable water as its operand. Lastly, the capabilities of the water pipes are displayed in between the end-points of the water pipe. Whenever the end-points are more than two units of length away, the capability is located at 75% of the pipe's length. Otherwise, the capability is located in the middle. The capabilities associated with the water pipes have potable water as a single operand.

Figure 4.14 provides a comparison between the Trimetrica electric power system topol-

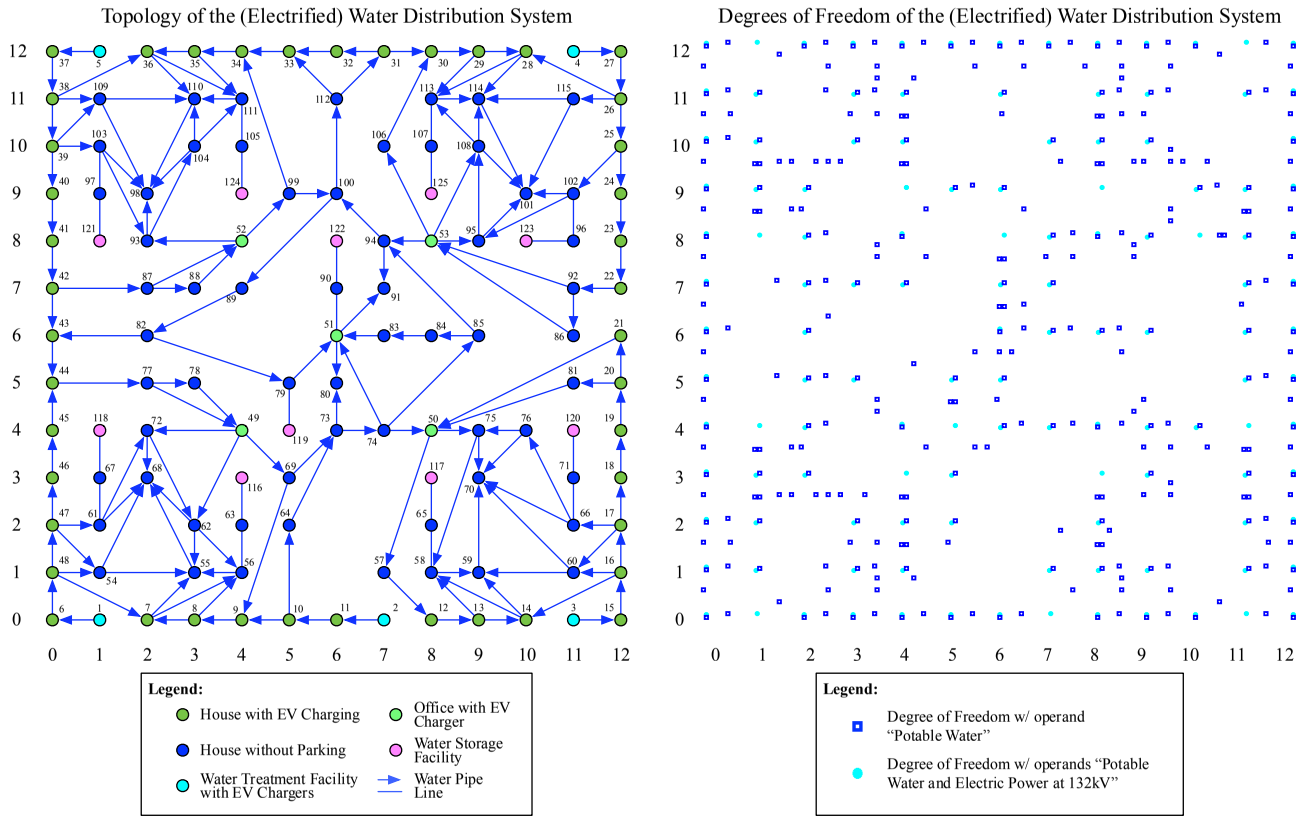


Fig. 4.13: Structural Degrees of Freedom of Water-related Operands.

ogy and its degrees of freedom. Three observations are made. First, there is overlap between Figures 4.13 and 4.14, because the hot water consumption, water treatment, and water storage processes all consume electric power, and transform potable water. Second, all power transmission lines have two associated degrees of freedom to represent the potential two-way flow of electric power. Lastly, the figure contains overlap with Figure 4.15, because it includes the charging capabilities of the transportation system; whether they are charging roads or conventional charge-by-wire facilities.

Figure 4.15 provides a comparison between the Trimetrica electrified transportation system topology and its degrees of freedom. Two observations are made. First, the charging capabilities in pink coincide with either two conventional transportation capabilities or a single parking capability. There are no locations where charging is the sole capability. Second, the intersections have a 'storage' capability. However, its capacity is zero, because parking a vehicle on an intersection is not allowed. Therefore, it represents the connection between two or more roads.

Figure 4.16 superimposes the degrees of freedom of the five operands in Trimetrica. The figure contains a window in the bottom left corner. The enlarged depiction of this window is presented in Figure 4.17, with the topological presentation of the same area. Figure 4.17 presents the degrees of freedom provided by the topology of the six resources. The House 1 (or resource 6) has five associated degrees of freedom:

1. consume cold potable water at house 1
2. consume hot potable water at house 1
3. consume electric power at house 1
4. park electric vehicle at house 1
5. charge electric vehicle by wire at house 1

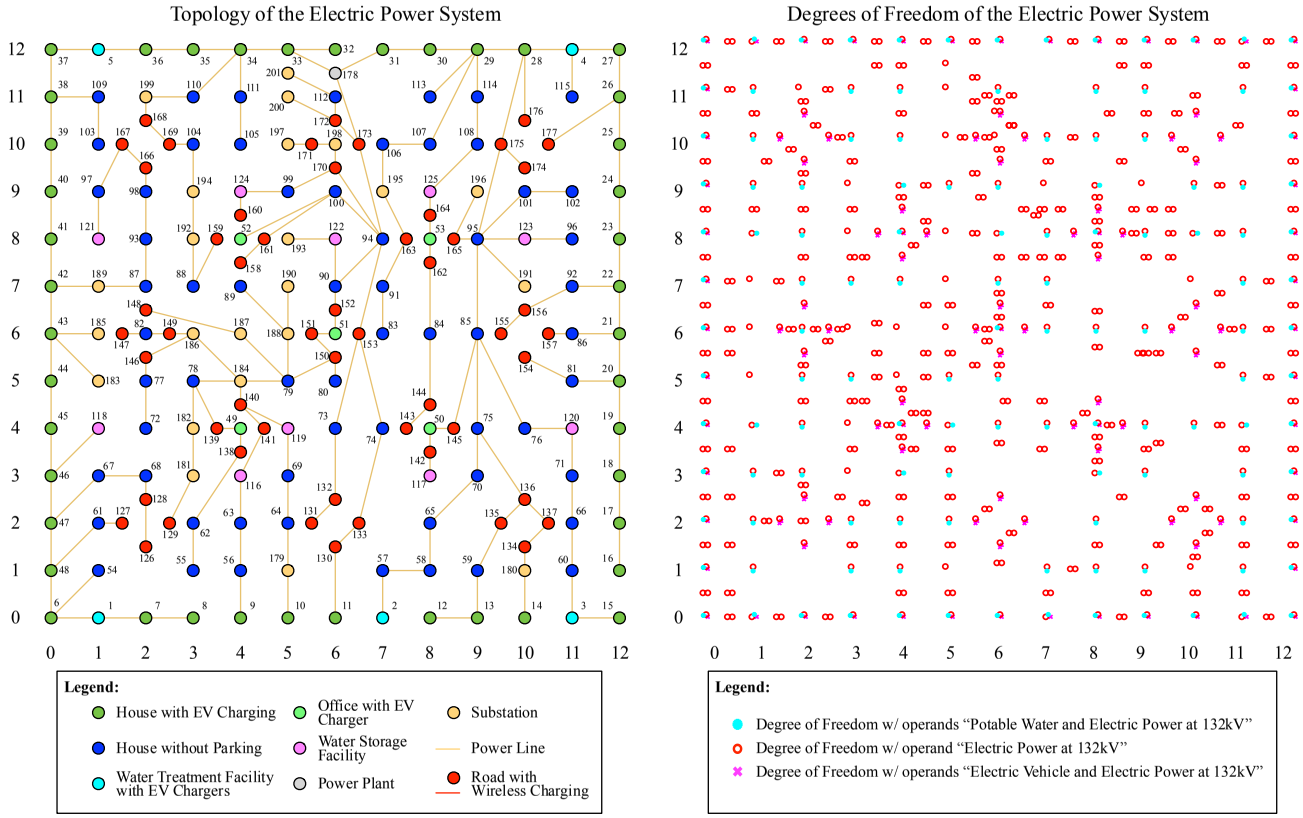


Fig. 4.14: Structural Degrees of Freedom of Electricity-related Operands.

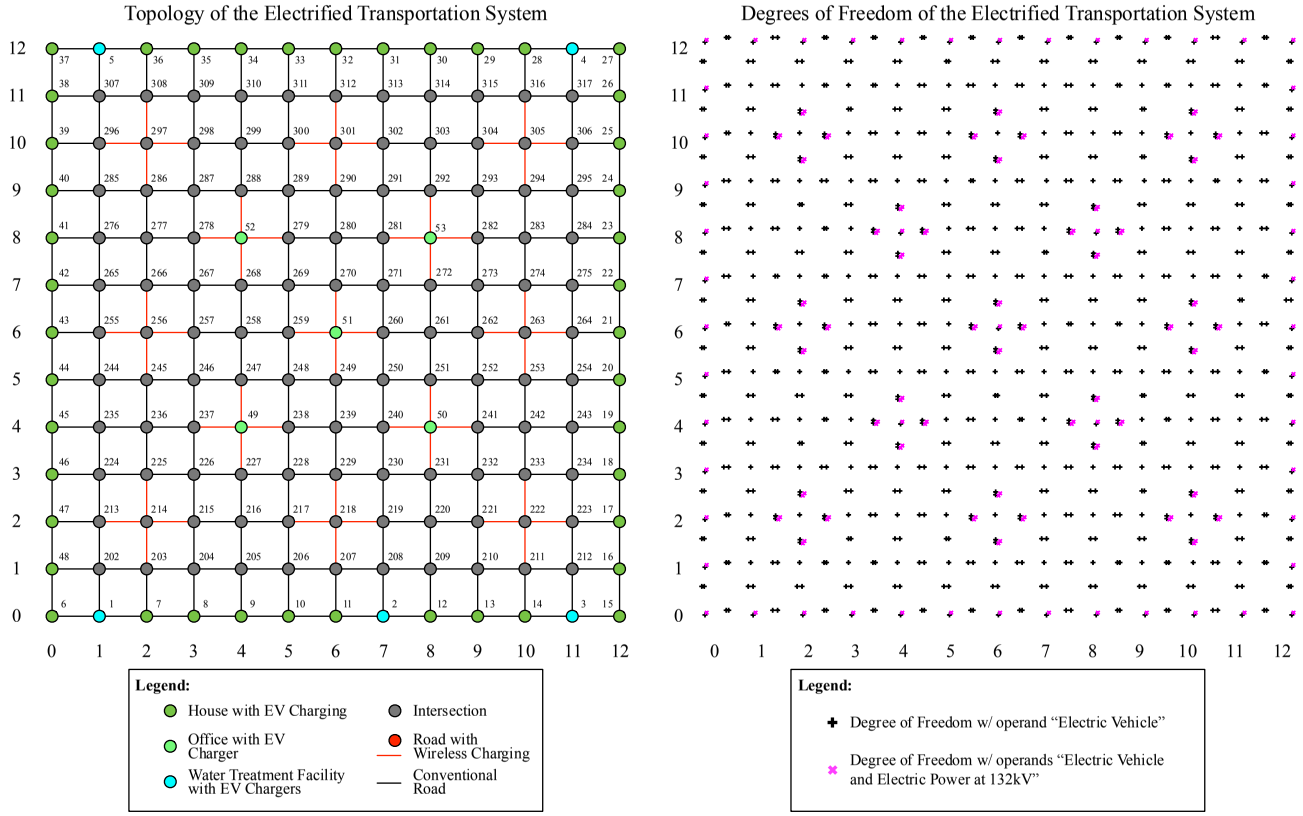


Fig. 4.15: Structural Degrees of Freedom of Transportation-related Operands.

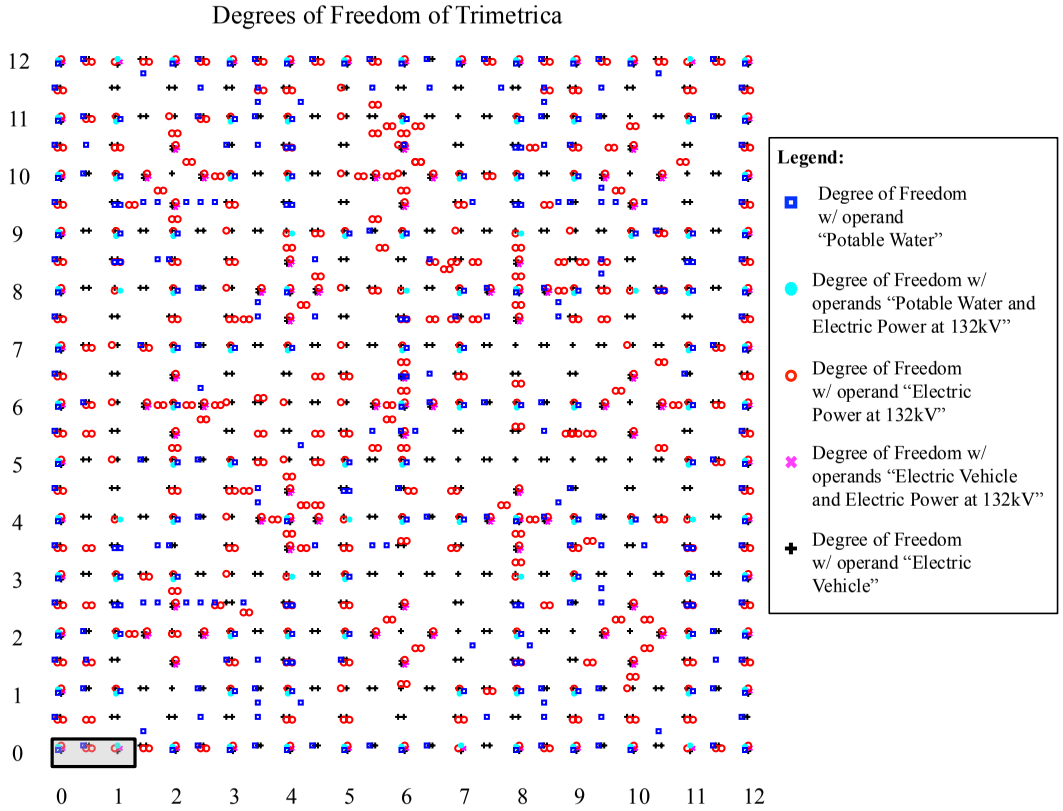


Fig. 4.16: Trimetrica's Structural Degrees of Freedom: The frame in the bottom-left corner indicates the detail presented in Figure 4.17 on Page 160.

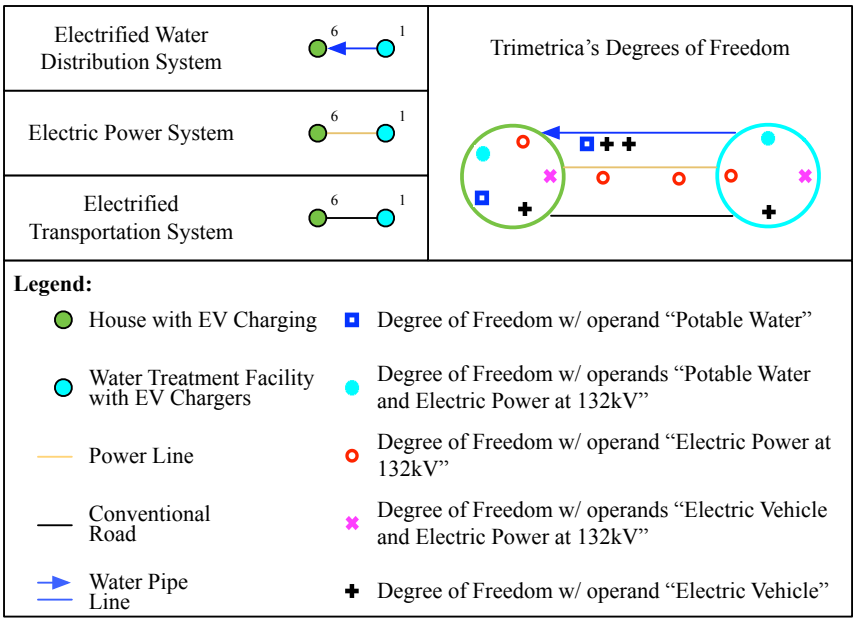


Fig. 4.17: A comparison between a detail of Trimetrica's topology and the same detail of Trimetrica's structural degrees of freedom as indicated in Figure 4.2 on Page 131 and Figure 4.16 on Page 159.

All degrees of freedom are of a different operand type, and are depicted with their associated symbols. Water treatment facility 1 has four associated degrees of freedom:

1. treat water at water treatment facility 1
2. consume power at water treatment facility 1
3. park EV at water treatment facility 1
4. charge EV by wire at water treatment facility 1

Trimetrica contains four different transportation resources between house 1 and water treatment facility 1. The first is a water pipeline from the water treatment facility to the house. The water pipeline transports water from the water treatment facility to the house, and has, therefore, a single associated degree of freedom. Note that the test case assumes the water pipeline is loss-less and does not require additional pressurization. The second resource between the house and the water treatment facility is an electric power line. It

facilitates transmission of electric power between the house and the water treatment facility in both directions. Consequently, there are two electric power degrees of freedom. Lastly, there are two roads, one originating in the house, and the other originating in the water treatment facility. Note that the roads are depicted as a single line. Each of the two roads has a single degree of freedom, and therefore, there are two degrees of freedom of operand class “electric vehicle".

In conclusion, this section first determined system form and system function for Trimetrica using the block definition diagram and activity diagram in SysML. Based on this conceptual framework, the set of system resources and processes were defined. Subsequently, the system knowledge base was calculated as a map of system function onto system form. Lastly, the section visualized the degrees of freedom by a comparison with topological lay-outs of Trimetrica’s water distribution, electric power, and electrified transportation systems.

4.4 Hetero-functional Adjacency Matrix

Following the discussion of Trimetrica’s degrees of freedom, that represent the capabilities of the smart city infrastructure system, this section continues to define their logical sequence. The section applies the theoretical foundation defined in Section 3.2 on Page 79 to the Trimetrica test case. First, the numerical results are discussed based on the application of the mathematical model. Then, the degree of freedom visualizations from the previous section are expanded to incorporate system sequence for a more intuitive understanding.

4.4.1 Calculating System Sequence

In the previous chapter, the Hetero-functional Adjacency Matrix A_ρ was first introduced as the Boolean difference between the system sequence knowledge base J_ρ and the system sequence constraints matrix K_ρ, using Equation 3.25 on Page 80. When applied to the

Trimetrica test case, the equation is:

$$A_{\rho SC} = J_{\rho SC} \ominus K_{\rho SC} \tag{4.9}$$

where $J_{\rho SC}$ is the system sequence knowledge base, and where $K_{\rho SC}$ is the system sequence constraints matrix. The size of matrices $A_{\rho SC}$, $J_{\rho SC}$, and $K_{\rho SC}$ equals: $\sigma(R_{SC})\sigma(P_{SC}) \times \sigma(R_{SC})\sigma(P_{SC})$, or numerically: $860,292,444 \times 860,292,444$.

Note that the system sequence knowledge base J_ρ defines all combinations of degrees of freedom (as in Definition 3.14 on Page 80). When Equation 3.26 is applied to the Trimetrica test case, $J_{\rho SC}$ is calculated following:

$$J_{\rho SC} = A^{V}_{SSC} A^{VT}_{SSC} \tag{4.10}$$

where $J_{\rho SC}$ has $DOF^2_{SSC} = 1991^2 = 3,964,081$ filled elements.

Since not all combinations of degrees of freedom are feasible, the system sequence constraints matrix is introduced (as in Definition 3.15 on Page 81). The system sequence constraints matrix enforces two types of sequence constraints on the system sequence knowledge base. First, it imposes the physical continuity constraints presented in Table 3.3 on Page 82. Second, it imposes the functional sequence constraints which are based on the reference architecture (or design pattern) of the modeled system.

The system sequence constraints matrix is calculated using the aforementioned constraints. Even though this approach is correct, it is not always practical from a computational perspective, because it requires $\left[\sigma(R)\sigma(P)\right]^2 = 860,292,444^2$ calculations. Since the $J_{\rho SC}$ matrix is sparse, it is computationally more efficient to check each of the logical DOF sequences for compliance with the imposed constraints. The latter approach facilitates the computation of $A_{\rho SC}$, avoids the calculation of the memory-intensive $K_{\rho SC}$ matrix, and requires only $DOF^2_{SSC} = 1991^2$ calculations. If the logical sequence of DOFs complies with the constraints, it is transferred to $A_{\rho SC}$. The calculation of the Hetero-functional

Adjacency Matrix for the Trimetrica test case uses this latter approach. The number of DOF sequences that comply with each of the physical continuity constraints is listed below:

- $DOF_{R1} = 1096$
- $DOF_{R2} = 2309$
- $DOF_{R3} = 2314$
- $DOF_{R4} = 10{,}670$

This reduces the total number of sequence degrees of freedom that adhere to the physical continuity constraints to 16,389.

The DOF sequences that comply with the functional sequence constraints are derived from the Trimetrica design pattern in Figure 4.11. The total number of DOF sequences that comply with the design pattern is 1,348,050.

The intersection of these two sets of DOF sequences are included in $A_{\rho SC}$. The resulting number of sequence degrees of freedom that comply with both the functional sequence and continuity constraints $DOF_{\rho SC}$ is 8,274.

Note that the hetero-functional adjacency matrix can be projected to reduce its size to more manageable levels. Equation 3.31 on Page 83 demonstrates the projection of any hetero-functional adjacency matrix. When applied to the Trimetrica test case, it follows:

$$\mathbb{P}_S A_{\rho SC} \mathbb{P}_S^T = \tilde{A}_{\rho SC} \tag{4.11}$$

The size of the projected matrix is $\sigma(\mathcal{E}_S) \times \sigma(\mathcal{E}_S)$. No information is lost in this process, and the number of sequence degrees of freedom remains naturally unchanged.

4.4.2 Visualizing System Sequence

Visualizations of the HFAM serve to facilitate its understanding. Figures 4.13, 4.14, and 4.15, which present DOFs as a collection of scattered nodes, are now revised with connecting

arcs that represent system sequence. Note that the arcs are all directional.

Figure 4.18 on page 165 compares the electrified water distribution system topology with the hetero-functional adjacency matrix. For clarity, this figure displays the HFAM such that only degrees of freedom associated with the operand “potable water" are depicted. Clearly, the visualization of the hetero-functional adjacency matrix closely resembles the topology of the water distribution system. Naturally, the physical interfaces between the resources (as depicted in the topology) require associated sequences of degrees of freedom (as depicted in the HFAM).

Figure 4.19 on page 166 compares the electric power system with the hetero-functional adjacency matrix, including only degrees of freedom associated with the operand “electric power". Other than the electrified water distribution system, the transportation resources each have two degrees of freedom to accommodate the bidirectional flow of electric power. Additionally, one can notice that all degrees of freedom are coupled, and consequently, all degrees of freedom are able to connect to the electric power system. This is essential, because the water consumption, treatment, and storage degrees of freedom, as well as the EV charging degrees of freedom rely on the power grid for their operation.

Figure 4.20 on page 167 compares the electrified transportation system with the hetero-functional adjacency matrix, including only degrees of freedom associated with the operand “electric vehicle". Note that the degrees of freedom of the roads are connected in a diamond like pattern. This is caused by the sequences in which two transportation processes around the same intersection are coupled.

Based on the previous three figures, it becomes clear that the degrees of freedom are coupled across the operand types. Conventionally, in graph theory, the systems with different operand types have been separated into layers per operand [135]. As discussed in Section 2.2.2, this practice limits the way the operand-specific networks are coupled. Hetero-functional graph theory, however, allows the coupling of nodes across layers based on physical continuity and functional sequence. In the end, an interdependent infrastructure

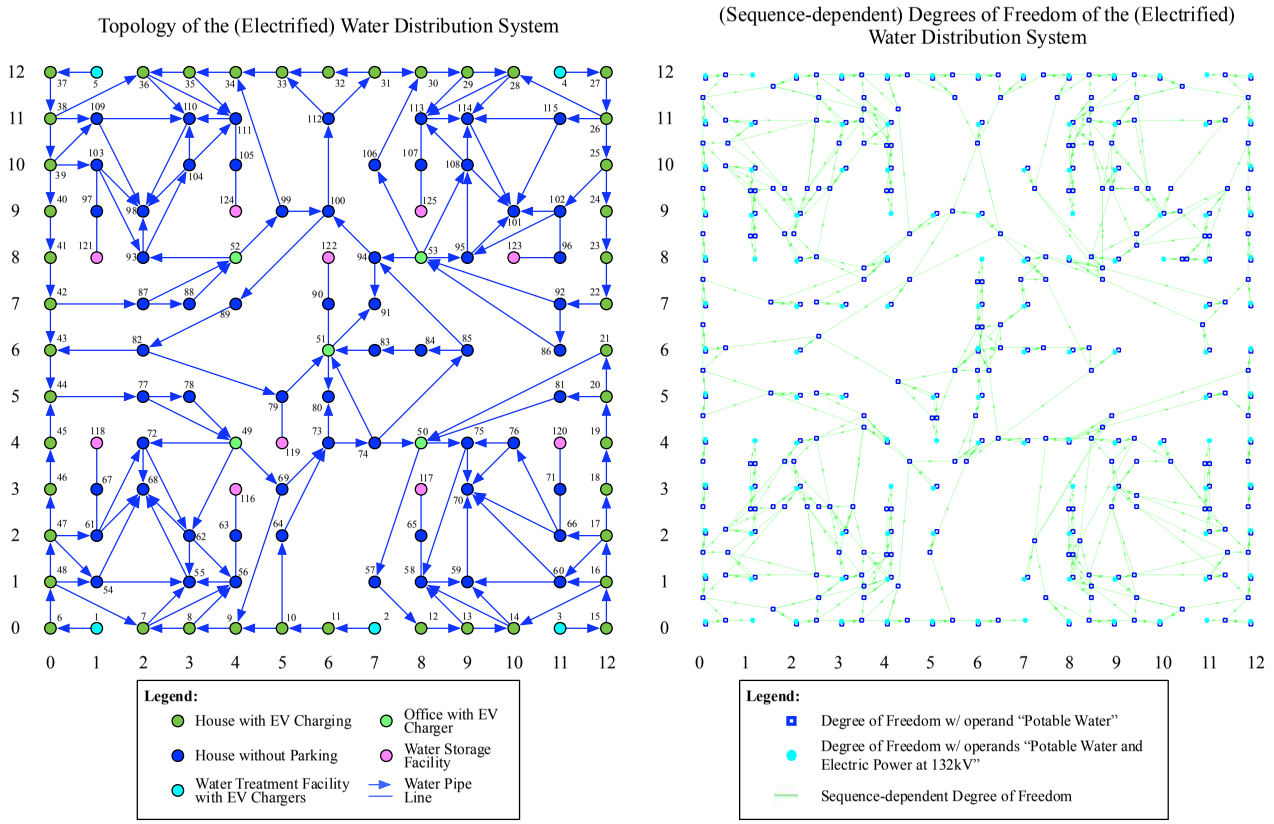


Fig. 4.18: Sequence-dependent Degrees of Freedom of the operands: water, and water with electric power at 132kV.

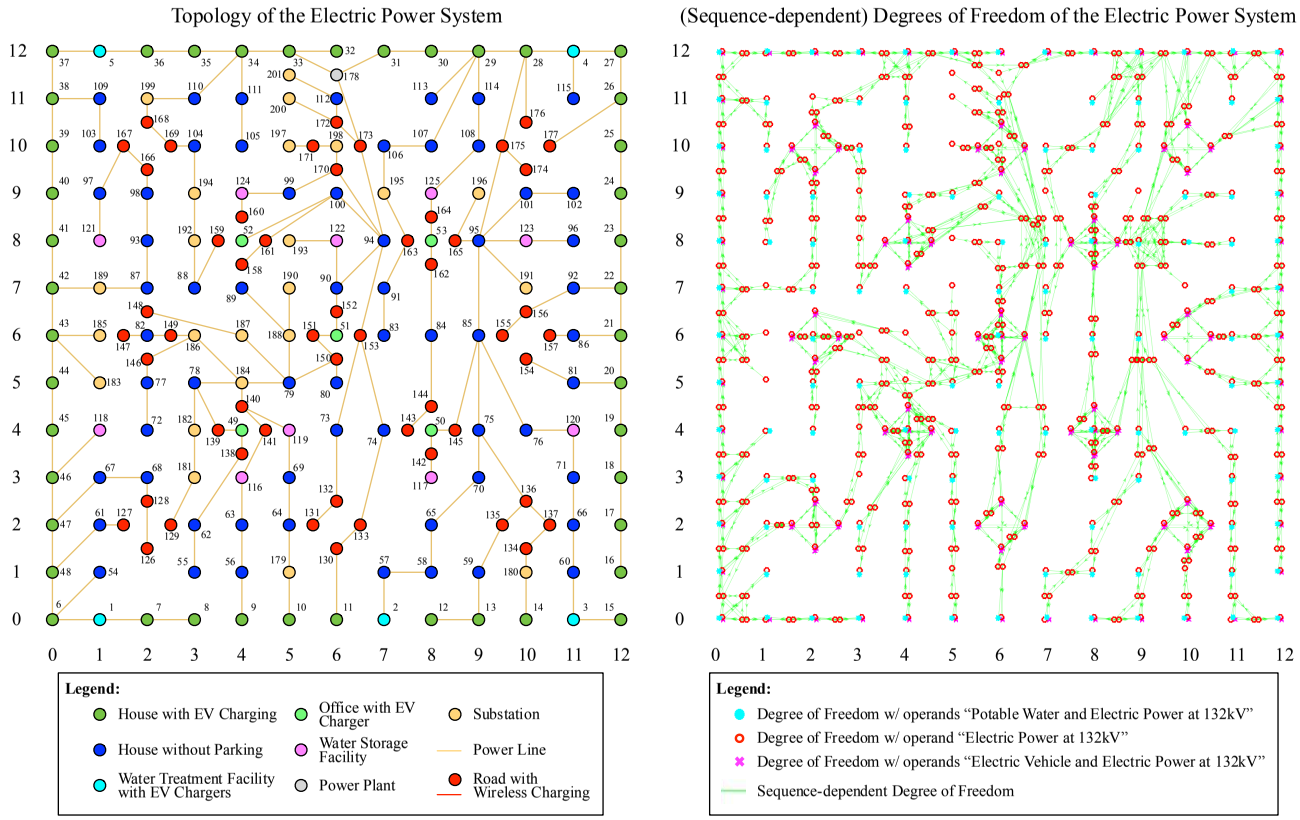


Fig. 4.19: Sequence-dependent Degrees of Freedom of the operands: Electric power at 132kV, Water with electric power at 132kV, and EV with electric power at 132kV.

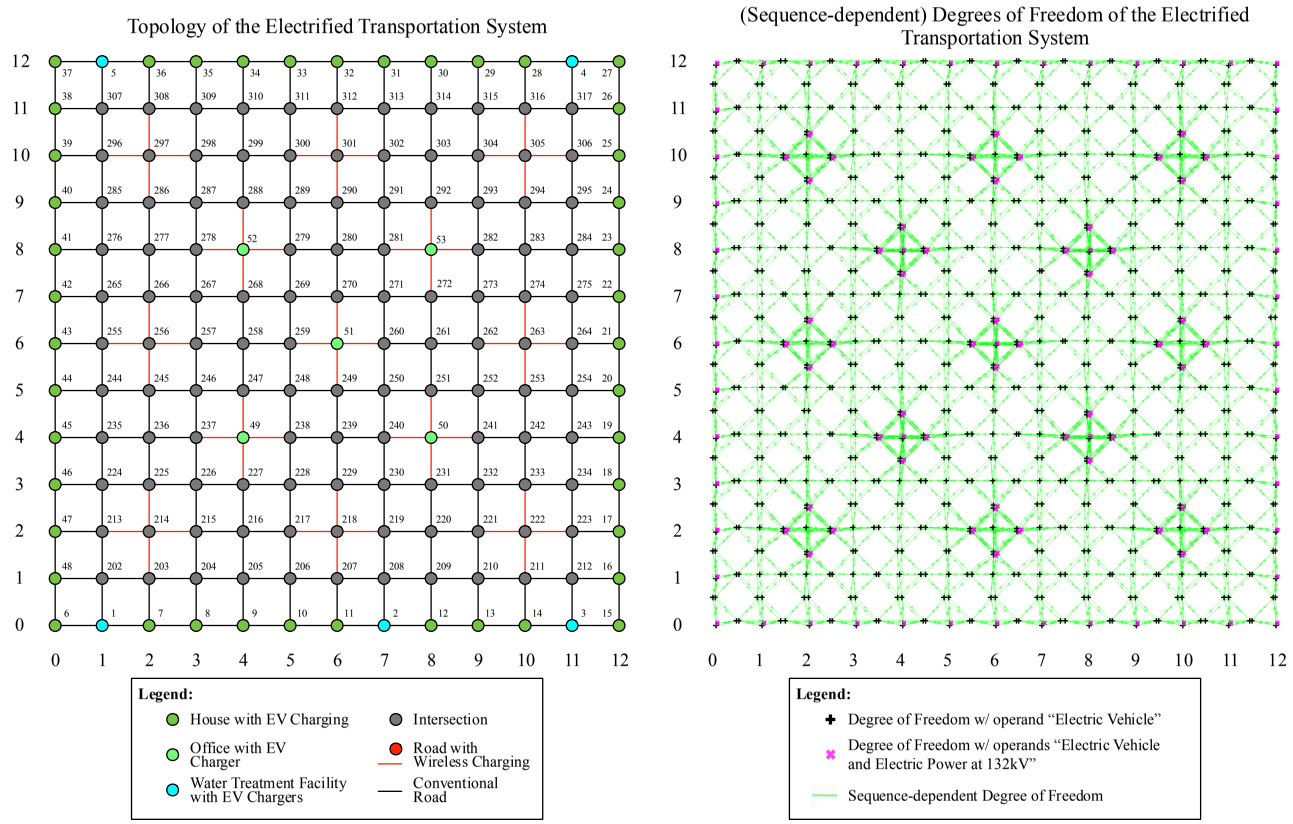


Fig. 4.20: Sequence-dependent Degrees of Freedom of the operands: EV, and EV with electric power at 132kV.

system is a single system; rather than a number of separate systems. Hetero-functional graph theory reflects this reality as a single network in which function, form, and operand type are uniquely defined. Figure 4.22 on page 169, now, presents the full hetero-functional adjacency matrix for Trimetrica. The complexity of the Trimetrica interdependent smart city infrastructure system creates a very crowded representation. Figure 4.21 on Page 168 provides a detail of the Trimetrica hetero-functional adjacency matrix, in which 25 sequences of 14 degrees of freedom are provided. This figure builds on Figures 4.2 and 4.17 on pages 131 and 160 respectively.

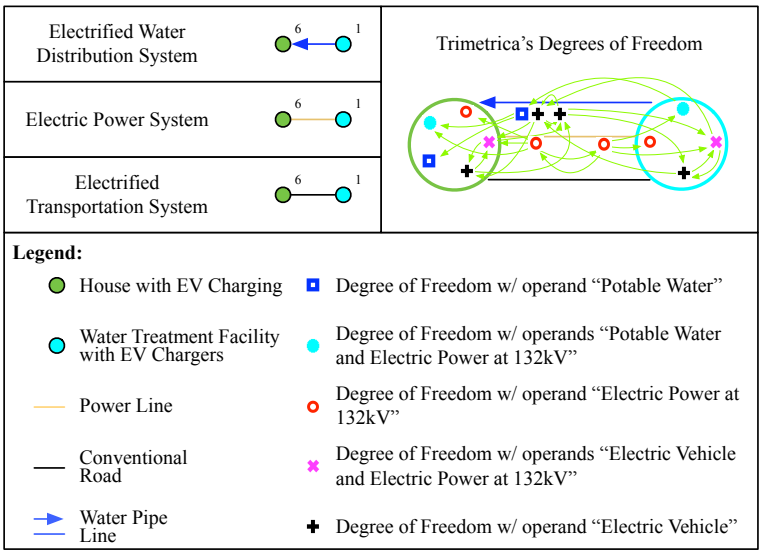


Fig. 4.21: A comparison between a detail of Trimetrica's topology and the same detail of Trimetrica's structural degrees of freedom as indicated in Figure 4.2 on Page 131 and Figure 4.16 on Page 159.

Figure 4.23 on page 170 represents the hetero-functional adjacency matrix as a network with five distinct layers. Each of the layers contains a unique set of degrees of freedom classified by operand type. System sequence across as well as within all five operand classes is provided.

In conclusion, this section has provided an overview of the mathematical derivation of the hetero-functional adjacency matrix for the Trimetrica test case. First, the set of feasible sequence degrees of freedom was calculated and captured in the hetero-functional

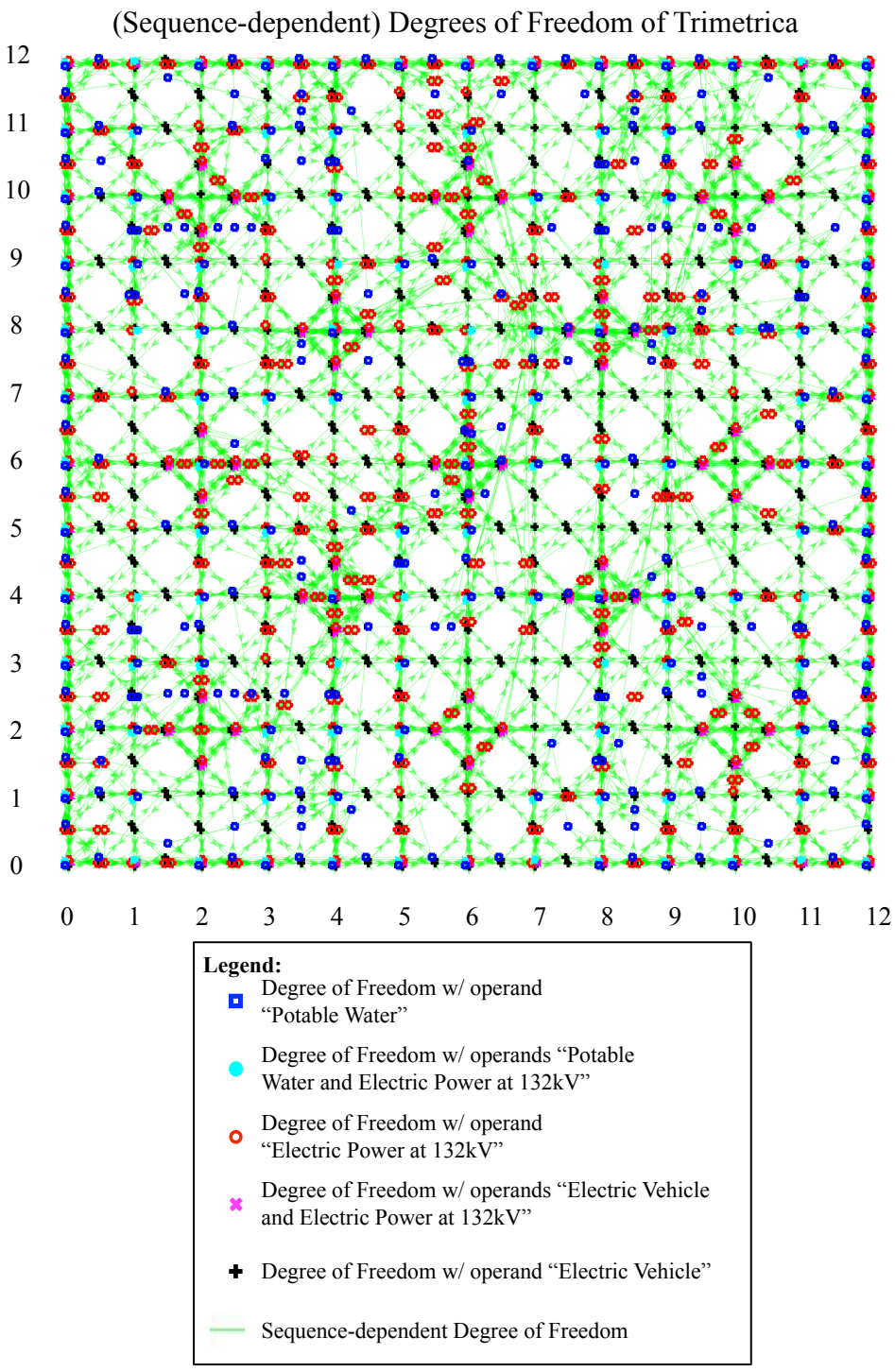


Fig. 4.22: Trimetrica's Hetero-functional Adjacency matrix with all five layers of degrees of freedom in a single plane.

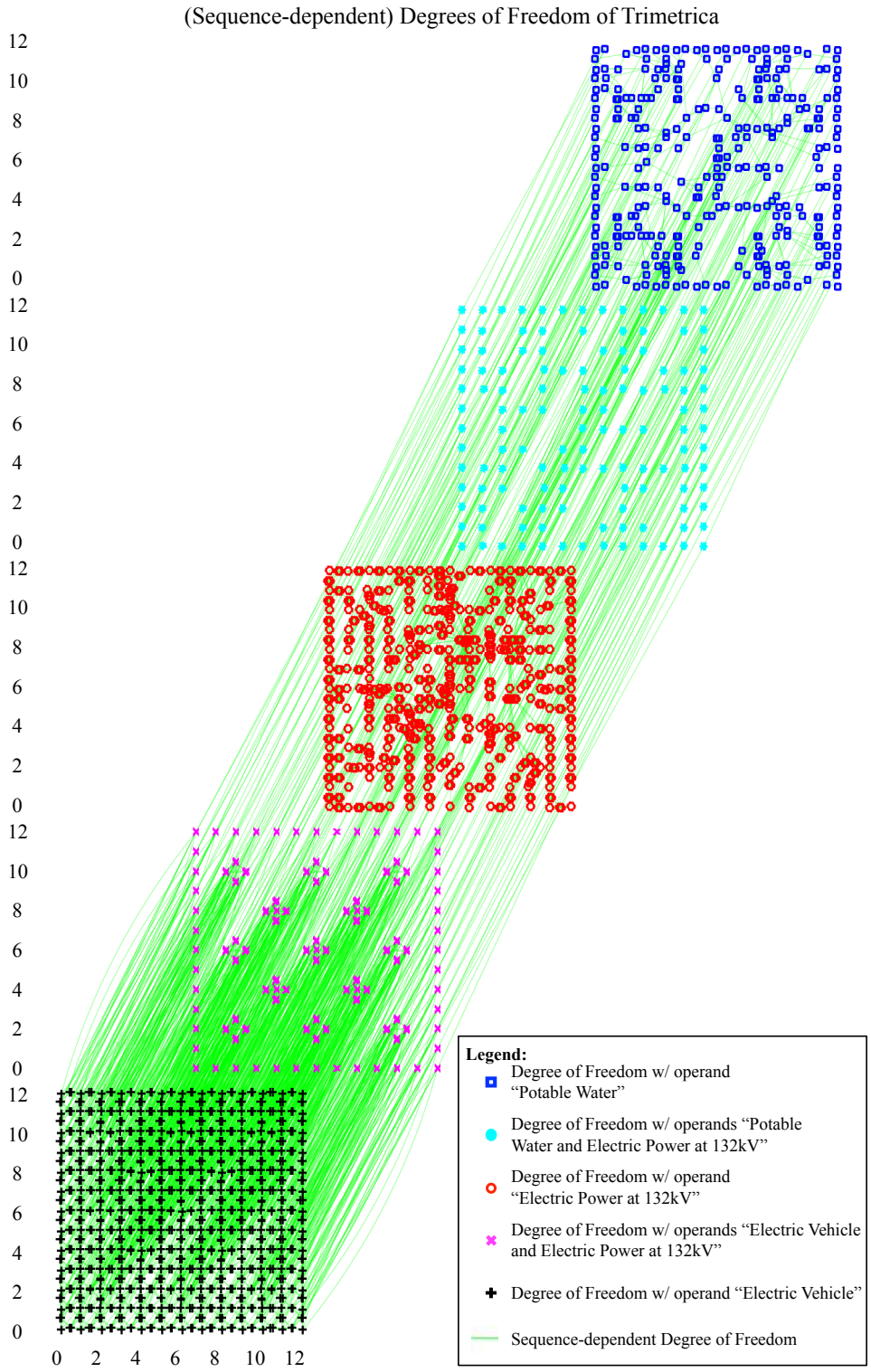


Fig. 4.23: Trimetrica's Hetero-functional Adjacency Matrix presented as a five layer network.

adjacency matrix. Then, the hetero-functional adjacency matrix for each of the infrastructure systems in Trimetrica was visualized as a means to compare the adjacency matrix to the physical topology. Lastly, all degrees of freedom were superimposed and coupled using the hetero-functional adjacency matrix. The hetero-functional adjacency matrix was used as a structural coupling of the three infrastructure layers with five different operand classes.

4.5 Controller Agency Matrix

The controller agency matrix is introduced in Section 3.3 on Page 86 as a means to differentiate between two systems of equivalent capabilities but different control structure. The Trimetrica test case contains a control structure that includes a water utility, an electric power utility, and an end user as its control agents. This example is succinct, but it highlights the importance of including the control structure when modeling interdependent smart city infrastructure systems. As a consequence of the systems' integrated nature, the legacy control structures have to cooperate. The controller agency matrix facilitates the holistic operation, because it clarifies the relation between physical resources and their controllers.

For the construction of the controller agency matrix, the section first expands the existing set of system resources to include Trimetrica's cyber-resources. Based on this expanded set, the controller agency matrix is calculated. Lastly, the section discusses the relationship between the cyber-resources and the degrees of freedom.

4.5.1 Expansion of System Resources

In Section 4.3, Trimetrica's set of system resources R_{SC} was defined with size 1,223. However, the section based itself on the definition of the physical resources R_{PSC}, without including the system cyber-resources Q_{SC}. Following the expansion of the system resources in Section 3.3 to include the cyber-resources, the set of Trimetrica's system resources is redefined to $R_{SC} = R_{PSC} \cup Q_{SC} = M_{SC} \cup B_{SC} \cup H_{SC} \cup Q_{SC}$.

Trimetrica's cyber-resources Q_{SC} are a combination of the dependent and independent cyber-resources, Q_{DSC} and Q_{ISC} respectively. For the Trimetrica test case, three types of cyber-resources have been defined:

1. Water Utility: the water utility manages the water treatment facilities and the water distribution network (water pipelines and water storage facilities).

2. Electric Power Utility: the electric power utility manages the power generation facility, the substations and the power transmission lines.

3. End Users: the end users are individuals who drive their electric vehicles on roads (with or without charging) and intersections, and consume power and water at home or in the office[8].

As a result of this definition, none of the physical resources have an internal (or dependent) cyber-resource. $Q_{DSC} = \emptyset$. The three cyber-resources are all distinct from the physical resources under their respective jurisdictions, and consequently are incorporated in the set of independent cyber-resources Q_{ISC}. The final set of Trimetrica's system resources R_{SC}, therefore, has a size of 1,223+3=1,226 as it adds three independent cyber-resources to the set defined in Section 4.3.

4.5.2 Smart City Controller Agency Matrix

The controller agency matrix is defined in Definition 3.18 on Page 87 as a binary matrix of size $\sigma(R_P) \times \sigma(R)$, whose element $A(v_1, v_2)$ is equal to one when the resource $r_{v_2} \in R$ has control jurisdiction over the physical resource $r_{v_1} \in R_P$. For Trimetrica, the resulting controller agency matrix A_{QSC} has a size of $1,223 \times 1,226$ with 2,446 filled elements. However, for modeling convenience, each physical resource is under the jurisdiction of a

[8]Note that the test case has chosen a simplified representation of the 'end users', as a single aggregated cyber-resource. One can also decide to represent each of the end users as a separate cyber-resource. However, the current representation has been chosen to reduce the complexity of the visualizations and maintain intuition.

single independent cyber-resource.

$$A_{QSC} = \left[\begin{array}{c|c} I^{\sigma(R_P)} & \overline{A}_{QSC} \end{array} \right] \tag{4.12}$$

As a result of this characteristic, the matrix can be reduced to $\overline{A}_{QSC}$ with a size of $\sigma(R_{PSC}) \times \sigma(Q_{ISC}) = 1,223 \times 3$ with 1,223 filled elements.

4.5.3 The relation between the Controller Agency Matrix and the Hetero-functional Adjacency Matrix

The previous two sections defined Trimetrica's interdependent smart city infrastructure system from a structural perspective using degrees of freedom and sequence-dependent degrees of freedom to create the Hetero-functional Adjacency Matrix. The structural degrees of freedom represent a mapping of the system processes onto the system resources. The controller agency matrix defines the control relation between the cyber-resources and the physical resources. Consequently, the controller agency matrix implies a coupling of the physical resources' degrees of freedom. The controller agency matrix can, therefore, be expanded to couple the controller agents to the hetero-functional adjacency matrix, with matrix $\widehat{A}_{QSC}$:

$$\widehat{A}_{QSC} = \overline{A}_{QSC} \otimes \mathbb{1}^{\sigma(P)} \tag{4.13}$$

where $\widehat{A}_{QSC}$ has size $\sigma(R)\sigma(P) \times \sigma(Q)$ Additionally, this matrix can be projected to create $\widetilde{A}_{QSC}$, so that controller agents are coupled to the projected set of capabilities.

$$\widetilde{A}_{QSC} = \mathbb{P}_S(\overline{A}_{QSC} \otimes \mathbb{1}^{\sigma(P)}) \tag{4.14}$$

where $\widetilde{A}_{QSC}$ has size $\mathcal{E}_S \times \sigma(Q_{SC}) = 1,991 \times 3$ with 1,991 filled elements.

Conventionally, jurisdiction has been imposed upon physical resources without consideration of their functions. For example, the water utility controls not just the water

treatment facility but also parking permits. Therefore, it determines who is allowed to park and charge at the water treatment facility's parking lot. Consequently, the system degrees of freedom associated with transportation or electric power may be uncoordinated with those infrastructures.

The controller agency matrix is represented in Figure 4.24 on Page 175 as a coupling between cyber-resources and system degrees of freedom. Combining this coupling (found in Figure 4.24 on Page 175) with the hetero-functional adjacency matrix in Figure 4.23 on Page 170 yields an integrated picture of cyber-physical couplings in Figure 4.25 on Page 176.

4.6 Controller Adjacency Matrix

Section 3.4 on Page 92 introduced the Controller Adjacency Matrix as the third of three types of interfaces identified by Hetero-functional Graph Theory. The first interface type is captured by the Hetero-functional Adjacency Matrix, and addresses the interface between physical capabilities. The second interface is captured by the Controller Agency Matrix and addresses interface between the physical resources and cyber-resources. The Controller Adjacency Matrix captures the third and final interface; and addresses the cyber-interface between the cyber-resources. The controller adjacency matrix in Definition 3.19 is a binary matrix of size $\sigma(Q) \times \sigma(Q)$. Its elements $A_C(v_1, v_2)$ are equal to one when cyber-resource $r_{v_1} \in Q$ passes information to cyber-resource $r_{v_2} \in Q$.

Trimetrica's set of cyber-resources Q_{SC} = {water utility, electric power utility, end user}, was introduced previously in Section 4.5. Consequently, the controller adjacency matrix A_{CSC} has a size of 3×3. The matrix captures all cyber-interfaces between the cyber-resources, and, therefore, all elements are connected.

- The End Users (physically) draw electric power and water from the respective systems. The physical exchange is facilitated by information about payments, location, and

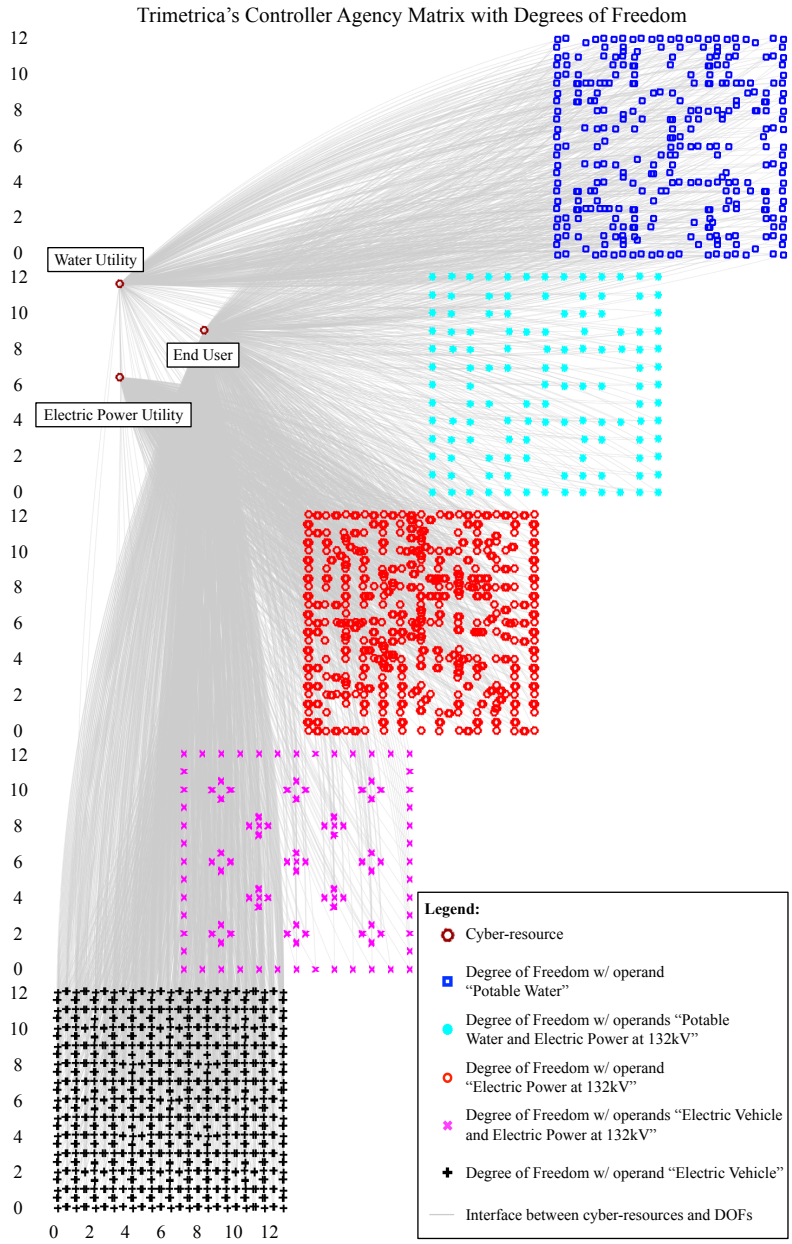


Fig. 4.24: Trimetrica's Controller Agency Matrix: It presents the control relations between the independent cyber-resources in the top-left and the degrees of freedom under their jurisdiction.

consumption.

- The Water Utility provides potable water to the end users. This physical exchange is enabled by a payment system, and water consumption measurement devices. Note that the water utility also draws electric power for its water treatment facilities. Therefore,

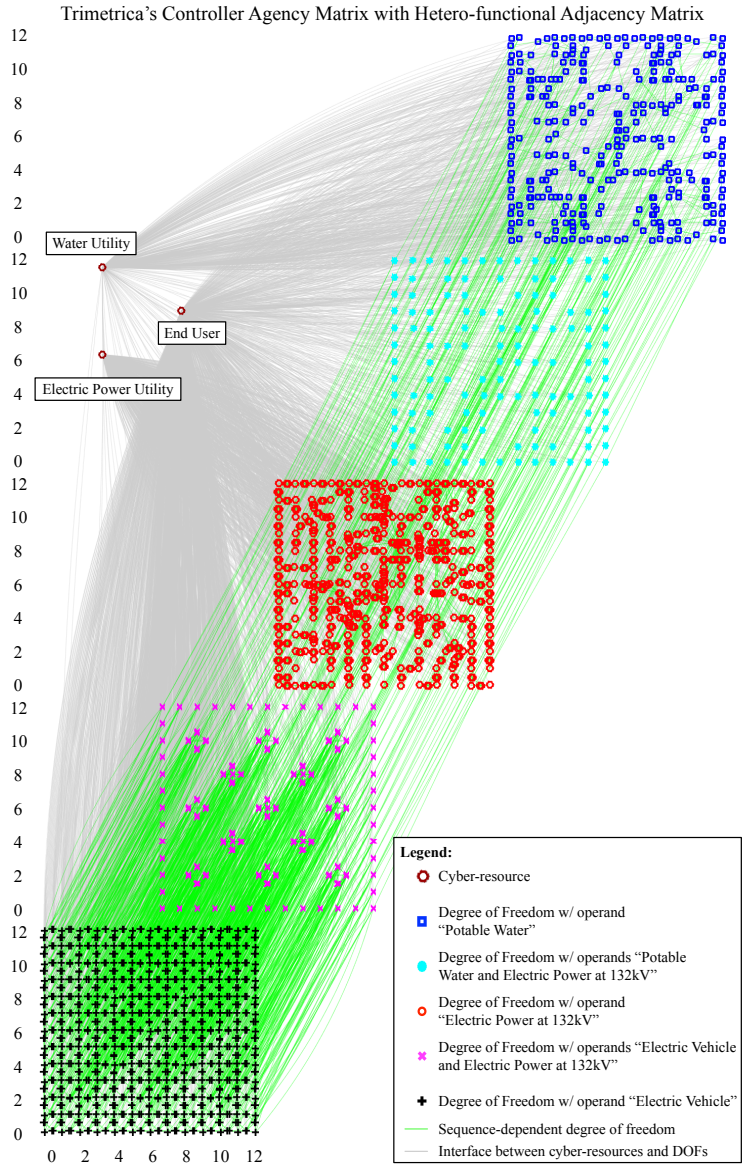


Fig. 4.25: Trimetrica's Controller Agency Matrix superimposed on the Hetero-functional Adjacency Matrix: The gray edges represent the control relations between the independent cyber-resources and the degrees of freedom under their jurisdiction. The green edges represent the sequence-dependent degrees of freedom as calculated in Section 4.4.

there is also a direct cyber-interface between the water utility and the electric power utility.

- The Electric Power Utility supplies electric power to the end users and the water utility. The physical exchange is facilitated by information about electric power consumption

patterns and a payment system.

As a result of these interactions, all 9 elements in Trimetrica's controller adjacency matrix are filled.

Figure 4.26 presents an overview of the three cyber-resources in the Trimetrica test case, with their respective cyber-interfaces. Logically, all three cyber-resources are directly connected.

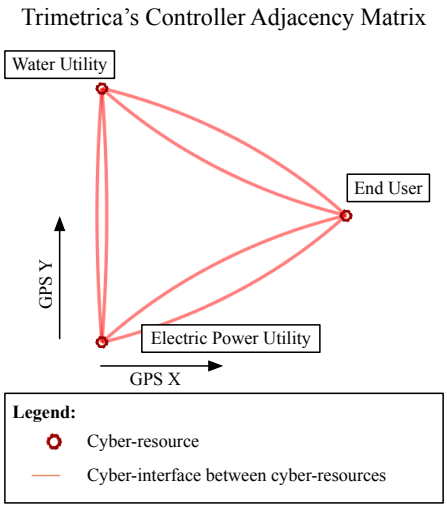


Fig. 4.26: Trimetrica's Controller Adjacency Matrix: It presents the informatic interfaces between the cyber-resources.

Figure 4.27 on Page 178 superimposes the controller adjacency matrix on Figure 4.25 as introduced in Section 4.5. The figure now shows all three types of interfaces between physical, and cyber-resources in Trimetrica.

4.7 Service as Operand Behavior

The previous sections discussed the control structure of Trimetrica's independent smart city infrastructure system. This section continues by describing the operands in Trimetrica as they move through the city's structure to deliver services. This section draws on the theory presented in Chapter 3.5 on Page 95. The section first discusses Trimetrica's service delivery

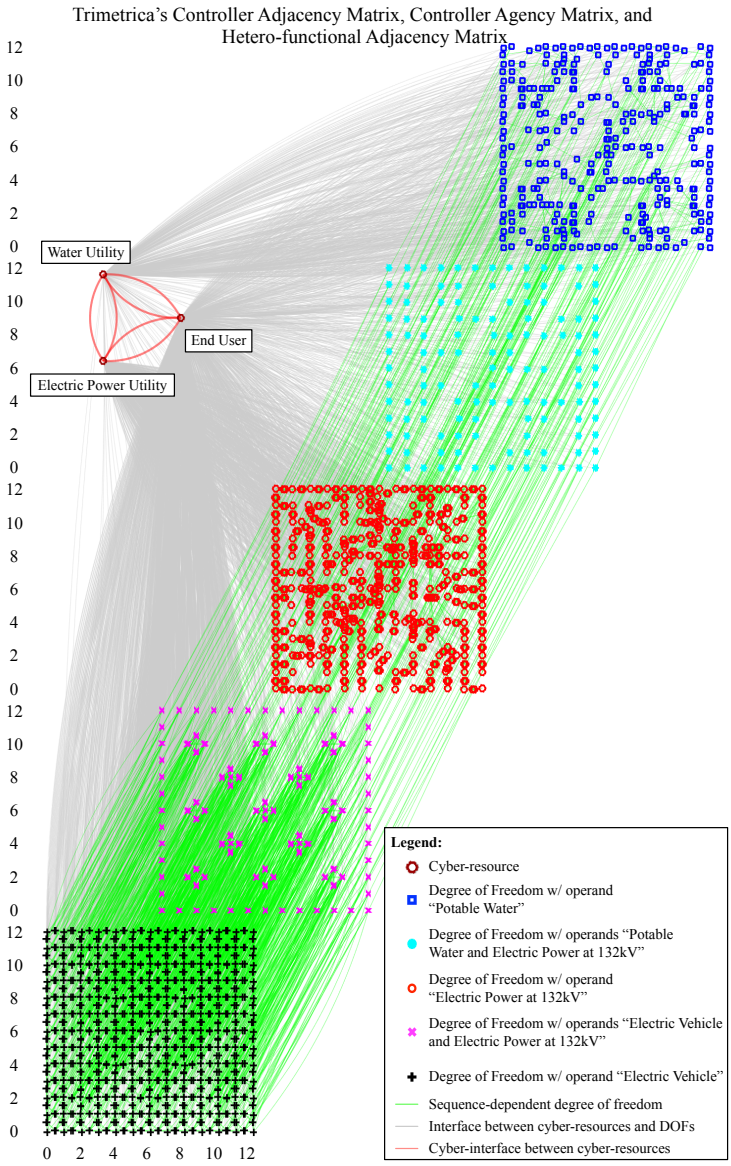


Fig. 4.27: Trimetrica's Controller Adjacency Matrix superimposed on Figure 4.25, which includes the Controller Agency Matrix and the Hetero-functional Adjacency Matrix. The red edges represent the cyber-interfaces between the cyber-resources.

in SysML. Thereafter, it derives the service nets for Trimetrica, which are then translated to a service graph. The following section couples the service graph to the hetero-functional adjacency matrix.

4.7.1 Service delivery in SysML

Trimetrica contains infrastructure to deliver three types of services in the smart city:

1. potable water for consumption of any kind,
2. electric power for consumption or work of any kind, and
3. electric transportation.

As introduced in Chapter 3.5, the operand state can be represented in SysML using State Machines.

The state machine for the first service, “deliver potable water", is displayed in Figure 4.28. The state machine contains three activities. The activity *“treat water()"* achieves two goals. First, it injects water into the potable water system, changing the state of water from “outside the potable water system" to “within the potable water system". Second, the water is transformed from its input state of surface water, to the state “is potable". The activities *“consume hot water()"* and *“consume cold water()"* oppose the first activity. They withdraw potable water from the potable water system and change its state from “potable water" to “hot waste water" or “cold waste water" respectively.

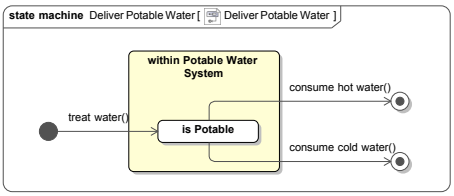


Fig. 4.28: State Machine for the Service “Deliver Potable Water" in the Trimetrica Interdependent Smart City Infrastructure System.

The state machine for the second service, “deliver electric power", is displayed in Figure 4.29. The state machine contains seven activities. The activity *“generate power()"* changes the state of the energy from outside the system to “within the electric power system" and changes the state also to “electric power at 132kV". Note that power generation would, for example, take natural gas as its input from outside the system boundary. The other six

activities all withdraw electric power from the electric power system, either by virtue of an activity in another infrastructure system, or, for example, to drive appliances at home or in the office. Naturally, the state of the electric power is changed as it is used for heat, work, or to charge an EV.

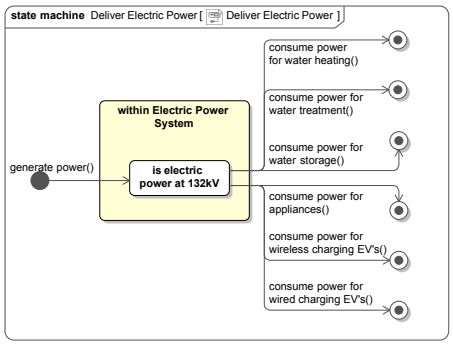


Fig. 4.29: State Machine for the Service “Deliver Electric Power" in the Trimetrica Interdependent Smart City Infrastructure System.

The last state machine for the service “Deliver Electric Vehicle" is displayed in Figure 4.30. The state machine contains three activities. Other than the previous two services, the state machine for deliver EV has two input activities. The EV is charged either wirelessly via induction charging in the electrified roads or by wire using the charging stations throughout Trimetrica. These activities take the electric power from power grid and change the state of the EV’s battery. Note that it would be more appropriate to use a continuous state for the EV because the state of its battery is not discrete. However, for simplicity of presentation the state machine representation is used. The activity *“discharge EV by driving()"* depletes the battery as the EV drives through the Trimetrica system. The work done by the vehicle is dissipated as heat and movement outside the system boundary.

4.7.2 Service delivery using Petri nets

The SysML diagrams provide detailed insight in the nature of each of the services. However, the SysML diagrams don’t lend themselves to quantitative analysis. Therefore, hetero-functional graph theory introduces Petri net-based service nets. Service nets facilitate the

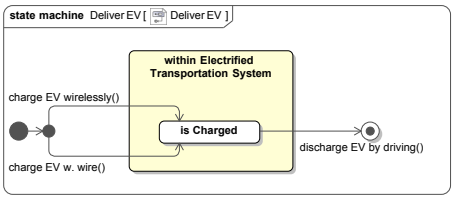


Fig. 4.30: State Machine for Service “Deliver Electric Vehicle" in the Trimetrica Interdependent Smart City Infrastructure System.

capture of the potentially complex behavior of operands quantitatively.

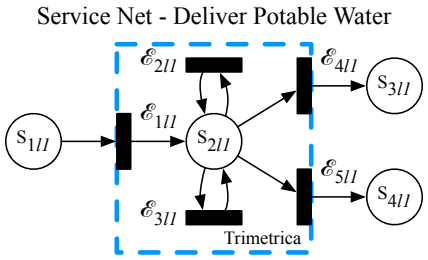


Fig. 4.31: Service net for the Service “Deliver Potable Water" in the Trimetrica Interdependent Smart City Infrastructure System.

The service net for the service “Deliver Potable Water" is presented in Figure 4.31. The figure shows four states of the operand with five transitions that evolve the state. The states of water are:

1. S_{1l_1}: Surface water
2. S_{2l_1}: Potable water
3. S_{3l_1}: Hot waste water
4. S_{4l_1}: Cold waste water

The five transitions are:

1. $\mathcal{E}_{1l_1}$: Treat water
2. $\mathcal{E}_{2l_1}$: Maintain potable water[9]

[9]The addition of a single “maintain operand state" transition for each place is absolutely necessary once holding processes of a transformative nature are added to the model. This has been discussed on Page 99.

3. $\mathcal{E}_{3l_1}$: Store potable water

4. $\mathcal{E}_{4l_1}$: Consume hot water

5. $\mathcal{E}_{5l_1}$: Consume cold water

Note that the system boundary is indicated using a blue striped box. Transitions $\mathcal{E}_{1l_1}$, $\mathcal{E}_{4l_1}$, and $\mathcal{E}_{5l_1}$ are located on the edge of the system. They import and export the operands across the system boundary. Additionally, states S_{1l_1}, S_{3l_1}, and S_{4l_1} are depicted outside the system boundary. These states provide context to the service net and support the reader's intuition.

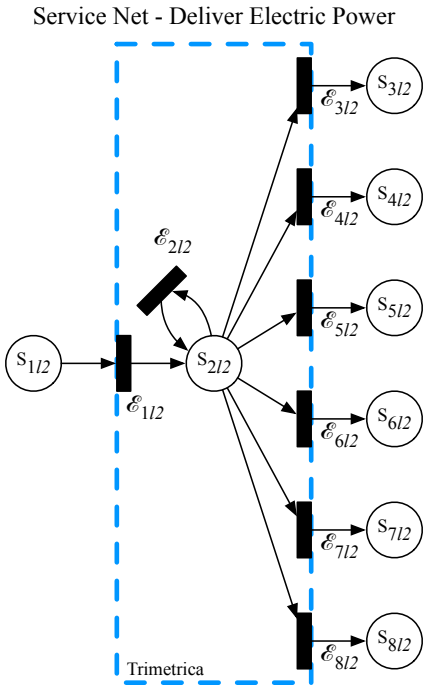


Fig. 4.32: Service Net for the Service "Deliver Electric Power" in the Trimetrica Interdependent Smart City Infrastructure System.

The service net for the service "Deliver Electric Power" is presented in Figure 4.32. The figure shows eight potential states of the operand with eight transitions that evolve the state. The states of power are:

1. S_{1l_2}: Fuel

2. S_{2l_2}: Electric power at 132kV

3. S_{3l_2}: Work for water treatment

4. S_{4l_2}: Heat for hot water consumption

5. S_{5l_2}: Work for water storage

6. S_{6l_2}: Electric power for appliances

7. S_{7l_2}: Power for wireless charging of EVs

8. S_{8l_2}: Power for wired charging of EVs

The eight transitions are:

1. $\mathcal{E}_{1l_2}$: Generate power

2. $\mathcal{E}_{2l_2}$: Maintain electric power at 132kV

3. $\mathcal{E}_{3l_2}$: Consume power for water heating

4. $\mathcal{E}_{4l_2}$: Consume power for water treatment

5. $\mathcal{E}_{5l_2}$: Consume power for water storage

6. $\mathcal{E}_{6l_2}$: Consume power for appliances

7. $\mathcal{E}_{7l_2}$: Consume power for wireless charging of EVs

8. $\mathcal{E}_{8l_2}$: Consume power for wired charging of EVs

Note that the system boundary is indicated using a blue striped box. All transitions except $\mathcal{E}_{2l_2}$ are located on the edge of the system such that they import and export the operands across the system boundary. Additionally, all states except S_{2l_2} are depicted outside the system boundary. These states provide context to the service net and support the reader's intuition.

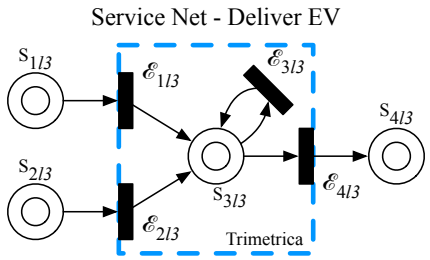


Fig. 4.33: Service Net for Service “Deliver Electric Vehicle" in the Trimetrica Interdependent Smart City Infrastructure System.

The final service net is for the service “Deliver EV" as presented in Figure 4.33. The figure shows four states and four transitions. The states represent the “state-of-charge" of the electric vehicle. Consequently, the states are the following:

1. S_{1l_3}: Electric power at wireless charger
2. S_{2l_3}: Electric power at wired charger
3. S_{3l_3}: State-of-charge electric vehicle
4. S_{4l_3}: Work as movement of electric vehicle

And the transitions are the following:

1. $\mathcal{E}_{1l_3}$: Charge EV wirelessly
2. $\mathcal{E}_{2l_3}$: Charge EV by wire
3. $\mathcal{E}_{3l_3}$: Maintain state-of-charge EV
4. $\mathcal{E}_{4l_3}$: Discharge EV by driving

Note that the states in this service net are presented differently from those presented in the previous service nets. The state S_{3l_3} represents state-of-charge in the battery of an electric vehicle as a rational number rather than as an integer. Consequently, a different type of Petri net is necessary [250]. Continuous Petri nets allow for the division of markers

into tokens to emulate a continuous behavior. It facilitates the non-discrete behavior of the electric vehicle's state of charge. The remaining states in the system are outside the system boundary and are also considered continuous for consistency.

4.7.3 Service Delivery as Service Graph

Thus far, services have been modeled as service (Petri) nets. Service activities have been represented as transitions. As discussed in Section 3.5.2, the adjacency of these transitions is calculated using a dual adjacency matrix:

$$A_{Tl_i} = M_{l_i}^{+T} M_{l_i}^{-} \tag{4.15}$$

where $M_{l_i}^{+}$ is the positive incidence matrix of the service net, $M_{l_i}^{-}$ is the negative incidence matrix of the service net, and both have a size of $\sigma(S_{l_i}) \times \sigma(\mathcal{E}_{l_i})$. The resulting matrix A_{Tl_i} has a size of $\sigma(\mathcal{E}_{l_i}) \times \sigma(\mathcal{E}_{l_i})$, and shows the feasible sequences in the service net. Visually, such a graph has the transitions as nodes and the directed arcs represent their adjacency. Each of the transition adjacency matrices is now calculated. Note that the incidence matrices only consider the places and transitions *within* the system boundary.

The service "Deliver potable water" has an incidence matrix of size $\sigma(S_{l_1}) \times \sigma(\mathcal{E}_{l_1}) = 1 \times 5$. The size of the transition adjacency matrix A_{Tl_1} is $\sigma(\mathcal{E}_{l_1}) \times \sigma(\mathcal{E}_{l_1}) = 5 \times 5$ with 12 filled elements. Figure 4.34 shows the visualization of A_{Tl_1}. When compared with the service net in Figure 4.31, the service graph closely resembles its transition structure.

The service "Deliver electric power" has an incidence matrix of size $\sigma(S_{l_2}) \times \sigma(\mathcal{E}_{l_2}) = 1 \times 8$. The size of the transition adjacency matrix A_{Tl_2} is $\sigma(\mathcal{E}_{l_2}) \times \sigma(\mathcal{E}_{l_2}) = 8 \times 8$ with 14 filled elements. Figure 4.35 shows the visualization of A_{Tl_2}. When compared with the service net in Figure 4.32, the service graph closely resembles its transition structure.

The service "Deliver electric vehicle" has an incidence matrix of size $\sigma(S_{l_3}) \times \sigma(\mathcal{E}_{l_3}) = 1 \times 4$. The size of the transition adjacency matrix A_{Tl_3} is $\sigma(\mathcal{E}_{l_3}) \times \sigma(\mathcal{E}_{l_3}) = 4 \times 4$ with six filled

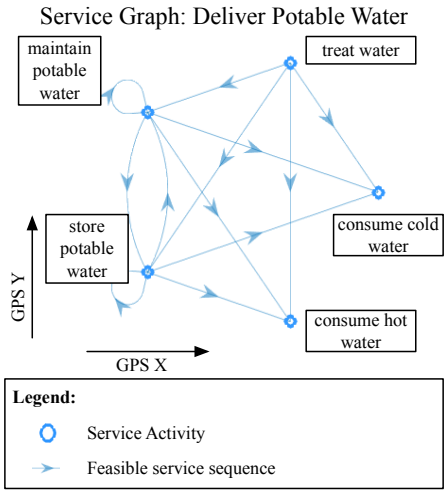


Fig. 4.34: Service Graph for the Service “Deliver Potable Water" in the Trimetrica Interdependent Smart City Infrastructure System.

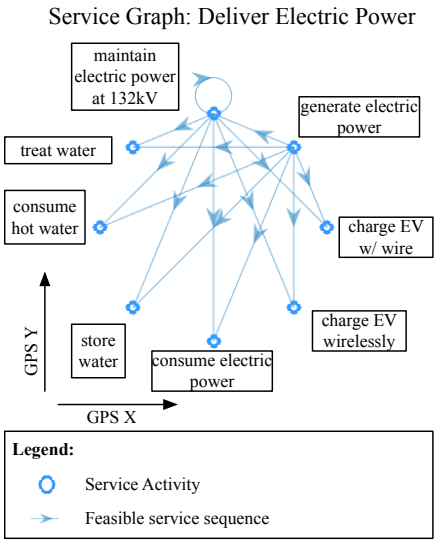


Fig. 4.35: Service Graph for the Service “Deliver Electric Power" in the Trimetrica Interdependent Smart City Infrastructure System.

elements. Figure 4.36 (on Page 187) shows the visualization of A_{Tl_3}. When compared with the service net in Figure 4.33, the service graph closely resembles its transition structure.

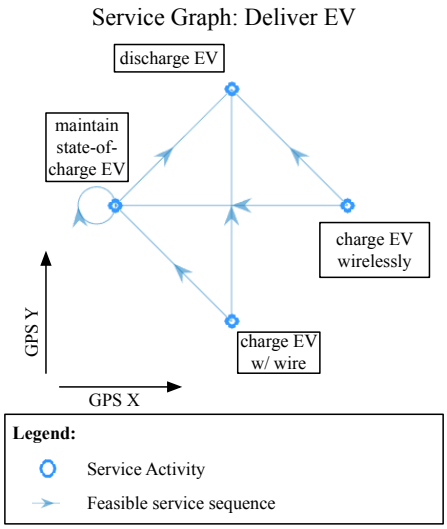


Fig. 4.36: Service Graph for the Service “Deliver Electric Vehicle" in the Trimetrica Interdependent Smart City Infrastructure system.

4.8 Service Feasibility Matrix

This section defines Trimetrica’s service feasibility matrix as the coupling of the engineering system to its services. Chapter 3.6 on Page 102 first introduced the service feasibility matrix. This section first provides the SysML description of the coupling between system structure and service delivery. Thereafter, it uses hetero-functional graph theory to describe the coupling explicitly.

SysML generally uses state machines to describe the state of operands in the system, as discussed in Section 4.7. On the other hand, the activities of the system are described in the activity diagram. In SysML, the state machines are not *explicitly* coupled to the activity diagram. The coupling of the activity diagram and the state machine is a result of both models describing the same system from different viewpoints. A triggered transformation process in the activity diagram evolves the state of the operand in the state machine. The coupling in SysML is, therefore, implicit.

Hetero-functional Graph Theory, however, describes the coupling between the service transitions and the transformation processes explicitly using the service feasibility matrix. Section 4.7 defined a separate service net for each service in Trimetrica. Consequently, each

of these services has a separate service feasibility matrix as well. The size of the service feasibility matrix is defined in Section 3.6 in Definition 3.25 on Page 104 as $\sigma(\mathcal{E}_{l_i}) \times \sigma(P)$ where $l_i \in L$ is the selected service. Its elements equal 1 if e_{xl_i} realizes system process p_w.

Thus far, the service feasibility matrix has been defined such that it couples the service activities to the system processes. However, the system structure has been defined using degrees of freedom as a mapping of system function onto system form. Consequently, by mapping the service feasibility matrix onto the system resources, the coupling allows for a direct link between the capabilities in the system and the service activities, as demonstrated in Equation 4.17. The resulting matrix $\widehat{\Lambda}_i$ has size $\sigma(\mathcal{E}_{l_i}) \times \sigma(R)\sigma(P)$. For the purpose of this work, this matrix can be projected to map the service activities to the projected set of capabilities, as follows in Equation 4.18. Matrix $\widetilde{\Lambda}_i$ has size $\sigma(\mathcal{E}_{l_i}) \times \sigma(\mathcal{E}_S)$.

$$\Lambda_{iSC} = \left[\Lambda_{\mu i} \quad | \quad \Lambda_{\gamma i} \otimes \mathbb{1}^{\sigma(P_\eta)T} \right] \tag{4.16}$$

$$\widehat{\Lambda}_{iSC} = \mathbb{1}^{\sigma(R)T} \otimes \Lambda_{iSC} \tag{4.17}$$

$$\widetilde{\Lambda}_{iSC} = \left(\mathbb{1}^{\sigma(R)T} \otimes \Lambda_{iSC} \right) \mathbb{P}_S^T \tag{4.18}$$

The section now continues to define each of the projected service feasibility matrices for the three services in Trimetrica.

The derivation of the service feasibility matrices follows four steps: first, the service transformation feasibility matrix and service transportation feasibility matrix are derived for each of the services. Second, Equation 4.16 is used to calculate each of the service feasibility matrices. Third, Equation 4.17 maps the service feasibility matrix onto the system adjacency matrix. The last step projects the service feasibility matrices to map onto the degrees of freedom using Equation 4.18. Each of these steps is performed for each of the services in Trimetrica.

4.8.1 Deliver Potable Water

The service transformation and transportation feasibility matrices for the service “deliver potable water" are derived from the mapping of service activities onto system processes. The set of service transitions for the service “deliver potable water" has a size of five and is defined as: $\mathcal{E}_{l_1}$ = {treat water, maintain potable water, store potable water, consume hot water, consume cold water}. The set of transformation processes in Trimetrica is $P_{\mu SC}$ = {treat water, generate electric power, consume hot water, consume cold water, consume electric power}. The service transformation feasibility matrix $\Lambda_{\mu l_1}$ has a size of $\sigma(\mathcal{E}_{l_1}) \times P_{\mu SC} = 5 \times 5$ and contains three filled elements. The set of holding processes in Trimetrica is $P_{\gamma SC}$ = {carry potable water, carry potable water while consuming electric power, carry electric power at 132kV, carry electric vehicle without affecting battery, carry electric vehicle while discharging, carry electric vehicle while charging by wire, carry electric vehicle while charging wirelessly}. The service transportation feasibility matrix $\Lambda_{\gamma l_1}$ has a size of $\sigma(\mathcal{E}_{l_1}) \times P_{\gamma SC} = 5 \times 7$ and contains two filled elements.

Based on the service transformation and transportation feasibility matrix, Equations 4.16, 4.17, and 4.18 calculate the (1) service feasibility matrix, (2) service feasibility matrix projected to the system adjacency matrix, and (3) service feasibility matrix projected to the degrees of freedom respectively:

1. Λ_{l_1} has a size of $\sigma(\mathcal{E}_{l_1}) \times \sigma(P_{SC}) = 5 \times 703,428$. It contains 200,981 filled elements.
2. $\widehat{\Lambda}_{1SC}$ has a size of $\sigma(\mathcal{E}_{l_1}) \times \sigma(R_{SC}) \sigma(P_{SC}) = 5 \times 860,292,444$. It contains 245,799,763 filled elements.
3. $\widetilde{\Lambda}_{1SC}$ has a size of $\sigma(\mathcal{E}_{l_1}) \times \sigma(\mathcal{E}_S) = 5 \times 1,991$. It contains 440 filled elements.

4.8.2 Deliver Electric Power

The second service is “deliver electric power." The set of service transitions has a size of eight and is defined as: $\mathcal{E}_{l_2}$ = {generate electric power, maintain potable water, treat water,

consume hot water, store water, consume electric power, charge EV wirelessly, charge EV w/ wire}. The set of transformation processes in Trimetrica is $P_{\mu SC}$ = {treat water, generate electric power, consume hot water, consume cold water, consume electric power}. The service transformation feasibility matrix $\Lambda_{\mu l_2}$ has a size of $\sigma(\mathcal{E}_{l_2}) \times P_{\mu SC} = 8 \times 5$ and contains four filled elements. The set of holding processes in Trimetrica is $P_{\gamma SC}$ = {carry potable water, carry potable water while consuming electric power, carry electric power at 132kV, transport electric vehicle without affecting battery, transport electric vehicle while discharging, carry electric vehicle while charging by wire, carry electric vehicle while charging wirelessly}. The service transportation feasibility matrix $\Lambda_{\gamma l_2}$ has a size of $\sigma(\mathcal{E}_{l_2}) \times P_{\gamma SC} = 8 \times 7$ and contains four filled elements.

Based on the service transformation and transportation feasibility matrix, Equations 4.16, 4.17, and 4.18 calculate the (1) service feasibility matrix, (2) service feasibility matrix projected to the system adjacency matrix, and (3) service feasibility matrix projected to the degrees of freedom respectively:

1. Λ_{l_2} has a size of $\sigma(\mathcal{E}_{l_2}) \times \sigma(P_{SC}) = 8 \times 703,428$. It contains 401,960 filled elements.
2. $\widehat{\Lambda}_{2SC}$ has a size of $\sigma(\mathcal{E}_{l_2}) \times \sigma(R_{SC}) \sigma(P_{SC}) = 8 \times 860,292,444$. It contains 491,597,080 filled elements.
3. $\widetilde{\Lambda}_{2SC}$ has a size of $\sigma(\mathcal{E}_{l_2}) \times \sigma(\mathcal{E}_S) = 8 \times 1,991$. It contains 883 filled elements.

4.8.3 Deliver Electric Vehicle

The third service is “deliver electric vehicle." The set of service transitions has a size of four and is defined as: $\mathcal{E}_{l_3}$ = {charge EV wirelessly, charge EV w/ wire, maintain state-of-charge EV, discharge EV}. The set of transformation processes in Trimetrica is $P_{\mu SC}$ = {treat water, generate electric power, consume hot water, consume cold water, consume electric power}. The service transformation feasibility matrix $\Lambda_{\mu l_3}$ has a size of $\sigma(\mathcal{E}_{l_3}) \times P_{\mu SC} = 4 \times 5$ and contains zero filled elements. The set of holding processes in Trimetrica is $P_{\gamma SC}$ =

{carry potable water, carry potable water while consuming electric power, carry electric power at 132kV, transport electric vehicle without affecting battery, transport electric vehicle while discharging, carry electric vehicle while charging by wire, carry electric vehicle while charging wirelessly}. The service transportation feasibility matrix $\Lambda_{\gamma l_3}$ has a size of $\sigma(\mathcal{E}_{l_3}) \times P_{\gamma SC} = 4 \times 7$ and contains four filled elements.

Based on the service transformation and transportation feasibility matrix, Equations 4.16, 4.17, and 4.18 calculate the (1) service feasibility matrix, (2) service feasibility matrix projected to the system adjacency matrix, and (3) service feasibility matrix projected to the degrees of freedom respectively:

1. Λ_{l_3} has a size of $\sigma(\mathcal{E}_{l_3}) \times \sigma(P_{SC}) = 4 \times 703,428$. It contains 401,956 filled elements.
2. $\widehat{\Lambda}_{3SC}$ has a size of $\sigma(\mathcal{E}_{l_3}) \times \sigma(R_{SC}) \sigma(P_{SC}) = 4 \times 860,292,444$. It contains 491,592,188 filled elements.
3. $\widetilde{\Lambda}_{3SC}$ has a size of $\sigma(\mathcal{E}_{l_3}) \times \sigma(\mathcal{E}_S) = 4 \times 1,991$. It contains 950 filled elements.

4.8.4 Visualizing the service feasibility matrix

As the service feasibility matrices describe the interface between the service graphs and the smart city capabilities, they can be represented as bipartite graphs. The rows of each of the service feasibility matrices represent the service transitions, and the columns represent the degrees of freedom. Since the service feasibility matrix is a bipartite graph, it can be visualized. Figure 4.37, on Page 192 couples the service graphs to the system capabilities, in a way similar to the coupling of the controller agents to the system capabilities. Note that many of the degrees of freedom realize more than one service transition. For example, the capability “treat water at water treatment facility 1" realizes the service transition “treat water" in both the “deliver potable water" and the “deliver electric power" services.

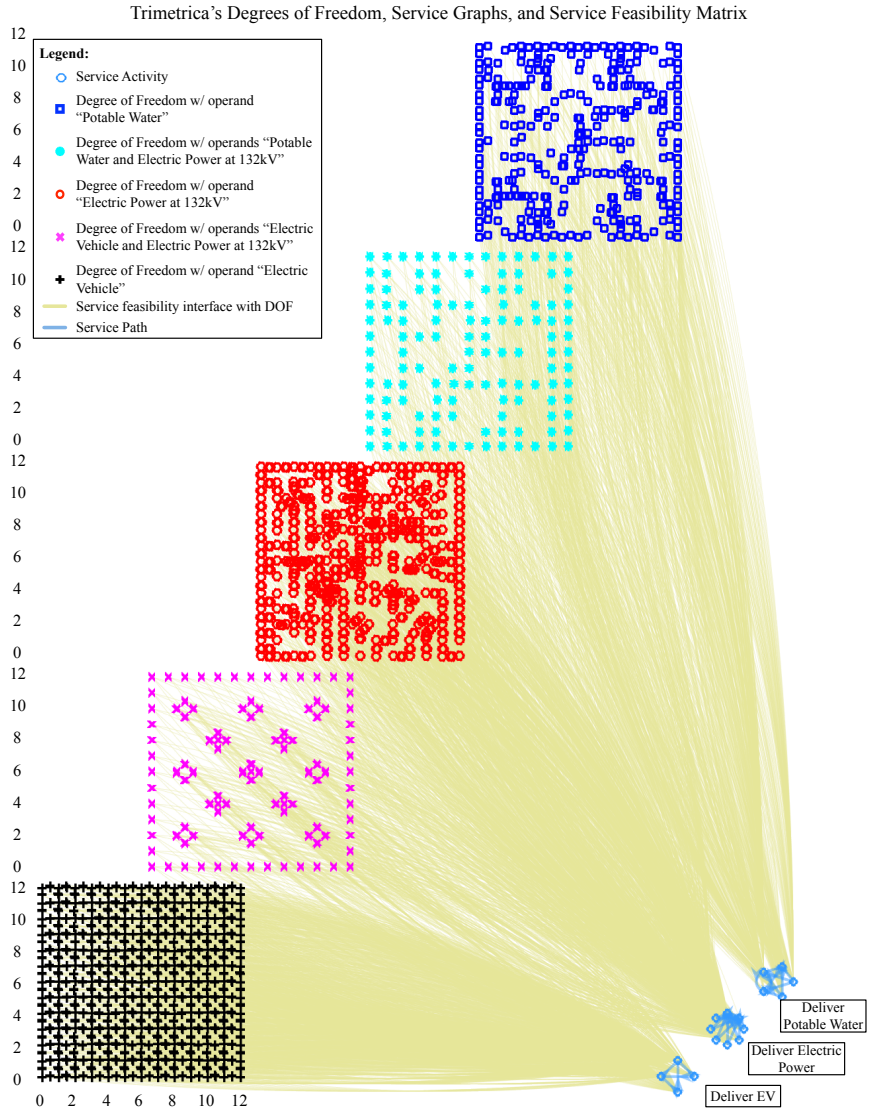


Fig. 4.37: Trimetrica's Service Graphs and the Service Feasibility Matrix: The service graphs are represented in the bottom-right corner of the figure and are drawn from Figures 4.34, 4.35, and 4.36. The service feasibility matrix is the interface shown in yellow between the service graphs and the degrees of freedom.

4.9 System Adjacency Matrix

So far, this chapter has introduced the Trimetrica test case, and described its structure, control, and services using the first six mathematical models in hetero-functional graph theory. This section integrates the six models into the system adjacency matrix for the Trimetrica case study. The system adjacency matrix was first introduced in Chapter 3.7 on

Page 112. This section first briefly revisits the elements of the system adjacency matrix, before going into detail. The second part of this section visualizes the system adjacency matrix as the hetero-functional graph for Trimetrica.

4.9.1 Trimetrica's System Adjacency Matrix

Equation 3.66 on Page 112 introduced the system adjacency matrix. For the Trimetrica test case, the system adjacency matrix $\mathbb{A}_{SC}$ is:

$$\mathbb{A}_{SC} = \begin{bmatrix} \mathbb{A}_{L} & \mathbb{A}_{L\rho} & \mathbf{0} \\ \mathbb{A}_{\rho L} & A_{\rho} & \mathbb{A}_{\rho C} \\ \mathbf{0} & \mathbb{A}_{C\rho} & A_{C} \end{bmatrix} \tag{4.19}$$

The system adjacency matrix describes Trimetrica's capabilities, control model, service model. Each of the blocks on the diagonal describes the core of those three pillars. The capabilities are described by the hetero-functional adjacency matrix A_{ρ}, the control model is described by the controller adjacency matrix A_{C}, and the operand behavior is described by $\mathbb{A}_{L}$. The control model is then coupled to the structural model by virtue of the controller agency matrix, as captured by $\mathbb{A}_{\rho C}$ and $\mathbb{A}_{C\rho}$. The service model is coupled to the structural model by virtue of the service feasibility matrix $\mathbb{A}_{\rho L}$ and its transpose. Equation 4.20 calculates the projected system adjacency matrix for the Trimetrica test case. This section continues to use the projected system adjacency matrix for its calculations.

$$\widetilde{\mathbb{A}}_{SC} = \begin{bmatrix} \mathbb{A}_{L} & \widetilde{\mathbb{A}}_{L\rho} & \mathbf{0} \\ \widetilde{\mathbb{A}}_{\rho L} & \widetilde{A}_{\rho SC} & \widetilde{\mathbb{A}}_{\rho C} \\ \mathbf{0} & \widetilde{\mathbb{A}}_{C\rho} & A_{CSC} \end{bmatrix} \tag{4.20}$$

Block $\widetilde{A}_{\rho SC}$: The hetero-functional adjacency matrix is the core of the system adjacency matrix, because it contains all system capabilities and their feasible sequences. It is defined in Section 4.4 on page 161. As mentioned previously, because the system adjacency matrix is projected, it dictates the sizes of the coupling matrices. The size of $\widetilde{A}_{\rho}$ is $\mathcal{E}_S \times \mathcal{E}_S =$

$1,991 \times 1,991$, and the number of filled elements in $\widetilde{A}_{\rho}$ is the number of sequence dependent constraints: 8,274.

Block A_{CSC}: The controller adjacency matrix describes the cyber-interfaces between the cyber-resources in Trimetrica. It is calculated in Section 4.6 on page 174. The controller adjacency matrix has size $\sigma(Q_{SC}) \times \sigma(Q_{SC}) = 3 \times 3$. The number of filled elements is 9.

Block $\mathbb{A}_L$: The upper left block of the system adjacency matrix describes the service model. It takes the service activities as nodes and shows the logical coupling between them. Equation 3.68 on Page 114 introduces the block diagonal form of the matrix so that each service behavior is uncoupled for the next. For the Trimetrica test case, matrix $\mathbb{A}_L$ is:

$$\mathbb{A}_L = \begin{bmatrix} A_{Tl_1} & 0 & 0 \\ 0 & A_{Tl_2} & 0 \\ 0 & 0 & A_{Tl_3} \end{bmatrix} \tag{4.21}$$

The elements A_{Tl_i} are calculated in Section 4.7.3 on Page 185. The size of $\mathbb{A}_L$ is $\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) \times \sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) = 17 \times 17$. The number of filled elements is 32.

Block $\widetilde{\mathbb{A}}_{\rho C}$ and $\widetilde{\mathbb{A}}_{C\rho}$: These two block matrices couple the structural degrees of freedom in $A_{\rho SC}$ and the independent cyber-resources Q_{SC}. Equation 4.14 on Page 173 calculates the projected controller agency matrix for the Trimetrica test case:

$$\widetilde{\mathbb{A}}_{\rho C} = \widetilde{A}_{QSC} = \mathbb{P}(\overline{A}_{QSC} \otimes \mathbb{1}^{\sigma(P)}) \tag{4.22}$$

where $\widetilde{A}_{\rho C}$ has a size of $\mathcal{E}_S \times \sigma(Q_{SC}) = 1,991 \times 3$. The matrix contains 1,991 filled elements. The Trimetrica test case does not differentiate between sensing and actuation signals, and, therefore, matrix $\widetilde{\mathbb{A}}_{C\rho}$ is:

$$\widetilde{\mathbb{A}}_{C\rho} = \widetilde{\mathbb{A}}_{\rho C}^{T} \tag{4.23}$$

where $\widetilde{\mathbb{A}}_{C\rho}$ has size $\sigma(Q_{SC}) \times \mathcal{E}_S = 3 \times 1,991$ with 1,991 filled elements.

Block $\mathbb{A}_{L\rho}$ and $\mathbb{A}_{\rho L}$: These matrices capture the coupling of Trimetrica's service nets

with the system structure. In Chapter 4.8, the service feasibility matrices were separately introduced for each service. However, the system adjacency matrix includes the service nets as a combination of each of the separate service graphs, which requires that the service feasibility matrices are combined as well. Using Equation 3.72 on Page 115, the concatenation of the service feasibility matrices follows:

$$\mathbb{A}_{L\rho} = (\mathbb{A}_{\rho L})^T = \begin{bmatrix} \widehat{\Lambda}_{1SC} \\ \widehat{\Lambda}_{2SC} \\ \widehat{\Lambda}_{3SC} \end{bmatrix} \tag{4.24}$$

$$\widetilde{\mathbb{A}}_{L\rho} = (\widetilde{\mathbb{A}}_{\rho L})^T = \begin{bmatrix} \widetilde{\Lambda}_{1SC} \\ \widetilde{\Lambda}_{2SC} \\ \widetilde{\Lambda}_{3SC} \end{bmatrix} \tag{4.25}$$

where $\widetilde{\mathbb{A}}_{L\rho}$ has a size of $\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) \times \mathcal{E}_S = 17 \times 1,991$ with 440 + 883 + 950 = 2,273 filled elements.

In conclusion, after the discussion of each of the seven block matrices within the system adjacency matrix, the total number of filled elements is calculated to be: $8,274 + 9 + 32 + 1,991 + 1,991 + 2,273 + 2,273 = 16,843$. The size of the system adjacency matrix $\mathbb{A}_{SC}$ is:

$$\left[\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) + \sigma(R_P)\sigma(P) + \sigma(Q)\right] \times \left[\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) + \sigma(R_P)\sigma(P) + \sigma(Q)\right] = 860,292,464 \times 860,292,464 \tag{4.26}$$

And the size of the projected system adjacency matrix $\widetilde{\mathbb{A}}_{SC}$ is:

$$\left[\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) + \sigma(\mathcal{E}_S) + \sigma(Q)\right] \times \left[\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) + \sigma(\mathcal{E}_S) + \sigma(Q)\right] = 2,011 \times 2,011 \tag{4.27}$$

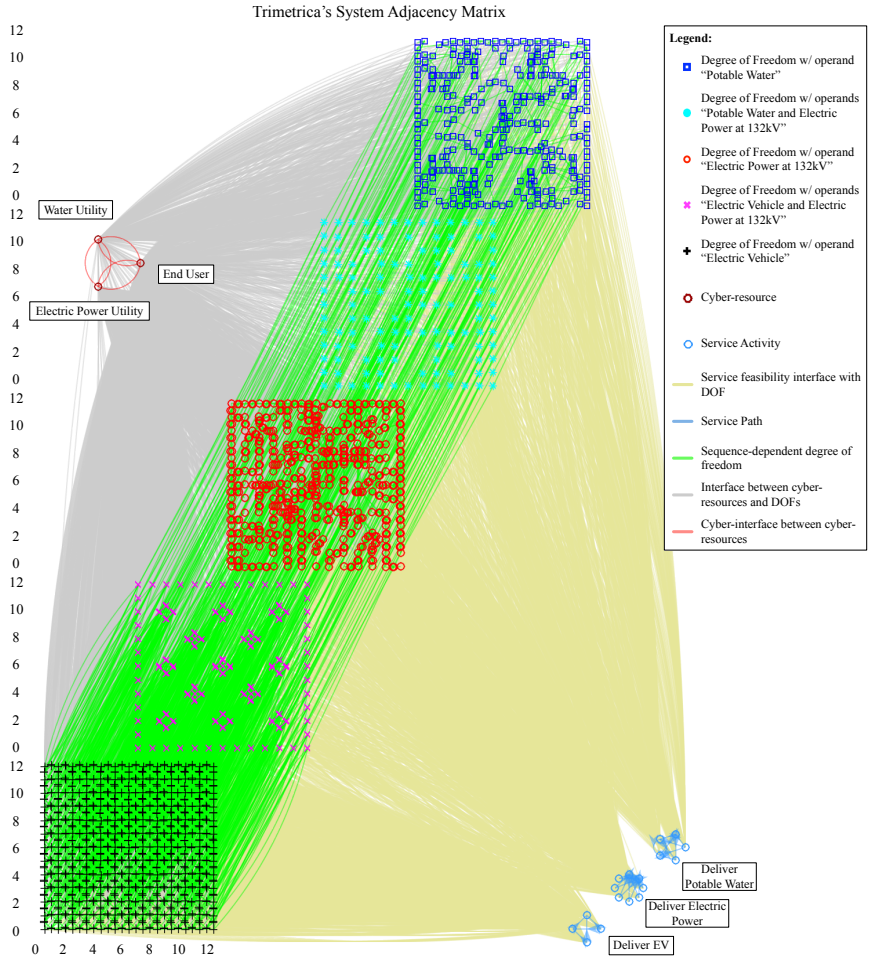


Fig. 4.38: The System Adjacency Matrix for the Trimetrica Interdependent Smart City Infrastructure System Presented as a Hetero-functional Graph.

4.9.2 Hetero-functional Graph Visualization

The system adjacency matrix describes the full Trimetrica interdependent smart city infrastructure test case in a single adjacency matrix. Consequently, the matrix can be represented as a graph. Intuitively, this chapter has presented pieces of the system adjacency matrix as each of the couplings were made. Figure 4.23 presented the hetero-functional adjacency matrix as a five layer network, with degrees of freedom as its nodes, and sequence-dependent degrees of freedom as the edges. Thereafter, the controller agency matrix coupled the cyber-resources to the degrees of freedom graphically in Figure 4.27. After the control structure, the operand behavior was defined and coupled to the hetero-functional adjacency

matrix as depicted in Figure 4.37. Finally, these different visualizations are pieced together in Figure 4.38, which presents the hetero-functional graph, a visual representation of the system adjacency matrix.

4.10 Discussion

This chapter has demonstrated the theoretical discussion of hetero-functional graph theory on an illustrative example. It has shown that an interdependent smart city infrastructure system with an arbitrary topology can be modeled using hetero-functional graph theory. First, the chapter provided a background of the development of a test case in Section 4.1. The primary advantage of test cases is that they facilitates the standardized comparison of different modeling and analysis methods. In Section 4.2, the Trimetrica test case was introduced. It delivers three services to the residents of the city by virtue of an interdependent physical infrastructure and a control structure.

After the introduction of the Trimetrica test case, Section 4.3 calculated the system knowledge base as the first model of hetero-functional graph theory so as to identify the capabilities in the smart city. These capabilities were consequently coupled sequentially in the hetero-functional adjacency matrix so as to identify the sequence-dependent degrees of freedom in Section 4.4. In Section 4.5, the cyber-resources that control Trimetrica's physical resources were related to the system's capabilities within a controller agency matrix. These cyber-resources also interface with one another. Therefore, Section 4.6 introduced the controller adjacency matrix as a type of social network. The delivered services are captured using the service nets in Section 4.7, which were consequently coupled to the system capabilities using the service feasibility matrices in Section 4.8. Finally, the System Adjacency Matrix succeeded to integrate all of the independent mathematical models, and generate a cyber-physical system adjacency matrix in Figure 4.38. The overview of the respective matrices with their respective sizes and filled elements is displayed in Table 4.2

on page 199.

This chapter has, therefore, demonstrated hetero-functional graph theory as a means of representing a smart city with three interdependent infrastructures. The work also shows that the incremental extension from two to three infrastructures is of permutation complexity. Whereas as two infrastructures interact in two ways, three infrastructures can interact in up to six ways. The Trimetrica test case, for example, required five operand layers to fully capture the interactions between infrastructures. In this sense, the work emphasizes the ability of hetero-functional graph theory to handle an arbitrary number of discipline-specific infrastructure. Finally, the broad versatility of hetero-functional graph theory is supported by its ontological foundations.

4.10.1 Ontological Analysis of Hetero-functional Graph Theory

The foundation of hetero-functional graph theory is specifically built on the ontological properties of soundness, completeness, lucidity, and laconicity, introduced in Section 2.3.1 on Page 43. Recall that the traditional application of graph theory violates the properties of completeness and lucidity as it (1) fails to represent the complete set of concepts in the domain abstraction with modeling primitives, and (2) overloads modeling primitives with multiple domain concepts.

Hetero-functional graph theory maintains the four ontological properties to ensure an isomorphic representation of the conceptual abstraction. The diverse nature of large flexible engineering systems require a diverse language, and the seven models in hetero-functional graph theory provide the necessary breadth. All Large Flexible Engineering Systems (LFESs) consist of a structural model, a controller model, and an operand behavior model. The structure of an LFES is modeled using capabilities, as processes mapped onto resources, and the hetero-functional adjacency matrix, as a definition of system sequence. The sets of processes and resources are defined as mutually exclusive and collectively exhaustive, which ensures an isomorphic representation of the physical structure.

Table. 4.2: An Overview of Trimetrica's Seven Mathematical Models of Hetero-functional Graph Theory

Model and Section	Matrix	Size	Filled Elements
Degrees of Freedom / Capabilities Section 4.3 on page 131	A_{SSC}	$\sigma(P) \times \sigma(R_P) = 703,428 \times 1,223$	Degrees of freedom: 1,991
Hetero-functional Adjacency Matrix Section 4.4 on page 161	$A_{\rho SC}$	$\sigma(R_P)\sigma(P) \times \sigma(R_P)\sigma(P) = 860,292,444 \times 860,292,444$	Sequence-dependent degrees of freedom: 8,274
	$\widetilde{A}_{\rho SC}$	$\sigma(\mathcal{E}_S) \times \sigma(\mathcal{E}_S) = 1,991 \times 1,991$	Sequence-dependent degrees of freedom: 8,274
Controller Agency Matrix Section 4.5 on page 171	A_{QSC}	$\sigma(R_P) \times \sigma(R) = 1,223 \times 1,226$	Cyber-physical interfaces: 2,446
	$\overline{A}_{QSC}$	$\sigma(R_P) \times \sigma(Q) = 1,223 \times 3$	Cyber-physical interfaces: 1,223
	$\widetilde{A}_{QSC}$	$\sigma(\mathcal{E}_S) \times \sigma(Q) = 1,991 \times 3$	Cyber-physical interfaces: 1,991
Controller Adjacency Matrix Section 4.6 on page 174	A_{CSC}	$\sigma(Q) \times \sigma(Q) = 3 \times 3$	Cyber-interfaces: 9
Service as Operand Behavior Section 4.7 on page 177	A_{Tl_1}	$\sigma(\mathcal{E}_{l_1}) \times \sigma(\mathcal{E}_{l_1}) = 5 \times 5$	Service sequences: 12
	A_{Tl_2}	$\sigma(\mathcal{E}_{l_2}) \times \sigma(\mathcal{E}_{l_2}) = 8 \times 8$	14
	A_{Tl_3}	$\sigma(\mathcal{E}_{l_3}) \times \sigma(\mathcal{E}_{l_3}) = 4 \times 4$	6
	$\mathbb{A}_L$	$\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) \times \sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) = 17 \times 17$	Service sequences: 32
Service Feasibility Matrix Section 4.8 on page 187	Λ_{l_i}	$\sigma(\mathcal{E}_{l_1}) \times \sigma(P) = 5 \times 703,428$	Number of filled elements: 440
		$\sigma(\mathcal{E}_{l_2}) \times \sigma(P) = 8 \times 703,428$	883
		$\sigma(\mathcal{E}_{l_3}) \times \sigma(P) = 4 \times 703,428$	950
	$\mathbb{A}_{L\rho}$	$\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) \times \sigma(\mathcal{E}_S) = 17 \times 1,991$	Number of filled elements: 2,273
System Adjacency Matrix Section 4.9 on page 192	$\mathbb{A}_{SC}$	$\left[\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) + \sigma(R_P)\sigma(P) + \sigma(R)\right] \times \left[\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) + \sigma(R_P)\sigma(P) + \sigma(R)\right]$	System Capabilities: 16,843
	$\widetilde{\mathbb{A}}_{SC}$	$\left[\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) + \sigma(\mathcal{E}_S) + \sigma(Q)\right] \times \left[\sum_{i=1}^{3} \sigma(\mathcal{E}_{l_i}) + \sigma(\mathcal{E}_S) + \sigma(Q)\right]$	System Capabilities: 16,843

The controller model consists of the controller adjacency matrix, as a social network, and a controller agency matrix, which defines the cyber-physical interfaces with the physical structure. The set of cyber-resources is a subset of the system resources, and is therefore also mutually exclusive and collectively exhaustive within the system boundary. Additionally, the set of processes includes the decision algorithms P_Q. Consequently, both the cyber-resources and the cyber-processes are mutually exclusive, ensuring an isomorphic representation of the control structure.

Finally, the operand behavior describes the services of the system. In this chapter, both discrete and continuous states have been modeled with the service model. Consequently, the service model facilitates the representation of a wide variety of services, which has been demonstrated in the literature. The Preface provides a brief discussion of the wide variety of application areas in which hetero-functional graph theory was applied. For an in-depth discussion, the reader is referred to Chapter 6 in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure Systems"* [5].

Throughout Chapters 3 and 4, the work assesses the need for formal constructs to completely represent the conceptualization of the system. Additionally, it assesses if the mapping between the formal constructs and the system conceptualization is one-to-one. These assessments emphasize the ontological foundations of hetero-functional graph theory.

4.10.2 Comparison with Multi-layer Networks

Beyond the ontological strengths of hetero-functional graph theory, its application to interdependent smart city infrastructures fulfills a pressing theoretical need. The theoretical versatility required to model smart city infrastructure systems quantitatively has not been realized previously. The multi-layer networks literature imposes constraints on the systems they model, as discussed by the extensive review of Kivelä et al. [95]. Naturally, such constraints, as discussed in Section 2.2.2 on Page 37, limit the ability of multi-layer networks to model an arbitrary number of arbitrarily coupled smart city infrastructures.

Hetero-functional graph theory does not impose these constraints. For each of the eight constraints identified by Kivelä et al., an example in the Trimetrica test case is provided to show that hetero-functional graph theory is not similarly limited. To facilitate the discussion, the three infrastructure systems in Trimetrica are considered to be the three layers of the multi-layer network.

In *Constraint 1*, some “multi-layer networks" require all layers to have vertically aligned nodes. Hetero-functional graph theory, however, does not impose such a requirement. For example, Trimetrica contains substations that only appear in the electric power system layer. Even if resources are geographically in the same location, they are not required to be connected. For example, Resource 190 and Resource 269 represent a substation and an intersection respectively. These have the same coordinates, but are not connected.

In *Constraint 2*, some “multi-layer networks" require disjoint layers. Hetero-functional graph theory, however, does not impose such a requirement. Trimetrica has resources that are part of more than a single infrastructure system. For example, houses with EV chargers appear in the water distribution system, electric power system, and the electrified transportation system.

In *Constraint 3*, some “multi-layer networks" require the same number of nodes in each layer. Hetero-functional graph theory, however, allows for an arbitrary number of nodes in each layer. For example, in the Trimetrica test case, the electric power system has 201 nodes, whereas the electrified transportation system has 169 nodes.

In *Constraint 4*, some “multi-layer networks" require exclusively “vertical"[10] inter-layer couplings. Hetero-functional graph theory, however, allows for arbitrary couplings between

[10]Kivelä et al. [95] refer to this constraint as “diagonal" couplings. This work adopts the term vertical to more closely reflect the depiction in Figure 2.4.

layers. For example, resource 6 is a house with an EV charger. This house consumes hot water which draws power that arrives by power line. Consequently, there is a coupling between the power line's transport power capability and the house's hot water consumption capability, which is not a vertical coupling.

In *Constraint 5*, some "multi-layer networks" require that all nodes in a given layer have identical couplings to nodes in another layer. Hetero-functional graph theory, however, is able to represent multiple distinct couplings to nodes in another layer. For example, in Trimetrica, Resource 1 couples the electric power system and the water distribution system as it is a water treatment facility with an EV charger that consumes electric power by virtue of its "treat water" process. However, Resource 6 also couples the electric power system and the water distribution system as it is a house with an EV charger that consumes electric power by virtue of its "consume hot water" process. The nature of these couplings is fundamentally different, as the processes are distinct.

In *Constraint 6*, some "multi-layer networks" require that each node is connected to all of its counterparts in other layers. Hetero-functional graph theory, however, does not require that each node is connected to all of its counterparts. For example, in Trimetrica, Resource 6 is defined as a House with an EV charger. The house has multiple, unrelated capabilities, such as "park EV at House 6" and "consume cold water at House 6". These capabilities are not sequentially coupled, as no sequence has been defined that couples the two capabilities (as follows from Figure 4.11 on Page 145). Consequently, even if the resource is part of multiple layers, the components of that resource are not necessarily coupled.

In *Constraint 7*, some "multi-layer networks" limit the number of layers to two. Hetero-functional graph theory, however, facilitates the modeling of an arbitrary number of layers. For example, the Trimetrica test case consists of three infrastructure systems.

In *Constraint 8*, some “multi-layer networks" require that each layer have no more than one aspect. Hetero-functional graph theory, however, allows for an arbitrary number of aspects in a layer. For example, the electrified transportation system has both conventional roads and electrified roads. The electrified roads have fundamentally different characteristics than the conventional road, but they are both represented within the transportation system.

4.11 Four-Layer Test Case

Section Abstract:

This section demonstrates the four-layer test case that was introduced in Chapter 2. This section has been directly adopted from Appendix A, called *“Representing a Four-Layer Network in Hetero-functional Graph Theory"*, in the book *“A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5]. ■

Smart cities create the need for greater integration between multiple infrastructures. Here, it is important to note that the coupling between these multiple infrastructures may take on a wide variety of potential topologies. In order to facilitate the diversity of interdependent smart city infrastructure, a modeling language with great mathematical versatility is needed. To that end, this book develops hetero-functional graph theory. Section 2.2.2 introduces a four-layer network that violates the limitations imposed by the existing multilayer network literature. This appendix discusses the four-layer network in more detail and presents the derivation of its hetero-functional graph representation. The goal of this appendix is to demonstrate that hetero-functional graph theory is able to represent a system that cannot be modeled with other multilayer network methods. Furthermore, the model is ontologically sound, complete, lucid, and laconic. Such a model is achieved by following the theory of Chapter 2.2.2 closely. However, for a more detailed demonstration of hetero-functional

graph theory, the reader is referred to the Trimetrica test case (Sections 4.2 through 4.10 in Chapter 4). This section proceeds as follows: first, it introduces the four-layer network. Section 4.11.1 derives the system concept. Section 4.11.2 calculates the hetero-functional adjacency matrix. Section 4.11.3 calculates the controller agency matrix. Section 4.11.4 derives the controller adjacency matrix. Section 4.11.5 derives the service model for the four-layer network. Section 4.11.6 calculates the service feasibility matrices. Finally, Section 4.11.7 integrates the 6 hetero-functional graph theory models to construct the system adjacency matrix.

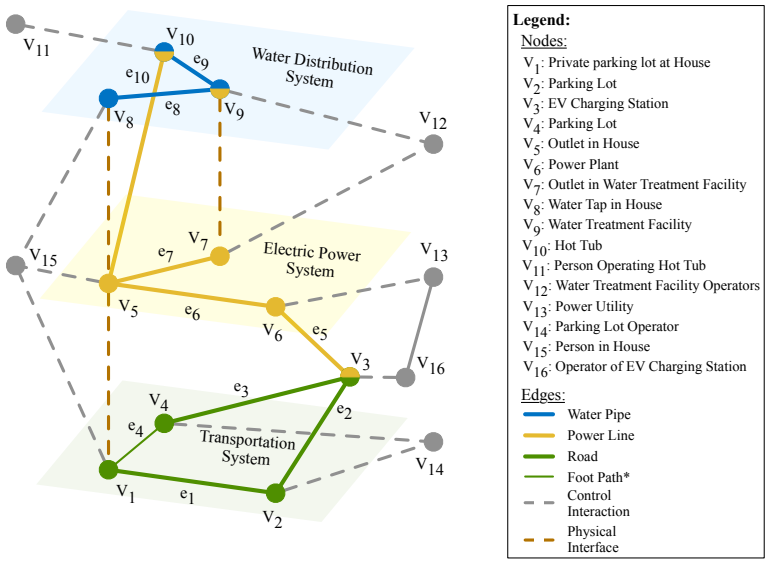


Fig. 4.39: A Hypothetical Four Layer Network: It represents transportation, electric power, and water distribution infrastructure with a super-imposed cyber-control layer. *: The foot path is part of the Transportation System, but differs in modality from the other edges in the system and is represented with a thinner edge. Its two-dimensional representation is presented in Figure 4.40.

Figure 4.39 presents the four-layer example network as used in Section 2.2.2. The *water distribution network* consists of three nodes and two edges. The water is supplied by the water treatment facility in Node v_9. This water is consumed as cold potable water in the house (Node v_8), and as hot water for the hot tub (Node v_{10}). The hot tub and the water treatment facility are both connected to the electric power system, because their processes require a supply of electric power.

The *electric power system* consists of five nodes and four edges. The power plant in node v_6 supplies electric power for the electric power grid. The hot tub in the water distribution system (node v_{10}) is also part of the electric power system, because it is receives electric power from power line e_{10}. The water treatment facility in v_9 is physically connected with electric power node v_7. One could imagine a large site with an electric power supply at the location of the offices, but the power for the water treatment process is supplied by cables on-site, between the office and the water treatment process. Node v_5 represents an outlet in the house to which, for example, an appliance is connected. The last node is the electric vehicle charging station in node v_3. The electric vehicle charging station is naturally also part of the transportation system infrastructure.

The *transportation system* consists of four nodes and four edges. As mentioned, node v_3 is an electric charging station that facilitates electric vehicle charging. Nodes v_2 and v_4 are parking lots that facilitate the storage of vehicles. Lastly, Node v_1 is a private parking location at the house. Note that edge e_4 differs from the regular roads; it is a footpath. The transportation system thus contains two modes of transportation. As a result of the two modalities in the transportation system, the end users of the system are able to switch between these modalities. Consequently, the nodes v_1 and v_4 facilitate the transition from travelling on foot to travelling by EV and back.

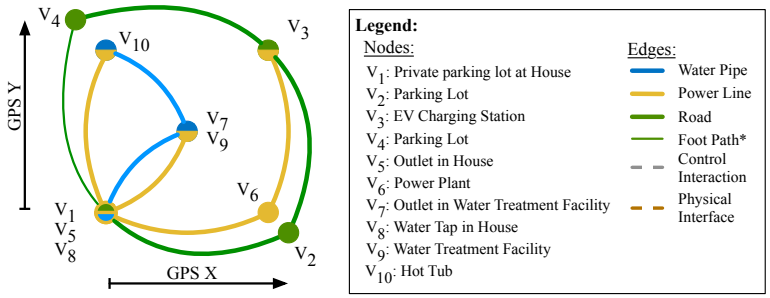


Fig. 4.40: 2D Presentation of a Hypothetical Four Layer Network: It represents transportation, electric power, and water distribution infrastructure from Figure 4.39.

The *control structure* of the four-layer network includes an end user in v_{15} that controls the nodes related to the house. Additionally, node v_{11} is a separate end user who uses the hot

tub. The water treatment facility is managed by the water utility, and similarly, the electric power utility manages the power plant. Note that the electric power utility works with the operator of the electric vehicle charging station to coordinate charging loads on the power grid. Finally, node v_{14} represents the operator of the two parking lots.

Figure 4.40 provides a two dimensional representation of Figure 4.39, without the control structure. The figure superimposes nodes that have physical couplings, such as the house (Nodes v_1, v_5, and v_8), and the water treatment facility (Nodes v_7, and v_9). Other nodes, such as the hot tub (Node v_{10}) and parking lot 2 (Node v_4), are not superimposed to emphasize that they are distinct. This new representation emphasizes the spatial topology of the infrastructure network and lends itself for a clearer comparison later in the appendix.

4.11.1 System Concept

The first model in hetero-functional graph theory is the system concept. The system concept maps system function onto system form. This section first discusses the system form of the four-layer network and then continues the definition of its system function. Finally, the mapping of the processes onto the resources is performed with the knowledge base. The resource architecture adheres to the LFES architecture as described in Chapter 3. Figure 4.41 presents the resource architecture for the four-layer network. The system classifies 11 resources in nine interface classes. Note that parking lot 1 and parking lot 2 have different connections. In contrast to parking lot 1, parking lot 2 connects to a footpath and a road and should facilitate modality change of passengers between EV and traveling on foot. The set of resources in the four-layer network is: $R = M \cup B \cup H$, of size: $\sigma(R) = \sigma(M) + \sigma(B) + \sigma(H) = 6 + 1 + 10 = 17$.

The four-layer network activity diagram, as presented in Figure 4.42, is similar to the activity diagram for the Trimetrica test case (Figure 4.11 on Page 145). The most notable difference is the addition of the modality “carry pedestrian". In order to facilitate sequences with a modality change, two transformation processes are added that allow for entering and

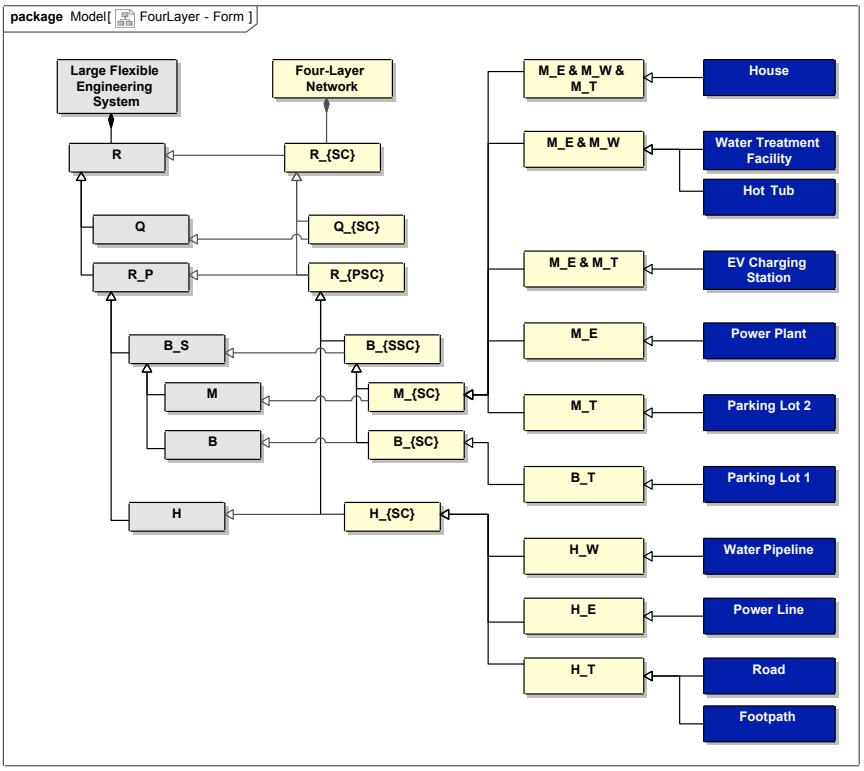


Fig. 4.41: SysML Block Definition Diagram of the Four-Layer Network as a specialization of the LFES meta-architecture.

exiting the electric vehicle. The set of transformation processes P_μ has size seven. The set is $P_\mu = \{$consume cold water, consume electric power, enter EV, exit EV, treat water, consume hot water, generate electric power$\}$. The set of transportation processes P_η has size $\sigma(B_S)^2 = 49$. The set of holding processes P_γ has size six. It contains: $P_\gamma = \{$potable water, electric power, park EV, charge EV, discharge EV, carry pedestrian$\}$.

The capabilities are calculated as the mapping of system processes onto system resources. The knowledge base J_S is calculated using Equation 3.16 on Page 74. The size of J_S is 301×17, and it contains 34 capabilities. The SysML block diagram in Figure 4.43 presents the unique set of capabilities for each of the resource classes in the four-layer network. The capabilities are visualized in Figure 4.44, in which their locations correspond to the location of their associated resources.

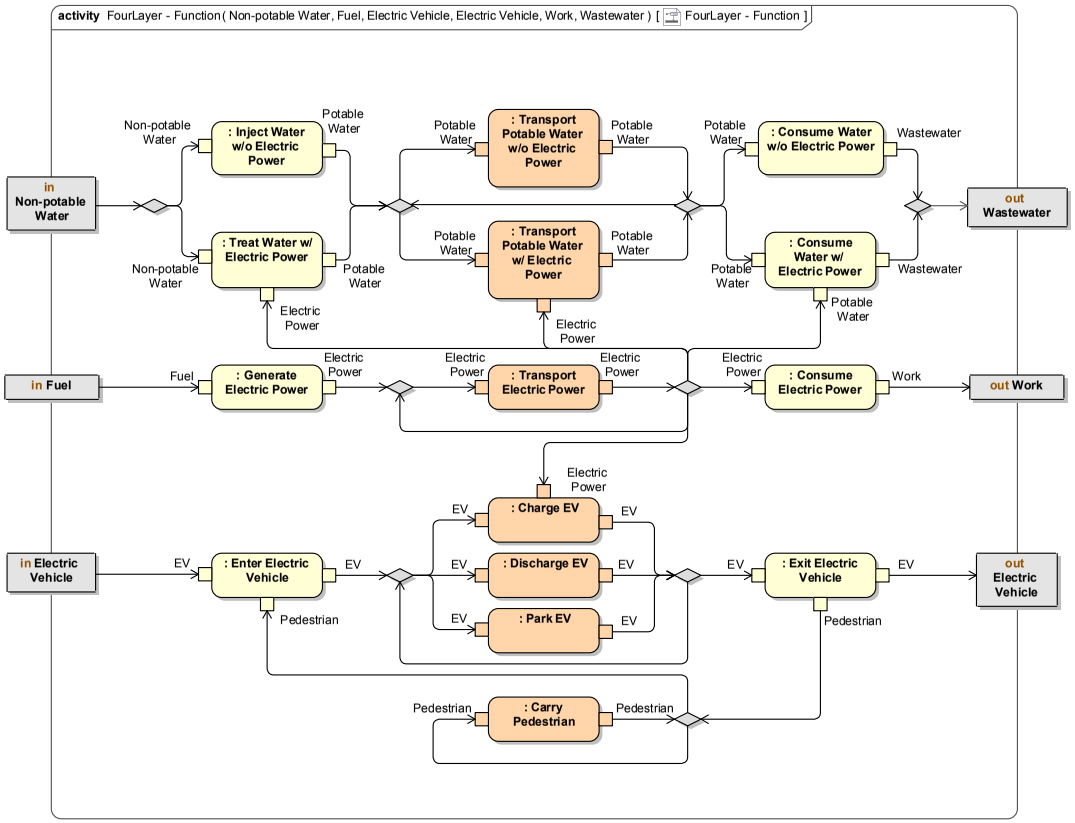


Fig. 4.42: Activity Diagram of the Four-Layer Network Reference Architecture: The four operands are water, electric power, EV, and pedestrians.

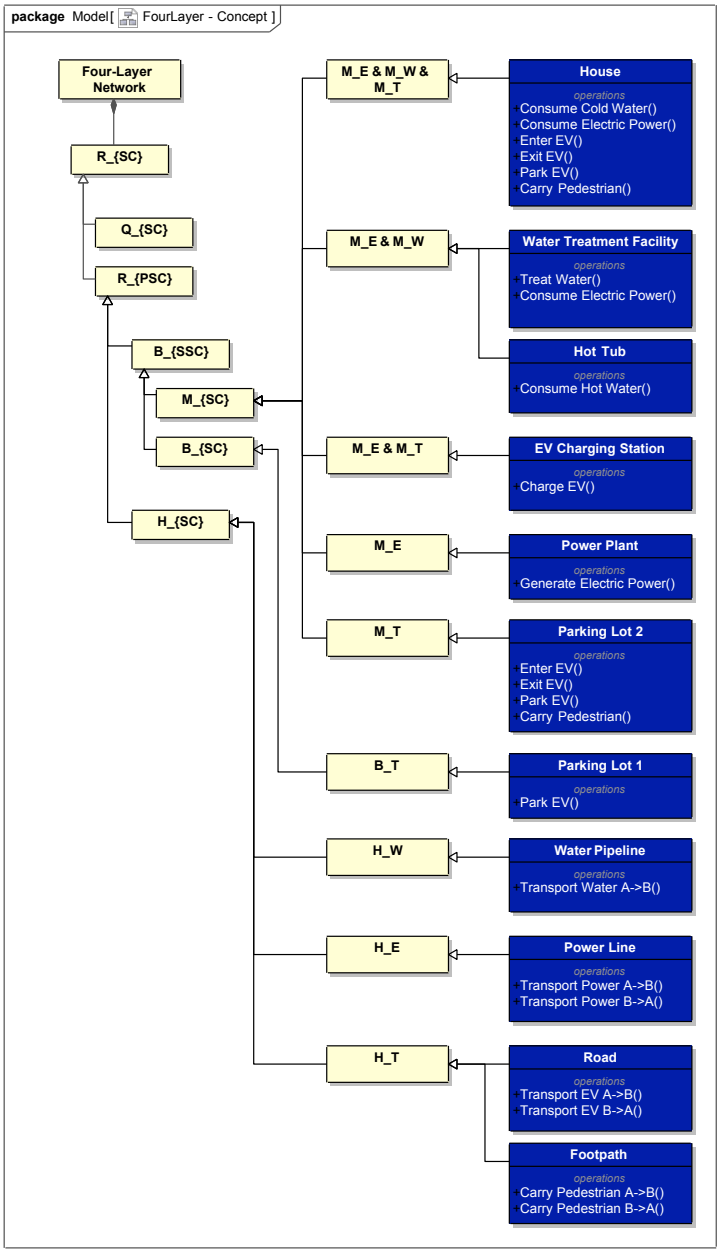


Fig. 4.43: SysML Representation of the System Concept for the Four-Layer Network: This figure contains the unique set of capabilities for each of the resource classes.

4.11.2 Hetero-functional Adjacency Matrix

The hetero-functional adjacency matrix couples the capabilities to represent the logical order of physical capabilities in a system. The hetero-functional adjacency matrix A_ρ and the system sequence degrees of freedom are calculated by Equations 3.28 and 3.29 on Page

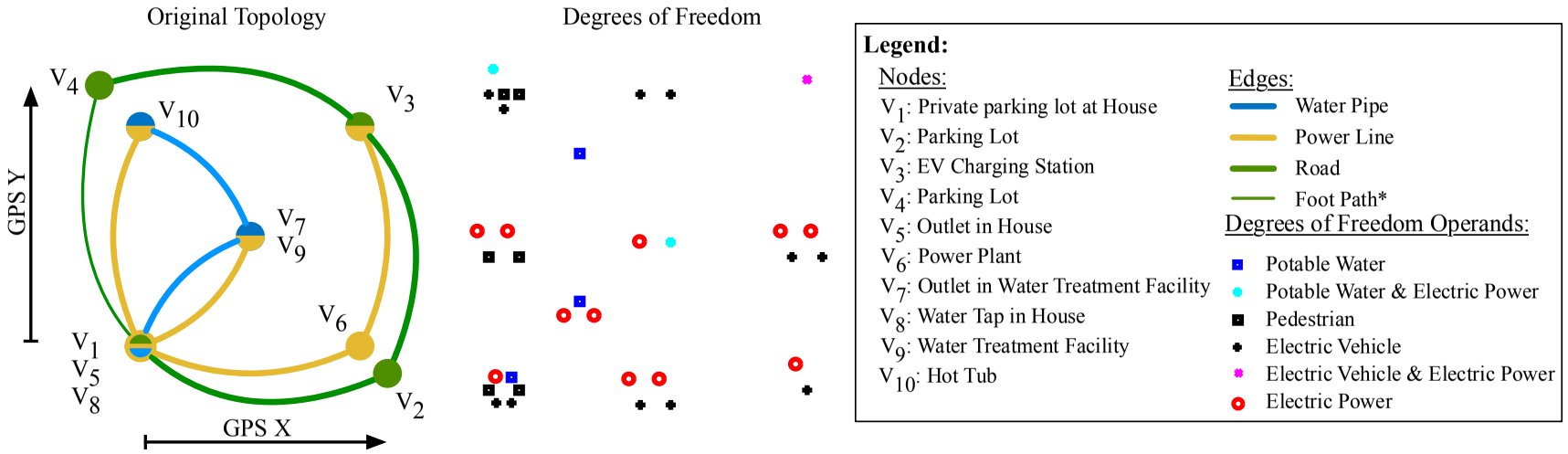


Fig. 4.44: Topological presentation of the Degrees of Freedom in the Four-Layer Network: The original network topology is presented on the left, and the structural degrees of freedom are presented on the right. The degrees of freedom are classified by their operand type, e.g. “exit EV at house" has operand Electric Vehicle, whereas “enter EV at house" has operand Pedestrian.

83. For these equations, Table 3.3 on Page 82 presents the physical continuity constraints, and the design pattern in Figure 4.42 represents the functional sequence constraints. The size of A_ρ is: $\sigma(R)\sigma(P) \times \sigma(R)\sigma(P) = 5,117 \times 5,117$. The projected matrix $\widetilde{A}_\rho$ has size $\sigma(\mathcal{E}_S) \times \sigma(\mathcal{E}_S) = 34 \times 34$. The number of sequence-dependent degrees of freedom is 79. Figure 4.45 presents the visualization of the hetero-functional adjacency matrix superimposed on the degrees of freedom.

4.11.3 Controller Agency Matrix

The controller agency matrix serves to differentiate between two systems of equivalent capabilities but different control structure. Figure 4.39 introduces six controller agents for the four-layer network. Each of these controller agents controls one or more nodes. However, the figure does not introduce a cyber-physical interface between the edges and the controller agents. The following control relationships are assumed: Node v_{12} controls the water pipelines, node v_{13} controls the power lines, and node v_{15} controls the footpath, and roads (it represents individuals that use the transportation network).

The independent controller agency matrix $\overline{A}_Q$ is defined in Definition 3.18 on Page 87. It maps the physical resources onto the cyber-resources Q. The set of cyber-resources has size six, and contains: $Q = \{$Person Operating Hot Tub, Water Treatment Facility Operators, Power Utility, Parking Lot Operator, Person in House, Operator of EV Charging Station$\}$. Consequently, the size of the independent controller agency matrix is $\sigma(R_P) \times \sigma(Q) = 17 \times 6$. The controller agency matrix is now mapped onto the capabilities, as demonstrated for Trimetrica in Equation 4.14 on Page 173. The size of the matrix now is: $\sigma(\mathcal{E}_S) \times \sigma(Q) = 34 \times 6$. The number of filled elements is 34. The matrix is visualized as a bipartite graph between the cyber-resources and the capabilities in Figure 4.46.

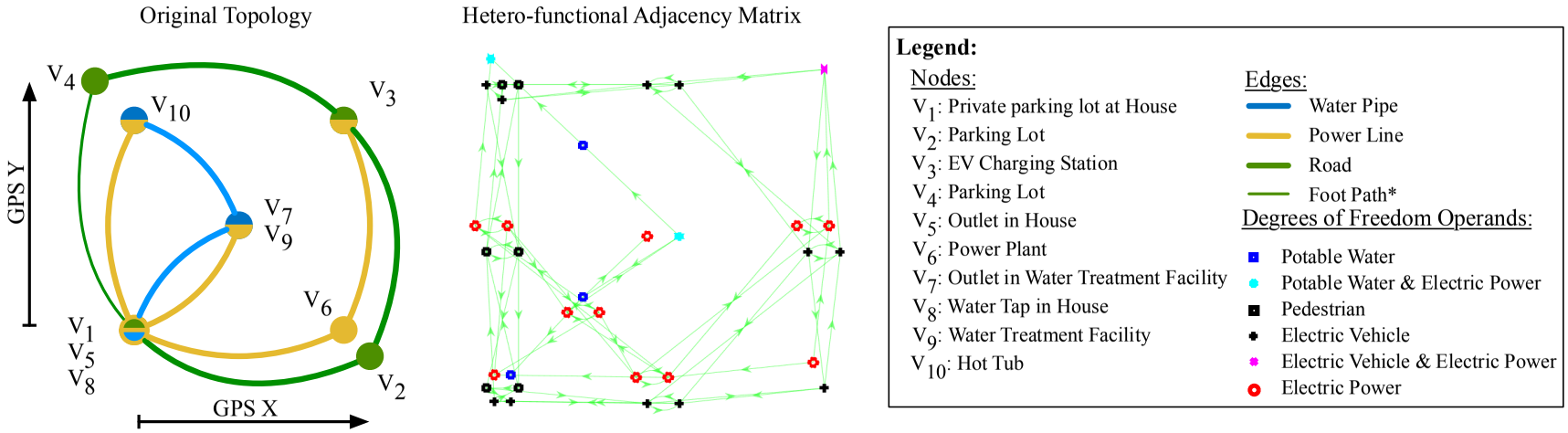


Fig. 4.45: Topological presentation of the Hetero-functional Adjacency Matrix: The original network topology is presented on the left, and the structural degrees of freedom and the hetero-functional adjacency matrix are presented on the right.

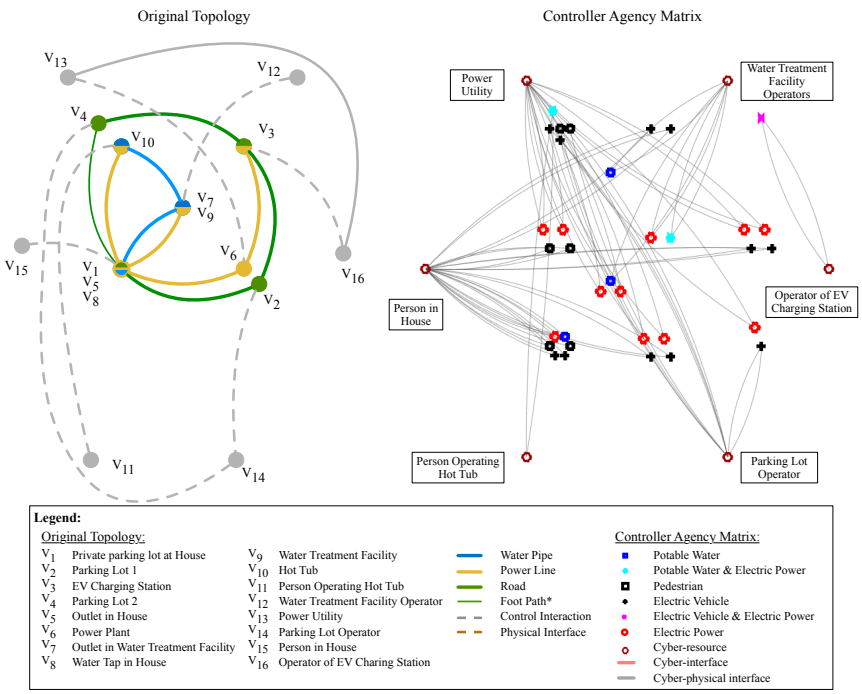


Fig. 4.46: Controller Agency Matrix for the Four-Layer Network.

4.11.4 Controller Adjacency Matrix

The controller adjacency matrix serves to describe the interactions between the cyber-resources as cyber-interfaces. The four-layer network has six cyber-resources, and consequently, its controller adjacency matrix has size 6×6. Figure 4.39 contains a single cyber-interface, between nodes v_{13} and v_{16}. Since the figure does not specify the cyber-interface in greater detail, a bidirectional interface is assumed. The controller adjacency matrix, therefore, contains two filled elements. It is visualized in Figure 4.47. The figure shows two interfaces, and four cyber-resources without any cyber-interfaces.

4.11.5 Service as Operand Behavior

The fifth model in hetero-functional graph theory describes the evolution of operands as they move through the physical infrastructure system. The system capabilities and their control structure have been described in the previous sections, and service model describes how the capabilities act on the operands.

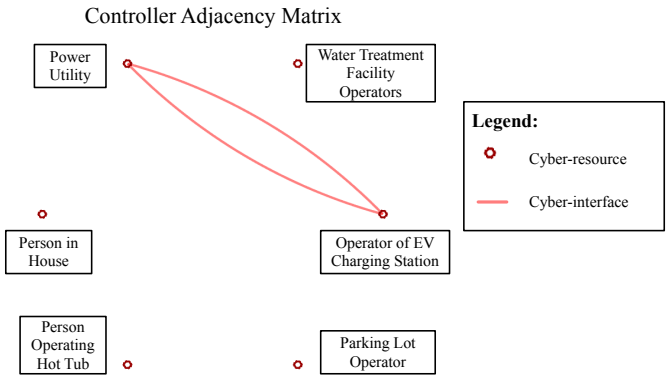


Fig. 4.47: Controller Adjacency Matrix for the Four-Layer Network.

The four-layer network performs three services, that are presented in Figure 4.48 as service nets. The service *"deliver water"* consists of four transitions and one state. The state S_{1l_1} is "potable water". The set of service transitions is: $\mathcal{E}_{l_1}$ = {treat water(), maintain water(), consume hot water(), consume cold water()}. The second service, *"deliver electric power"*, consists of six transitions and one state. The state S_{1l_2} is "electric power". The set of service transitions is: $\mathcal{E}_{l_2}$ = {generate electric power(), maintain electric power(), treat water(), consume hot water(), consume electric power(), charge EV()}. The last service, *"deliver EV"*, consists of three states and seven service transitions. Note that the name of this service does not cover all of its content, because the service also includes the movement of pedestrians. The set of the three states is: S_{l_3} = {pedestrian in EV, pedestrian outside EV, state-of-charge EV}. The set of service transitions is: $\mathcal{E}_{l_3}$ = {enter EV(), exit EV(), walk outside EV(), stay outside EV(), charge EV(), maintain EV(), discharge EV()}. Note that this service has a particularly interesting service net, as the pedestrian has state "in EV", while the EV operates. The EV cannot operate if the pedestrian is not in the EV.

Furthermore, the service nets are converted to service graphs, that show the adjacency of the service transitions. Figure 4.49 shows the three service graphs for each of the services in the four-layer network. The service adjacency matrix for *deliver water* has a size of 4×4, with six filled elements that represent the adjacency. The service adjacency matrix for *deliver electric power* has a size of 6×6, with ten filled elements. Finally, the service

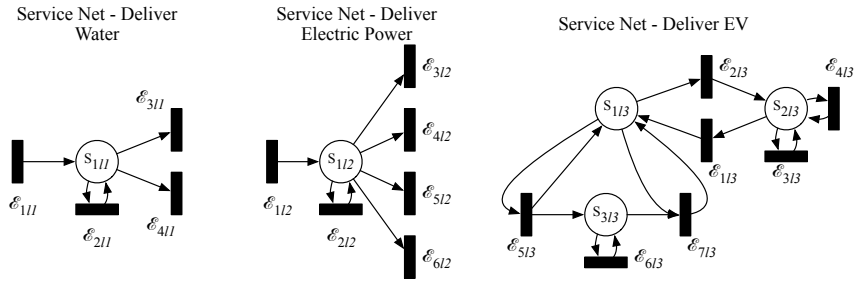


Fig. 4.48: Service Nets for the Four-Layer Network: The four-layer network delivers three services: (1) deliver potable water, (2) deliver electric power, and (3) deliver EV.

adjacency matrix for *deliver EV* has a size of 7×7, with 22 filled elements[11].

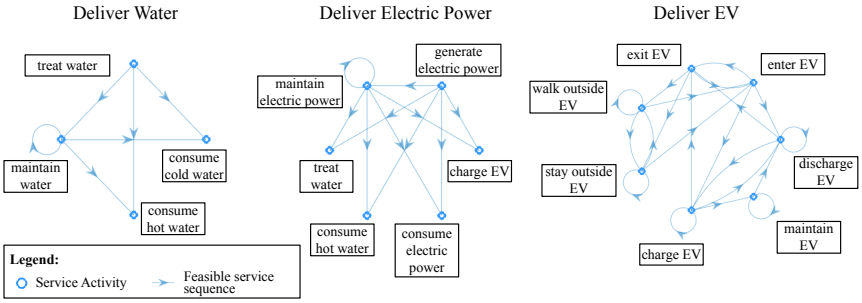


Fig. 4.49: Service Graphs for the Four-Layer Network.

4.11.6 Service Feasibility Matrix

The service feasibility matrices complete the service model by coupling the service transitions to the structural capabilities. As a result, it ensures that as the state of the engineering system's operands evolve, the state of the engineering system itself also evolves. Figure 4.50 presents the service feasibility matrix for the four-layer network as a bipartite graph between the service transitions and the structural degrees of freedom. The service feasibility matrix is calculated for each of the operands by Equation 3.72 on Page 115. The service feasibility matrix for the service *deliver water* is Λ_1, which has a size of 4×301. When projected, $\widetilde{\Lambda}_1$ has a size of 4×34, with five filled elements. The service feasibility matrix for the service *deliver electric power* is Λ_2, which has a size of 6×301. When projected,

[11] Note that the service net contains two paths that connect $\mathcal{E}_{4l_3}$ and $\mathcal{E}_{6l_3}$.

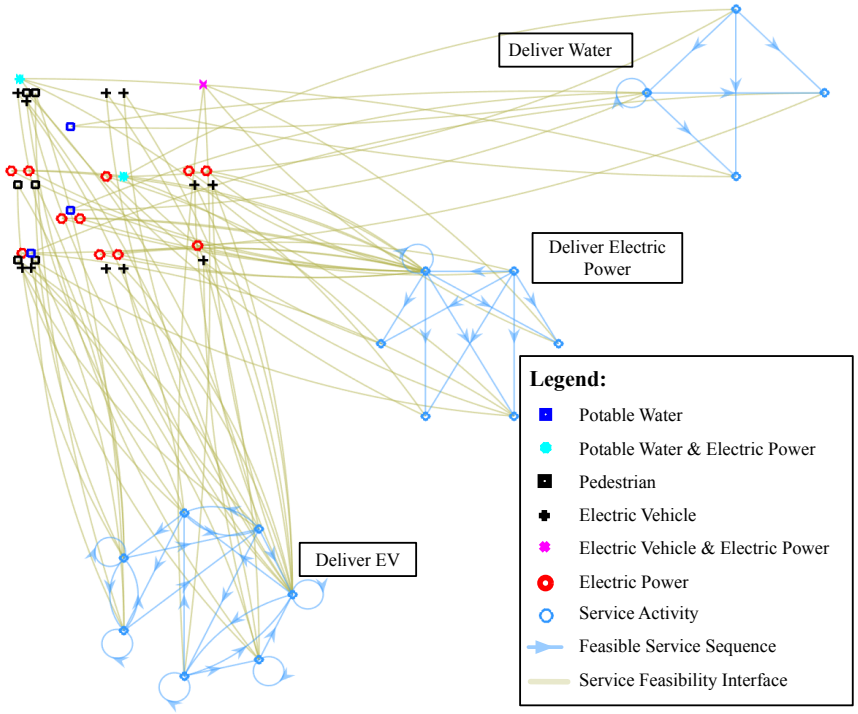


Fig. 4.50: Service Feasibility Matrix as a Bipartite Graph for the Four-Layer Network.

$\widetilde{\Lambda}_2$ has a size of 6×34, with 14 filled elements. The service feasibility matrix for the service *deliver electric vehicle* is Λ_3, which has a size of 7×301. When projected, $\widetilde{\Lambda}_3$ has a size of 7×34, with 18 filled elements.

4.11.7 System Adjacency Matrix

The system adjacency matrix is the final model of hetero-functional graph theory. It integrates the six previous models to provide a holistic representation of the cyber-physical engineering system. Equation 3.67 on Page 112 provides the calculation of the projected system adjacency matrix $\widetilde{\mathbb{A}}$:

$$\widetilde{\mathbb{A}} = \begin{bmatrix} \mathbb{A}_L & \widetilde{\mathbb{A}}_{L\rho} & \mathbf{0} \\ \widetilde{\mathbb{A}}_{\rho L} & \widetilde{A}_{\rho} & \widetilde{\mathbb{A}}_{\rho C} \\ \mathbf{0} & \widetilde{\mathbb{A}}_{C\rho} & A_C \end{bmatrix} \tag{4.28}$$

The matrix $\widetilde{A}_{\rho}$ is the hetero-functional adjacency matrix, as calculated in Section 4.11.2. Matrix A_C is the controller adjacency matrix, as calculated in 4.11.4. Matrix $\mathbb{A}_L$ is the concatenation of the service transition adjacency matrices. The matrices are calculated in Section 4.11.5, and concatenated as demonstrated in Equation 3.68 on Page 114. Matrix $\widetilde{\mathbb{A}}_{\rho C}$ maps the independent cyber-resources to the structural degrees of freedom, as calculated in Section 4.11.3. Finally, matrix $\widetilde{\mathbb{A}}_{L\rho}$ is the service feasibility matrix that maps the service transitions to the structural degrees of freedom, as calculated in Section 4.11.6.

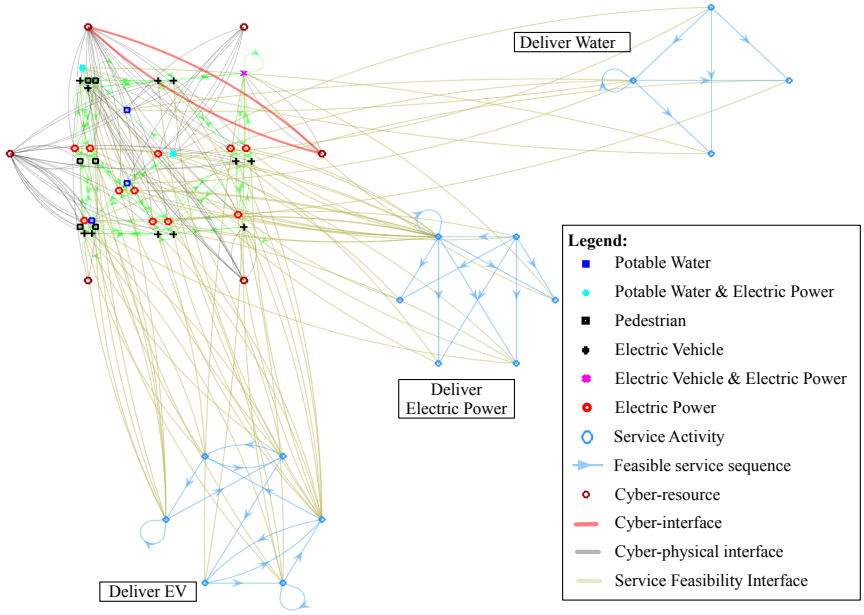


Fig. 4.51: System Adjacency Matrix for the Four-Layer Network.

Chapter Summary:

The chapter demonstrates the application of hetero-functional graph theory by modeling two different engineering system test cases. Sections 4.1 through 4.10 introduce, model, and discuss *"Trimetrica"*. This example shows that hetero-functional graph theory can be applied to large scale systems with extreme heterogeneity. The example also shows that hetero-functional graph theory overcomes the ontological and modeling constraints found in the multilayer networks literature. Section 4.11 models the 4 Layer Test Case to explicitly show that hetero-functional graph theory can model engineering systems that cannot be described with multilayer networks. ■

Chapter 5

Dynamic System Modeling with Hetero-functional Graph Theory

Chapter Abstract:

This chapter leverages the hetero-functional graph theory structural model as a foundation for the development of a dynamic microgrid-enabled production system model. The content of this chapter is directly adopted from a 2017 journal article entitled *"A Dynamic Model for the Energy Management of Microgrid-Enabled Production Systems"* in the *Journal of Cleaner Production* [35][1].

This chapter advances contributions in three areas. First, the chapter develops a hetero-functional graph theory structural model of a a microgrid-enabled production system. It leverages system concept, service as operand behavior, and the service feasibility matrix.

Thereafter, a dynamic model of a microgrid-enabled production system is developed. The model is based on the hetero-functional graph theory structural model and constructed as a system of device models, associated with the structural degrees of freedom. The goal of the dynamic model is to describe the product and power flows in the microgrid-enabled

[1]Note that the work in this chapter was developed *before* the hetero-functional graph theory book [5]. The method to develop the structural model in this chapter is different than the approach of Chapter 3. Both approaches are equally correct, though the approach from Chapter 3 is more scalable as a result of more extensively use of SysML during the definition of the system concept

production system. This is the first integrated dynamic model of a microgrid-enabled production system and it allows for a system of arbitrary size, topology, and coupling.

Finally, the chapter demonstrates the dynamic model by implementing and simulating a test case. The test case has the goal to highlight the interactions between the discrete-event production system dynamics and the continuous time power flow dynamics. Furthermore, the carbon emissions are calculated as a proxy of sustainability (as a *life-cycle property* of this *engineering system*, see Section 2.2.1).

5.1 Introduction

Industrial facilities are devoting ever greater attention to the importance of energy in their operations [262]. In some cases, industrial processes are energy intensive [263]. In other cases, such as with electric drives and motors, the energy supply must be of a very high quality [264]. Finally, many industrial companies are increasingly concerned with decarbonization either as imposed by regulation [265, 266], or stimulated by corporate social responsibility [267, 268]. To that end, onsite distributed renewable energy are often integrated. The combined objectives of high availability, quality and sustainability are often beyond the capabilities of the local utility and so this work considers *microgrid-enabled production systems* where the production enterprise has decided to in-source its energy supply so as to tailor its management and delivery to the needs of its production system. Microgrids have the potential to reduce local costs, energy losses, and greenhouse gases while enhancing energy reliability [269–271]. They also give owners the necessary autonomy to operate islanded from the local electric utility.

The introduction of microgrids to a production system setting opens several new industrial energy management activities in both planning and operations [272]. Energy planning focuses on energy efficiency. This static energy management approach optimizes the produc-

tion system before it has been built, or when new technologies are incorporated in upgrades and retrofits [273, 274]. This not only considers production system resources, but also power systems resources, such as a microgrid infrastructure or the integration of renewable energy. Operations, or dynamic energy management, tries to optimize the usage of a given set of resources for energy consumption or cost [275, 276]. Such decisions coincide with the dynamics of the production system and may respond to several externalities, such as changes in the power grid balance, customer demand, and time-of-use prices [277–280].

Dynamic energy management is of particular importance to microgrid power balancing activities [281–285]. The trend of increasing renewable energy resources penetration in the power grid causes increases in variability and stochasticity of the power supply [286, 287]. Meanwhile, dynamic energy management controls the loads and has the potential to be a part of demand response [288, 289]. Furthermore, when operating in islanded model, the size of the microgrid limits the number of levers that control the balance; especially when the generation is stochastic. As a result, the production system and microgrid dynamics are coupled and cannot be studied separately. Energy consumption is a *byproduct* of production system activities, and the added value of energy differs per activity [290–292]. Consequently, online approaches that rely on multi-agent structures, or reactive control [293] have been developed. Meanwhile offline approaches, based on resource constraint scheduling methods or weighted multi-objective optimization have been discussed extensively [294, 295].

However, these studies do not include multi-discipline models, where the production system and microgrid dynamics are described in an unambiguous language. The need for such a multi-layer model has been discussed extensively by the network sciences community. An important remark made in [95] is that hetero-functional multi-layer models capture the interdependency and interaction of systems in a way that two separate mono-layer models cannot comprehend. This comprehension is essential to understanding structural properties such as resilience, robustness, and centrality. The approach presented in this work has been already used to study the resilience of hetero-functional networks [14].

5.1.1 Scope

In developing a dynamic model for the energy management of microgrid-enabled production systems, the work restricts scope to discrete-part production systems. It also allows the microgrid to have a significant penetration of variable (renewable) energy resources. To ground the discussion, it is useful to consider a bird feeder manufacturing system [2, 6, 20] which is detailed later in Section 5.4 as an illustrative example.

Table. 5.1: Microgrid-Enabled Production System Operations Management Decisions

Product Dispatch	When a given product part should undergo production in the facility.
Integrated Planning & Scheduling	Which sequence of processes should be used to create the product.
W.I.P. Management	When & where the product parts should be buffered.
Peak Load Management	At a given machine, when the product parts should be processed to meet the production schedule and power grid constraints.
Ancillary Services	Given the dynamics of the power grid, how can the planning & scheduling of production activities support stabilization.

The dynamic model for microgrid-enabled production systems must also be able to accommodate several operations management decisions. Such decisions, when made, serve to advance the physical state and outputs of the microgrid-enabled production system so as to achieve its operations management objectives. These are summarized in Table 5.1. The first decision is dispatch of product parts to different machines to undergo transforming (value-adding) processes. Second, planning and schedule determines the sequence and timing of production processes as the product parts advance through the facility. Third, it is important to actively manage where and if work-in-progress will be buffered. These three decisions are inherently coupled to the microgrid power balance. They must now also simultaneously manage the peak load constraints of the microgrid. Finally, it is possible that the operations management decisions of the production system can serve to provide ancillary services that stabilize the microgrid.

5.1.2 Contribution

This paper seeks to develop a dynamic model for the energy management of microgrid-enabled production systems. The goal of the model is to simultaneously analyze the flows of

products in a discrete-part production system and the power flows in the coupled microgrid. The model is developed in such a way as to describe a production system and microgrid of arbitrary size, topology and coupling. The model also specifically includes renewable energy to support the objective of greater sustainability. The load balance requirements resulting from the renewable energy integration impose the need for a dispatchable generator, with related carbon emissions.

Meanwhile, this work advances a broader body of literature in the network sciences called *multi-layer networks* [95]. Very few works present models where the layers do not have to be aligned or of the same size. The work presented in this paper advances the theory of multi-layer networks, as a result of the aforementioned coupling of the production system and the microgrid.

The model developed here is based on a *hetero-functional graph theory* rooted in the Axiomatic Design for Large Flexible Engineering Systems [4, 98]. The approach has been used to develop measures of reconfigurability of production systems [2,6,15,20]. Later it was applied to model other complex multi-layer systems such as the transportation [25,30,31,33], water [296], and power domains [27]. Several recent works also demonstrate how the theory may be applied to decision-marking; including the development of Multi-Agent System Design principles for both Future Power Systems [27] and Reconfigurable Mechatronic Systems [22].

Figure 5.1 provides an overview of the logical flow of the paper. The block elements are numbered consistently with their associated section numbers. Section 5.2 lays the methodological foundation of hetero-functional graph theory. Section 5.2.1 utilizes Axiomatic Design for Large Flexible Engineering Systems (AD4LFES) as generic model of system structure and Section 5.2.2 uses Petri nets as a generic model of spatially distributed discrete-event dynamics. Section 5.3 then develops the dynamic model for energy management by instantiation. Section 5.3.1 uses AD4LFES to develop a structural model of a microgrid-enabled production system. Section 5.3.2 and Section 5.3.3 then superimpose

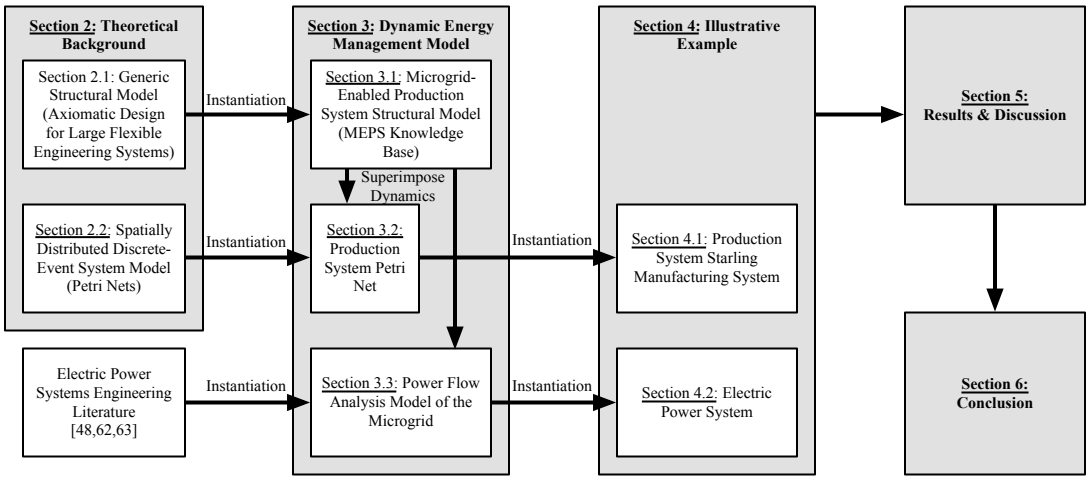


Fig. 5.1: Visual representation of the document structure.

dynamics upon this structure to develop a production system petri net model and a power flow analysis model of the microgrid respectively. Section 5.4 introduces a case study consisting of an example production system connected to an example microgrid electric power system. Section 5.5 then presents and discusses the results of the case study as a validation of the model. The article is concluded with Section 5.6.

5.2 Background

The methodological foundation of this work is a hetero-functional graph theory rooted in Axiomatic Design for Large Flexible Engineering Systems and Petri nets. The former is a concise way of describing the system structure and is discussed in subsection 5.2.1. The latter describes the system behavior as discrete-event dynamics. It is discussed in subsection 5.2.2.

5.2.1 Axiomatic Design for Large Flexible Engineering Systems

Microgrid-enabled production systems may be classified as a Large Flexible Engineering System. This subsection recalls how Axiomatic Design treats these systems in terms of

their system resources, system processes, and the allocation of the latter to the former [8, 15, 22, 25, 27, 30, 31, 296]

Definition 5.1: Large Flexible Engineering System (LFES) [3, 14]: an engineering system with many functional requirements that not only evolve over time, but also can be fulfilled by one or more design parameters. ■

Although this paper relies on an Axiomatic Design structural model, it departs from its terminology without any change in interpretation. The functional requirements and the design parameters are understood to be mutually exclusive and collectively exhaustive sets of the system's processes (P) and resources (R).

As a theory, Axiomatic Design for Large Flexible Engineering Systems presents several advantages that are directly relevant to microgrid-enabled production systems. These include modeling support for:

- systems of heterogeneous modes of production (but identical function),
- systems of a fundamentally hetero-functional nature,
- the allocation of system function to form,
- systems of variable structure to support reconfigurable operation.

The set of system's resources $R = M \cup B \cup H$ are classified as the set transforming resources $M = \{m_1, \ldots, m_{\sigma(M)}\}$, independent buffers $B = \{b_1, \ldots, b_{\sigma(B)}\}$, and transporting resources $H = \{h_1, \ldots, h_{\sigma(H)}\}$, with $\sigma()$ being the size of the set. For later simplicity, the set of buffers $B_S = M \cup B$ is also defined [8, 15]. Since Axiomatic Design for LFES supports modeling of complex systems, aggregating of resources R can be useful to combine different systems.

The aggregation matrix Ξ assigns resource r_j to aggregated resource $\overline{r}_a$ to create the set of aggregated resources $\overline{R}$ [6, 20].

$$\overline{R} = \Xi \circledast R \tag{5.1}$$

where Ξ is an aggregation matrix and $\circledast$ is the aggregation operator.

Definition 5.2 – Aggregation Operator $\circledast$ [2, 6, 20]: Given boolean matrix A and sets B and C,

$$C(i,k) = \bigcup_{j} a(i,j) \odot b(j,k) = A \circledast B \tag{5.2}$$

■

The set of system's processes P can be classified as the set of transforming, transporting, and holding processes.

Definition 5.3: Transformation Process [8, 15, 20]: A resource-independent, technology-independent process $p_{\mu_j} \in P_\mu = \{p_{\mu_1} \dots p_{\mu_{\sigma(P_\mu)}}\}$ that transforms an artifact from one form into another. ■

Definition 5.4: Transportation Process [8, 15, 20]: A resource-independent process $p_{\eta u} \in P_\eta = \{p_{\eta 1} \dots p_{\eta\sigma(P_\eta)}\}$ that transports artifacts from one buffer b_{sy_1} to b_{sy_2}. There are $\sigma^2(B_S)$ such processes of which $\sigma(B_S)$ are "null" processes where no motion occurs. The following convention of indices is adopted:

$$u = \sigma(B_S)(y_1 - 1) + y_2 \tag{5.3}$$

■

This convention implies a directed bipartite graph between the set of independent buffers and the transportation processes whose incidence out M_H^- and incidence in M_H^+ matrices are given by [27]:

$$M_H^- = \sum_{y1=1}^{\sigma(B_S)} e_{y1}^{\sigma(B_S)} [e_{y1}^{\sigma(B_S)} \otimes \mathbb{1}^{\sigma(B_S)}]^T \tag{5.4}$$

$$M_H^+ = \sum_{y2=1}^{\sigma(B_S)} e_{y2}^{\sigma(B_S)} [\mathbb{1}^{\sigma(B_S)} \otimes e_{y2}^{\sigma(B_S)}]^T \tag{5.5}$$

where $\mathbb{1}^n$ is a column ones vector of predefined length n, e_i^n is the i^{th} elementary basis

vector, and $\otimes$ is the kronecker product.

Definition 5.5: Holding Process [8, 15, 20]: A transportation independent process $p_{\gamma g} \in P_{\gamma}$ that holds artifacts during the transportation from one buffer to another. ■

Together, the system processes and resources address hetero-functionality in a LFES.

System processes P are allocated to system resources R via the system knowledge base J_S. The Axiomatic Design equation describes the allocation [8, 15, 20]:

$$P = J_S \odot R \tag{5.6}$$

where $\odot$ is matrix boolean multiplication and J_S is defined as:

Definition 5.6: System Knowledge Base [8, 15, 20]: A binary matrix J_S of size $\sigma(P) \times \sigma(R)$ whose element $J_S(w, v) \in \{0, 1\}$ is equal to one when action e_{wv} (in the SysML sense [69]) exists as a system process $p_w \in P$ being executed by a resource $r_v \in R$. ■

The system knowledge base J_S is constructed based on the smaller knowledge bases that address transformation, transportation and holding processes individually.

$$\begin{aligned} P_{\mu} &= J_M \odot M \\ P_{\eta} &= J_H \odot R \\ P_{\gamma} &= J_{\gamma} \odot R \end{aligned} \tag{5.7}$$

J_S then becomes [8, 15, 20]:

$$J_S = \left[\begin{array}{c|c} J_M & \mathbf{0} \\ \hline & J_{\bar{H}} \end{array}\right] \tag{5.8}$$

where in order to account for the simultaneity of holding and transportation processes into refined transportation processes $P_{\bar{\eta}}$ [8, 15, 20]:

$$J_{\bar{H}} = \left[J_{\gamma} \otimes \mathbb{1}^{\sigma(P_{\eta})}\right] \cdot \left[\mathbb{1}^{\sigma(P_{\gamma})} \otimes J_H\right] \tag{5.9}$$

where $\cdot$ is the hadamard product.

The system knowledge base has an additional property in that it defines the LFES structural degrees of freedom. These are important in the construction of the dynamic system state vector in large complex systems.

Definition 5.7: Structural Degrees of Freedom [8, 15, 20]: The set of independent actions $\mathcal{E}_S$ that completely defines the available processes in a LFES. Their number is given by:

$$DOF_S = \sigma(\mathcal{E}_S) = \sum_{w}^{\sigma(P)} \sum_{v}^{\sigma(R)} J_S(w,v) \ominus K_S(w,v) \tag{5.10}$$

■

where $\ominus$ is boolean subtraction and K_S is a constraints matrix of appropriate size that eliminates actions from the action set. It serves to distinguish between the existence and the availability of a degree of freedom. For example, scheduled maintenance can disable a degree of freedom removing it from the system structure.

Previous work has applied a vectorization of the system knowledge base for mathematical convenience [27]. The shorthand $()^V$ is used to replace $vec()$. Additionally, a projection operator projects the vectorized knowledge base onto a one's vector to eliminate sparsity $\mathbb{P}_S J_S^V = \mathbb{1}^{\sigma(\mathcal{E}_S)}$. While solutions for $\mathbb{P}_S$ are not unique, this work chooses:

$$\mathbb{P}_S = \left[e_{\psi_1}^{\sigma(P)\sigma(R)}, \ldots, e_{\psi_{\sigma(\mathcal{E}_S)}}^{\sigma(P)\sigma(R)} \right] \tag{5.11}$$

where $e_{\psi_i}^{\sigma(P)\sigma(R)}$ is the ψ_i^{th} elementary row vector corresponding to the first up to the last structural degree of freedom in increasing order.

5.2.2 Petri Nets

Petri nets are a concise and commonly used tool to model discrete part production system dynamics. This subsection describes their untimed and timed variants for later use.

Definition 5.8: Marked Petri Net (Graph) [129]: A bipartite directed graph represented as a 5-tuple $\mathcal{N} = \{S, \mathcal{E}, \mathbf{M}_{PN}, W, Q_{PN}\}$ where:

- S is a finite set of places of size $\sigma(S)$.
- $\mathcal{E}$ is a finite set of transitions/events of size $\sigma(\mathcal{E})$.
- $\mathbf{M}_{PN} \subseteq (S \times \mathcal{E}) \cup (\mathcal{E} \times S)$ is a set of arcs of size $\sigma(\mathbf{M}_{PN})$ from places to transitions and from transitions to places in the graph.
- $W : \mathbf{M}_{PN} \rightarrow \{0, 1\}$ is the weighting function on arcs.
- Q_{PN} is a marking (or discrete state) vector of size $\sigma(S) \times 1 \in \mathbb{N}^{\sigma(S)}$.

■

The arcs of the Petri net graph and its weightings define the Petri net incidence matrix.

Definition 5.9: Petri Net Incidence Matrix [129]: An incidence matrix M_{PN} of size $\sigma(S) \times \sigma(\mathcal{E})$ where:

$$M_{PN} = M_{PN}^{+} - M_{PN}^{-} \tag{5.12}$$

where $M_{PN}^{+}(y, \psi) = w(e_{wv}, b_y)$ and $M_{PN}^{-}(y, \psi) = w(b_y, e_{wv})$ and ψ is a unique index mapped from the ordered pair (w, v). ■

The Petri net structure leads directly to the definition of its discrete-event dynamics.

Definition 5.10: Petri Net (Discrete-Event) Dynamics [129]: Given a binary firing vector $U[k]$ of size $\sigma(\mathcal{E}) \times 1$ and a Petri net incidence matrix M_{PN} of size $\sigma(S) \times \sigma(\mathcal{E})$, the evolution of the marking vector Q_{PN} is given by the state transition function $\Phi(Q_{PN}[k], U[k])$:

$$Q_{PN}[k+1] = \Phi(Q_{PN}, U[k]) = Q_{PN}[k] + MU[k] \tag{5.13}$$

■

Beyond ordinary Petri nets, timed Petri nets, as their name suggests, introduce time into the dynamics definition.

Definition 5.11: Timed Petri Net (Discrete-Event) Dynamics [249]: Given a binary input firing vector $U^+[k]$ and a binary output firing vector $U^-[k]$ both of size $\sigma(\mathcal{E}) \times 1$, and the positive and negative components M^+_{PN} and M^-_{PN} of the Petri net incidence matrix of size $\sigma(S) \times \sigma(\mathcal{E})$, the evolution of the marking vector Q_{PN} is given by the state transition function $\Phi_T(Q_{PN}[k], U^-[k], U^+[k])$:

$$Q_{PN}[k+1] = \Phi_T(Q_{PN}[k], U^-[k], U^+[k]) \tag{5.14}$$

where $Q_{PN} = [Q_S; Q_\mathcal{E}]$ and

$$Q_S[k+1] = Q_S[k] + M^+_{PN} U^+[k] - M^-_{PN} U^-[k] \tag{5.15}$$

$$Q_\mathcal{E}[k+1] = Q_\mathcal{E}[k] - U^+[k] + U^-[k] \tag{5.16}$$

■

Note that Timed Petri Nets separate the infinitesimally short firing vector of ordinary Petri nets into distinct input and the output firing vectors separated by processing times. Q_S and $Q_\mathcal{E}$ track the location of the tokens in the production system. Q_S tracks the states of the buffers and $Q_\mathcal{E}$ the state of (ongoing) transitions [249, 297]. The transitions are fired based on a scheduled event list that combines the discrete events with a time interval.

Definition 5.12: Scheduled Event List [129]: A tuple $\mathcal{S} = (u_\psi[k], t_k)$ consisting of all elements $u_\psi[k]$ in firing vectors $U^-[k]$ and their associated times t_k. For every element, $u^-_\psi[k] \in U^-[k]$, there exists another element $u^+_\psi[\kappa] \in U^+[\kappa]$ which occurs at time t_κ d_ψ time units later. $t_\kappa = t_k + d_\psi$. ■

5.3 Model Development

This section develops the Dynamic Energy Management Model for Microgrid-Enabled Production Systems using Axiomatic Design, timed Petri nets and physical modeling. First,

Table. 5.2: System Processes & Resources in a microgrid-enabled production system [6,7,20]

	P_μ	P_η	P_γ	M	B	H	J_S
LFES	Transformation	Transportation	Holding	Transforming Resource	Independent Buffer	Transportation Resource	Knowledge Base
Production System	Transformation ($P_{\mu PS}$)	Transportation ($P_{\eta PS}$)	Holding ($P_{\gamma PS}$)	Machines (M_{PS})	Input & Output Buffers (B_{PS})	Material Handlers (H_{PS})	Production System Knowledge Base (J_{PS})
Microgrid	Generation & Consumption ($P_{\mu MG}$)	Transmission ($P_{\eta MG}$)	Voltage Level ($P_{\gamma MG}$)	Generators & Loads (M_{MG})	Substations & Storage (B_{MG})	Lines (H_{MG})	Microgrid Knowledge Base (J_{MG})

a single knowledge base for the microgrid-enabled production system is constructed as a succinct description of system structure. The remainder of the section addresses the system behavior using the Axiomatic Design knowledge base as a foundation. A timed Petri net model is derived to describe the production system's discrete event dynamics. Finally, a power flow analysis model is added as a physical model of the microgrid that allows for energy management by analyzing the system behavior [298]. The development represents a logical extension of the production system model in [34] and the transportation electrification model in [30, 33].

5.3.1 Microgrid-Enabled Production System Knowledge Base

This subsection constructs the Microgrid-Enabled Production System Knowledge Base as an overlapping union of two underlying knowledge bases; one for the production system and the other for the microgrid. Table 5.2 provides an overview of the definitions of the resources and processes for each of these systems.

5.3.1.1 Production System Knowledge Base

The production system knowledge base J_{PS} is constructed based on the concepts discussed in Section 5.2.1. First, the production system resources R_{PS} are defined as; the transformation resources M_{PS} that represent the transforming (value-adding) machines in the production system, the independent buffers B_{PS} that represent the input, output, and intermediate buffers, and the transportation resources H_{PS} that represent the material handlers that

relocate product parts [15,20,34].The set of all buffers is also defined as $B_{S_{PS}} = M_{PS} \cup B_{PS}$. Additionally, Definitions 5.3, 5.4, and 5.5 are followed to define the production system's transformation $P_{\mu PS}$, transportation $P_{\eta PS}$, and holding processes $P_{\gamma PS}$, resulting in the set of production system processes: $P_{PS} = P_{\mu PS} \cup P_{\eta PS} \cup P_{\gamma PS}$. Finally, the production system processes are mapped on the production system resources [15,20,34]. The system knowledge base consists of a combination of the underlying knowledge bases, using Equation 5.6, 5.8 and 5.9.

$$J_{PS} = \left[\begin{array}{c|c} J_{M_{PS}} & \mathbf{0} \\ \hline & J_{\bar{H}_{PS}} \end{array}\right] \tag{5.17}$$

where

$$J_{\bar{H}_{PS}} = \left[J_{\gamma PS} \otimes \mathbb{1}^{\sigma(P_{\eta PS})}\right] \cdot \left[\mathbb{1}^{\sigma(P_{\gamma PS})} \otimes J_{H_{PS}}\right] \tag{5.18}$$

5.3.1.2 Microgrid Knowledge Base

The power system knowledge base is constructed similarly. First, the power system resources are defined. The transformation resources M_{MG} include the systems generators and loads. The independent buffers B_{MG} represent buffering capacity in the system, such as energy storage (e.g. batteries) or buses. Substations are modeled as independent buffers without storage capacity. The transportation resources H_{MG} represent the power lines between the buses, generators, and loads [27]. Next, Definitions 5.3, 5.4, and 5.5 are applied to define the microgrid's system processes. The transformation processes $P_{\mu MG}$ represent the generation and consumption of power. The transportation processes $P_{\eta MG}$ represent the power flow through the system. The holding processes $P_{\gamma MG}$ represent different voltage levels as different ways of "carrying" power. These types of processes are combined in the set of microgrid processes $P_{MG} = P_{\mu MG} \cup P_{\eta MG} \cup P_{\gamma MG}$. Finally, the microgrid system processes are mapped on to its resources [27]. The system knowledge base consists of a combination of the underlying knowledge bases, using Equation 5.6, 5.8 and 5.9.

$$J_{MG} = \left[\begin{array}{c|c} J_{M_{MG}} & \mathbf{0} \\ \hline & J_{\bar{H}_{MG}} \end{array}\right] \tag{5.19}$$

where $J_{\bar{H}_{MG}} = J_{H_{MG}}$ is assumed, given that microgrids usually only retain one voltage level. Note that every transportation resource (power line) is related to two transportation processes, one for each direction. Also note that the transforming resources can realize either power generation or a consumption process but not both.

5.3.1.3 Microgrid Enabled Production System Knowledge Base

The Microgrid-Enabled Production System knowledge base is a union of the two previously identified knowledge bases J_{PS} and J_{MG}. However, the two systems are not mutually exclusive. The transformation resources in the power system overlap with production system's resources. The resulting transformation resources for the Microgrid-Enabled Production System (M_{MPS}) can be defined as: $M_{MPS} = R_{PS} \cup \tilde{M}_{MG} = M_{PS} \cup B_{PS} \cup H_{PS} \cup \tilde{M}_{MG}$ where $\tilde{M}_{MG}$ is taken as the set of generators, and loads unrelated to production. Similarly, the production system processes are combined with the microgrid transformation processes: $P_{\mu MPS} = P_{PS} \cup \tilde{P}_{\mu MG} = P_{\mu PS} \cup P_{\bar{\eta} PS} \cup \tilde{P}_{\mu MG}$ where $\tilde{P}_{\mu MG}$ is taken as the set of generating and consuming processes unrelated to production. In order to relate these sets of processes and resources, three mutually exclusive knowledge bases are introduced: $J_{M_{MPS}}$, $J_{\bar{H}_{MPS}}$, and $\tilde{J}_{M_{MG}}$ where:

$$\begin{aligned} P_{\mu MPS} &= J_{M_{MPS}} \odot M_{PS} \\ P_{\bar{\eta} PS} &= J_{\bar{H}_{MPS}} \odot R_{PS} \\ \tilde{P}_{\mu MG} &= \tilde{J}_{M_{MG}} \odot \tilde{M}_{MG} \end{aligned} \tag{5.20}$$

similarly to as defined in Equation 5.7.

Consequently, the Microgrid-Enabled Production System knowledge base J_{MPS}, as a type of large flexible engineering system, takes on the structural form of Equation 5.8:

$$= \left[\begin{array}{c|c} J_{M_{MPS}} & \mathbf{0} \\ \hline & J_{\bar{H}_{MPS}} \end{array}\right] \tag{5.21}$$

Then substituting in Equation 5.17 gives:

$$= \left[\begin{array}{c|c|c} J_{PS} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \tilde{J}_{M_{MG}} & \mathbf{0} \\ \hline & J_{\bar{H}_{MPS}} & \end{array}\right] \tag{5.22}$$

A final decomposition by substitution of Equation 5.20 gives:

$$= \left[\begin{array}{c|c|c|c} J_{M_{PS}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline & J_{\bar{H}_{PS}} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \tilde{J}_{M_{MG}} & \mathbf{0} \\ \hline & & J_{\bar{H}_{MPS}} & \end{array}\right] \tag{5.23}$$

5.3.2 Dynamics of the Production System Domain

This subsection uses Petri nets to describe the discrete-event dynamics in the production system domain. Two types of Petri nets are constructed. The first is called the production system Petri net and describes the evolution of the system processes and resources of the production system as they were originally defined in the Axiomatic Design knowledge base. The second is called the product Petri net[2] and describes the evolution of products as they evolve from raw goods into finished ones. Structurally, it is also built upon the Axiomatic Design knowledge base. Dynamically, it is synchronized with the evolution of the

[2]The Product Petri net, or Product net, is equivalent to the service net, as introduced in Section 3.5

production system Petri net. It is worth mentioning, that at first glance, the developed Petri nets resemble “resource-oriented Petri nets" in the literature [299]. While there are some similarities, a careful inspection reveals that the novel tie to Axiomatic Design assigns very different physical meanings to the Petri net models. Those developed here have the added benefit of being graphically intuitive and can be validated to the axiomatic design knowledge base by inspection. The production system and product Petri nets are now discussed in turn.

5.3.2.1 Production System Petri Net

A discrete-part production system is governed by spatially-distributed discrete-event dynamics. Petri nets have been used extensively to model the behavior of production systems [129, 249, 297, 300, 301]. Here, a production system Petri net is defined as a timed Petri net according to Definitions 5.8 and 5.11. It is at this point that the production system knowledge base in Equation 5.17 is tied to the production system Petri net structure. The production system buffers B_S are equivalent to the production system Petri net places S. Similarly, the Petri net transitions $\mathcal{E}$ are equivalent to the structural degrees of freedom $\mathcal{E}_S$. Consequently, the Petri net incidence matrix in Definition 5.9 follows the convention in Equation 5.3. It may be calculated by Equation 5.12 where M_{PS}^{-} and M_{PS}^{+} are straightforwardly derived using Equations 5.4, 5.5 and 5.8.

$$M_{PS}^{-} = \sum_{y1=1}^{\sigma(B_{S_{PS}})} e_{y1}^{\sigma(B_{S_{PS}})} \left[\mathbb{P}\left(X_{y_1}^{-} \right)^{V} \right]^{T} \tag{5.24}$$

where

$$X_{y_1}^{-} = \left[\begin{array}{c|c} \mathbb{1}^{\sigma(P_{\mu PS})} e_{y1}^{\sigma(B_{S_{PS}})T} & \mathbf{0}^{\sigma(P_{\mu PS})\times\sigma(R_H)} \\ \hline \mathbb{1}^{\sigma(P_{\gamma PS})} \otimes e_{y1}^{\sigma(B_{S_{PS}})} \otimes \mathbb{1}^{\sigma(B_{S_{PS}})} & \otimes \mathbb{1}^{\sigma(R_{PS})T} \end{array} \right] \tag{5.25}$$

$$M_{PS}^{+} = \sum_{y2=1}^{\sigma(B_{S_{PS}})} e_{y2}^{\sigma(B_{S_{PS}})} \left[\mathbb{P}\left(X_{y_2}^{+} \right)^{V} \right]^{T} \tag{5.26}$$

where

$$X_{y_2}^{+} = \left[\begin{array}{ccc} \mathbb{1}^{\sigma(P_{\mu PS})} e_{y2}^{\sigma(B_{S_{PS}})T} & | & \mathbf{0}^{\sigma(P_{\mu PS})\times\sigma(R_H)} \\ \hline \mathbb{1}^{\sigma(P_{\gamma PS})} \otimes \mathbb{1}^{\sigma(B_{S_{PS}})} & \otimes e_{y2}^{\sigma(B_{S_{PS}})} & \otimes \mathbb{1}^{\sigma(R_{PS})T} \end{array} \right] \tag{5.27}$$

To complete the production system Petri net model, capacity constraints can be applied. A capacity vector C_C of size $\sigma(\mathcal{E}_S) \times 1 \in \mathbb{N}^{\sigma(\mathcal{E}_S)}$ limits the number of tokens within a given transition.

$$Q_{\mathcal{E}}[k] \leq C_C \qquad \forall k \tag{5.28}$$

Each element of the input firing vectors is also constrained by the availability of the associated system resource and its capacity.

$$u_{\psi}[k] \cdot \left[J_S(\omega, v) \ominus K_S(\omega, v) \right] < C_C(\psi) \tag{5.29}$$

5.3.2.2 Product Petri Net

In addition to the production system Petri net, a product Petri net is required to describe the evolution of products as each evolve from raw to finished state.

Definition 5.13: Product Petri Net [2,6,20,34]: Given product l_i, a product net is a marked Petri net graph $\mathcal{N}_{l_i} = \{S_{l_i}, \mathcal{E}_{l_i}, \mathbf{M}_{l_i}, W_{l_i}, Q_{l_i}\}$ where:

- S_{l_i} is the set of product places that represents a product component at a raw, work-in-

progress, or final stage of production.

- $\mathcal{E}_{l_i}$ is the set of product events that describe the transformation of the product between product places.
- $\mathbf{M}_{l_i} \subseteq (S_{l_i} \times \mathcal{E}_{l_i}) \cup (\mathcal{E}_{l_i} \times S_{l_i})$ is the product arc relations that describe which products or components receive which product events.
- $W_{l_i} : \mathbf{M}_{l_i} \rightarrow \{0, 1\}$ is the weighting function on product arcs.
- Q_{l_i} is a marking vector that describes the product's evolution.

where a product event is defined as:

Definition 5.14: Product Event: A specific transformation process that may be applied to a given product. ■

And where the associated (untimed) state transition function follows Definition 5.10.

$$Q_{l_i}[k+1] = Q_{l_i}[k] + M_{l_i} U_{l_i}[k] \tag{5.30}$$

■

where $U_{l_i}[k]$ is the firing vector corresponding with product l_i at time k.

Note that a different product net is needed for each product type; each of which must be instantiated to support each individual product on the shop floor. This supports mass-customized production and the intelligent product paradigm. The interested reader is referred to the underlying references for detailed discussion [6, 7].

As expected, the production system Petri net and the product Petri net dynamics are inherently coupled. A product firing matrix is introduced to synchronize the production system Petri net firing vectors with those of the product Petri nets.

Definition 5.15: Product Firing Matrix [22, 25, 30]: a binary product firing matrix $\mathcal{U}[k]$ of size $\sigma(\mathcal{E}_S) \times \sigma(L)$ whose element $u_{\psi,l}[k] = 1$ when the k^{th} firing timing triggers a product

l to take structural degree of freedom ψ for action. ■

Additionally, a product transformation feasibility matrix is required for each product to link product events to their associated transformation processes in the production system.

Definition 5.16: Product Transformation Feasibility Matrix [2, 6, 22]: A binary matrix $\Lambda_{\mu i}$ of size $\sigma(\mathcal{E}_{l_i}) \times \sigma(P_\mu)$ whose value $\Lambda_{\mu i}(x, j) = 1$ iff $\epsilon_{x l_i}$ realizes transformation process $p_{\mu j}$. ■

Consequently, the production system input firing vectors at a given moment k become [2, 6, 22]:

$$\Lambda_{\mu i}^{T} U_{l_i} = A_{i,j} \cdot \mathcal{U} e_{l_i}^{\sigma(L)T} \tag{5.31}$$

where e_i^n represents the i^{th} elementary basis vector of predefined length n and $A_{i,j} = 1$ iff $p_{\mu i}$ is used in ϵ_{S_j}.

5.3.3 Power Flow Analysis Model of the Microgrid

To complete the dynamic energy management model for a microgrid-enabled production system, a power flow analysis model of the microgrid is introduced. To this end, the link between Axiomatic Design and power flow analysis has been previously established [27] and is summarized here. Six steps are taken to establish the elements of the power flow analysis.

The derivation of the power flow analysis model of the microgrid rests in the recognition that the production system acts as a load on the microgrid and must be included. The first step specifies a device model for every structural degree of freedom in J_{MPS}. In the device model, the set of algebraic state variables is defined as $w_{E\psi} = [P_\psi, Q_\psi, v_\psi, \theta_\psi]$, where

- P_ψ – is active power injection from ground.
- Q_ψ – is the associated reactive power injection from ground.
- v_ψ – is the associated voltage magnitude relative to ground.

- θ_{ψ} – is the associated voltage angle relative to a predefined reference bus.

For each structural degree of freedom ψ [27], the device model for transportation structural degrees of freedom contains a specific set of algebraic equations:

$$g_{\psi} = P_{E_{\psi}} + jQ_{\psi} = (v_{\psi}\angle\theta_{\psi})y_{\psi}^{*}(v_{\psi}\angle\theta_{\psi})^{*} \tag{5.32}$$

In the second step, for convenience, the transportation degrees of freedom $\mathcal{E}_{\psi H}$ are distinguished from the transformation degrees of freedom $\mathcal{E}_{\psi M}$. Each element in the transportation structural degrees of freedom is also assigned an admittance $y_{\psi h}$. These are organized into a transportation degree of freedom admittance matrix, which can be compared with the line admittance matrix in traditional power flow analysis [27].

$$\mathcal{Y} = diag(y_{\psi h_1},...,y_{\psi h_{\sigma(2H)}}) \tag{5.33}$$

The third step defines the transportation degree of freedom incidence matrix $M_{\mathcal{E}_{\psi H}}$, using Equations 5.4 and 5.5. The transportation degree of freedom incidence matrix can be compared with the bus incidence matrix in traditional power flow analysis [27].

$$M_{\mathcal{E}_{\psi H}} = M^{+}_{\mathcal{E}_{\psi H}} - M^{-}_{\mathcal{E}_{\psi H}} \tag{5.34}$$

where

$$M^{-}_{\mathcal{E}_{\psi H}} = \sum_{y1=1}^{\sigma(B_{S_{MPS}})} e_{y1}^{\sigma(B_{S_{MPS}})}[\mathbb{P}_H(X^{-}_{y_1})^V]^T \tag{5.35}$$

with:

$$X^{-}_{y_1} = e_{y1}^{\sigma(B_{S_{MPS}})} \otimes \mathbb{1}^{\sigma(B_{S_{MPS}})} \otimes \mathbb{1}^{\sigma(R_{MPS})T} \tag{5.36}$$

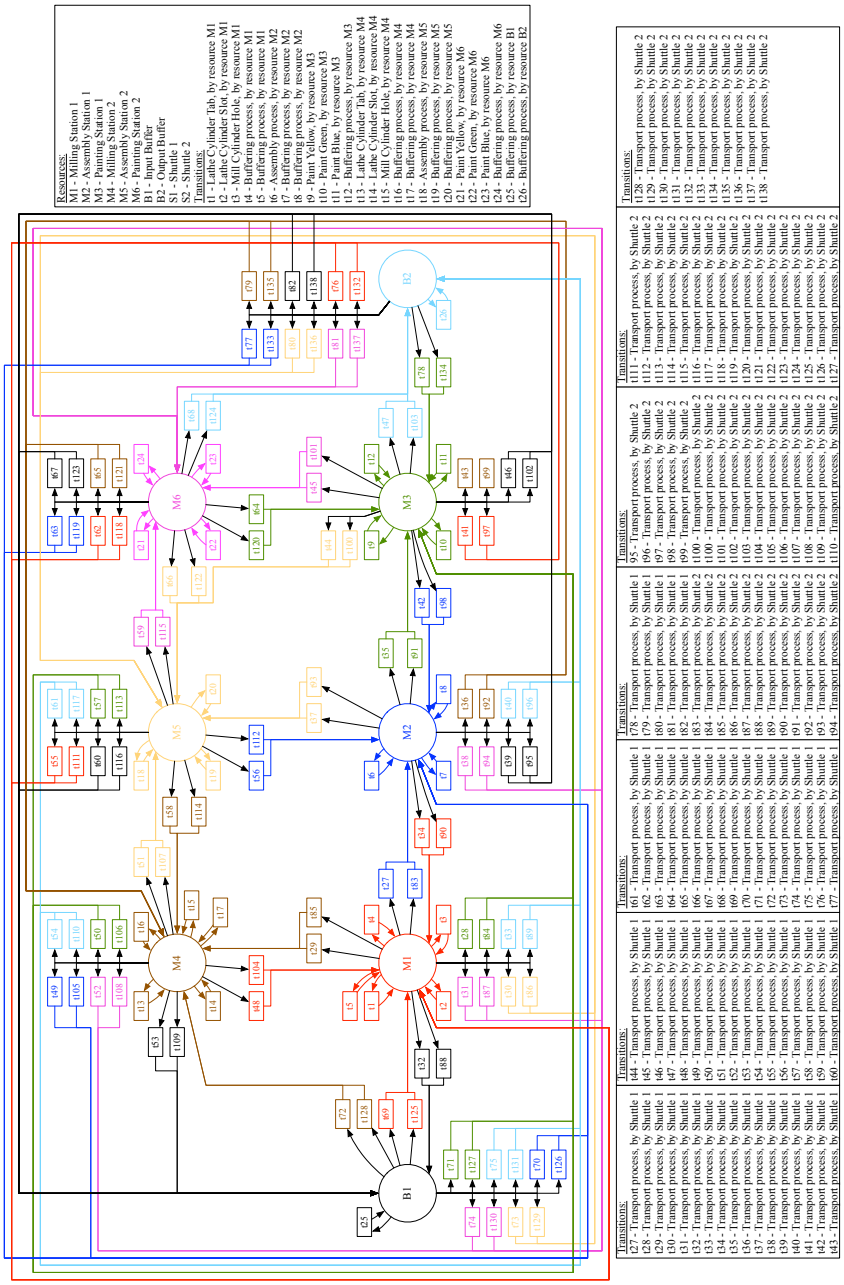


Fig. 5.2: Petri Net Graph of the Starling Manufacturing System.

and where

$$M^{+}_{\mathcal{E}_{\psi H}} = \sum_{y2=1}^{\sigma(B_{S_{MPS}})} e^{\sigma(B_{S_{MPS}})}_{y2} [\mathbb{P}_H (X^{+}_{y_2})^V]^T \tag{5.37}$$

with:

$$X^{+}_{y_2} = \mathbb{1}^{\sigma(B_{S_{MPS}})} \otimes e^{\sigma(B_{S_{MPS}})}_{y2} \otimes \mathbb{1}^{\sigma(R_{MPS})T} \tag{5.38}$$

and where

$$\mathbb{P}_H = [e^{\sigma(P_\eta)\sigma(H)}_{\psi H_1}, \ldots, e^{\sigma(P_\eta)\sigma(H)}_{\psi H_n}] \tag{5.39}$$

The fourth step combines the transportation degrees of freedom admittance matrix (Equation 5.33) and incidence matrix (Equation 5.34), to calculate the bus admittance matrix with size $\sigma(B_{S_{MPS}}) \times \sigma(B_{S_{MPS}})$, as traditionally defined in power systems engineering [27, 302].

$$\mathbf{Y} = M_{\mathcal{E}_{\psi H}} * \mathcal{Y} * M^{T}_{\mathcal{E}_{\psi H}} \tag{5.40}$$

The fifth step extends the previously established derivation of the power flow analysis as the term M_E is introduced to allow for multiple degrees of freedom in one system resource [27].

$$M_E = \sum_{y_1=1}^{\sigma(B_S)} e^{\sigma(B_S)}_{y_1} [\mathbb{P}_M (\mathbb{1}^{\sigma(P_\mu)} e^{\sigma(M)T}_{y_1})^V]^T \tag{5.41}$$

where

$$\mathbb{P}_M = [e^{\sigma(P_\mu)\sigma(M)}_{\psi M_1}, \ldots, e^{\sigma(P_\mu)\sigma(M)}_{\psi M_n}] \tag{5.42}$$

As the final step, the power flow equations follow straightforwardly from Kirchoff's Current Law [27, 302, 303].

$$M_E[\mathbf{P}_E + j\mathbf{Q}] = diag(\mathbf{V})\mathbf{Y}^*\mathbf{V}^* \tag{5.43}$$

where $\mathbf{P}$, $\mathbf{Q}$ are associated with transformation degrees of freedom. $\mathbf{V}$ has size $\sigma(B_{S_{MPS}}) \times \mathbf{1}$

and contains the nodal voltage levels, as defined in the device model. In such a way, the relatively abstract Axiomatic Design knowledge base is shown to be entirely consistent with traditional power flow analysis.

In conclusion, this section formulates a dynamic model for the energy management of microgrid-enabled production systems. The model is dynamic in the sense that it represents the production system's discrete event dynamics using Petri nets. It facilitates energy management in that the energy consumption of the production system is tightly integrated with the underlying microgrid.

5.4 Illustrative Example

This section discusses an example for a Microgrid-Enabled Production System. This specific example is chosen because it includes elements that cover all of the definitions in the microgrid-enabled production system model while still retaining enough simplicity so as to give relatively intuitive results. The production system part of the system is further detailed in Subsection 5.4.1. The microgrid part of the system is further detailed in Subsection 5.4.2.

5.4.1 Production System

The Starling Manufacturing System is a discrete part production system of bird feeders and depicted in Figure 5.3 [2, 14, 20]. It consists of ten resources; three pairs of transforming machines M_{PS} = {Machining Station 1, Assembly Station 1, Painting Station 1, Machining Station 2, Assembly Station 2, Painting Station 2}, two independent buffers B_{PS} = {Input Buffer, Output Buffer}, and two transportation resources H_{PS} = {Shuttle A, Shuttle B}. The independent buffers store product elements before and after processing in the production system. The transportation resources are shuttles that move on tracks; following the direction of the arrows (as displayed in Figure 5.3).

The bird feeders consist of three parts; a bottom, middle, and top cylinder. It is

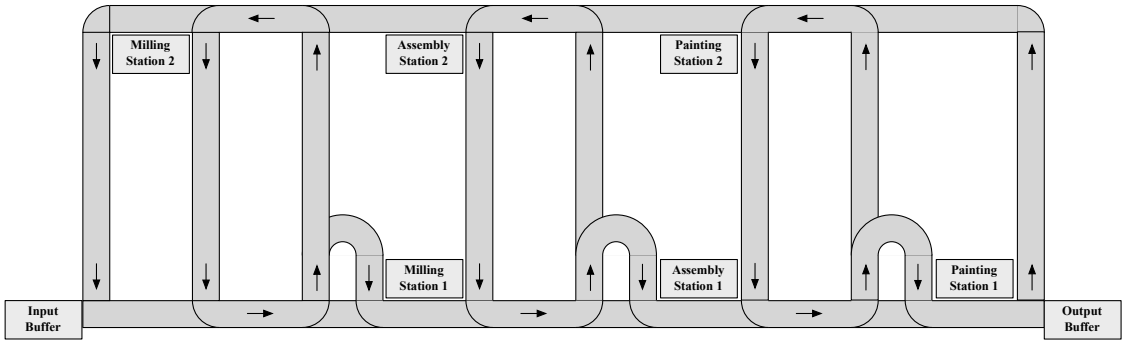


Fig. 5.3: Starling Manufacturing System Overview

produced in seven transformation processes $P_{\mu PS}$ = {Lathe Tab, Lathe Slot, Mill Hole, Laminate, Paint Yellow, Paint Green, Paint Blue}. The transportation processes are $P_{\eta PS} = \{m_i m_j, m_i b_k, b_k m_i, b_k b_l\} \forall i, j = 1, 2, 3, 4, 5, 6; k, l = 1, 2$ and move the product parts between buffers. The holding processes account for three fixture configurations $P_{\gamma PS}$ = {Small Radial, Big Radial, Axial}. The allocation of the processes onto the resources using knowledge bases is demonstrated in earlier work.

Figure 5.2 on Page 239 shows the Petri net representation of the production system shown in Figure 5.3. Note that its lay-out is similar to the physical production system lay-out. The Petri net contains places for the set of buffers B_S, that are oriented in a similar way as in Figure 5.3. Every process is related to one of the transitions. These represent the production degrees of freedom. The incidence matrix is defined by Definition 5.9, or by inspection. The transitions are fired based on a scheduled events list. This events list is imported from an external source and assumed to be constant. The fired transitions are shown as a list in Table 5.3 on Page 243.

The product net is based on the production sequence. The different stages are represented by places and the transformations equal transitions between product places. The product net for a yellow bird feeder is shown in Figure 5.4. The product net dynamics are related to the firing vectors via the Production System Feasibility Matrix (Definition 5.16). When a product part is neither processed nor buffered, it appears as a queue in the buffer state vector

Table. 5.3: Production System Firing Vectors [2]

Production System			Time		Product Net		
List Index (rows)	Transition	# Tokens Fired	T start	T end	Transition	# Tokens Fired	Name
1	25	7					Buffer
2	72	1	1	4			Transportation
3	125	1	1	4			Transportation
4	1	1	5	8	1	1	Lathe Cylinder Tab
5	13	1	5	8	4	1	Lathe Cylinder Tab
6	2	1	8	11	2	1	Lathe Cylinder Slot
7	14	1	8	11	5	1	Lathe Cylinder Slot
8	3	1	11	14	3	1	Mill Cylinder Hole
9	15	1	11	14	6	1	Mill Cylinder Hole
10	49	1	14	18			Transportation
11	83	1	14	18			Transportation
12	6	1	18	23	10	1	Assembly Process
13	72	1	18	22			Transportation
14	125	1	18	22			Transportation
15	1	1	22	25	7	1	Lathe Cylinder Tab
16	13	1	22	25	7	1	Lathe Cylinder Tab
17	7	1					Buffer
18	2	1	25	28	8	1	Lathe Cylinder Slot
19	14	1	25	28	8	1	Lathe Cylinder Slot
20	3	1	28	31	9	1	Mill Cylinder Hole
21	15	1	28	31	9	1	Mill Cylinder Hole
22	49	1	31	35			Transportation
23	83	1	31	35			Transportation
24	6	1	35	40	11	1	Assembly Process
25	7	1					Buffer
26	72	1	35	39			Transportation
27	125	1	35	39			Transportation
28	1	1	39	42	1	1	Lathe Cylinder Tab
29	13	1	39	42	4	1	Lathe Cylinder Tab
30	35	1	40	45			Transportation
31	2	1	42	45	2	1	Lathe Cylinder Slot
32	14	1	42	45	5	1	Lathe Cylinder Slot
33	3	1	45	48	3	1	Mill Cylinder Hole
34	9	1	45	51	12	1	Paint Yellow
35	15	1	45	48	6	1	Mill Cylinder Hole

Production System (cont'd)			Time (cont'd)		Product Net (cont'd)		
List Index (rows)	Transition	# Tokens Fired	T start	T end	Transition	# Tokens Fired	Name
36	49	1	48	52			Transportation
37	83	1	48	52			Transportation
38	12	1					Buffer
39	6	1	52	57	10	1	Assembly Process
40	72	1	52	56			Transportation
41	125	1	52	56			Transportation
42	1	1	56	59	1	1	Lathe Cylinder Tab
43	13	1	56	59	4	1	Lathe Cylinder Tab
44	47	1	56	60			Transportation
45	6	1	57	62	11	1	Assembly Process
46	2	1	59	62	2	1	Lathe Cylinder Slot
47	14	1	59	62	5	1	Lathe Cylinder Slot
48	26	1					Buffer
49	3	1	62	65	3	1	Mill Cylinder Hole
50	15	1	62	65	6	1	Mill Cylinder Hole
51	35	1	62	67			Transportation
52	4	1					Buffer
53	49	1	65	69			Transportation
54	10	1	67	73	12	1	Paint Green
55	83	1	67	73			Transportation
56	7	1					Buffer
57	72	1	69	73			Transportation
58	6	1	71	76	10	1	Assembly Process
59	13	1	73	76	7	1	Lathe Cylinder Tab
60	47	1	73	77			Transportation
61	7	1					Buffer
62	14	1	76	79	8	1	Lathe Cylinder Slot
63	26	1					Buffer
64	15	1	79	82	9	1	Mill Cylinder Hole
65	49	1	82	86			Transportation
66	6	1	86	91	11	1	Assembly Process
67	35	1	91	96			Transportation
68	11	1	96	102	12	1	Paint Blue
69	47	1	102	106			Transportation

Q_S. However, all places have a transition that represents the buffering process. As a result, queues only occur when the capacity of the place is exceeded. The transition state vector $Q_{\mathcal{E}}$ shows the tokens per transition. The transitions consume a predefined amount of energy, which is added to the idling consumption of the machines. Additionally, the assumption is made that the transitions related to transportation resources and independent buffers do not consume power.

5.4.2 Electric Power System

The microgrid lay-out is based on the microgrid in the book by Saadat [51]. The load buses are corresponding with the machines in the production system. The system consists of nine transforming or buffering resources $B_{S_{MG}}$ = {Load Bus 1, Load Bus 2, Load Bus 3, Load

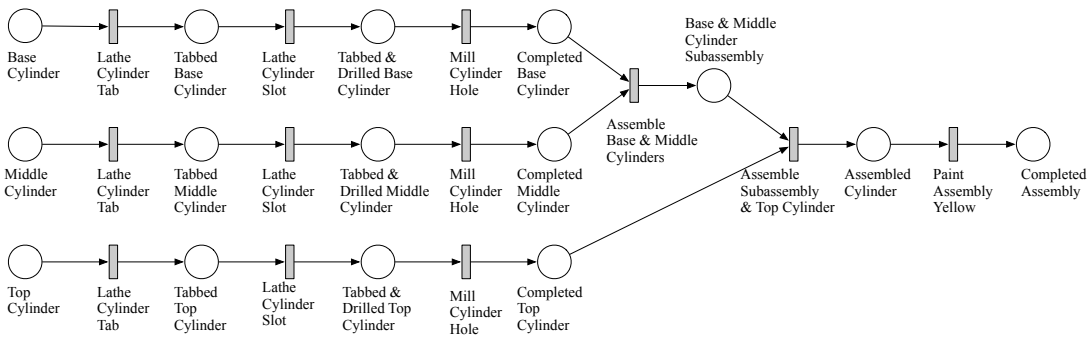


Fig. 5.4: Product net for a yellow birdfeeder [2]

Bus 4, Load Bus 5, Load Bus 6, Slack Bus, RE Generator 1, RE Generator 2}, where the Slack Bus is considered to be a dispatchable generator (i.e. a connection to the larger power grid, a battery, etc). The transportation resources are the power lines H_{MG} = {Power Line 1, … , Power Line 10}. This set of energy resources captures the breadth of functionality described in the Methodological Development (Section 5.3). The transformation processes in the system are $P_{\mu MG}$ = {Generate Power, Consume Power}. The holding processes $P_{\gamma MG}$ are assumed to be the same for every bus and every power line and can thus be neglected. The transportation processes $P_{\eta MG}$ can be derived from Figure 5.5, in which the arrows leaving the buses indicate the loads.

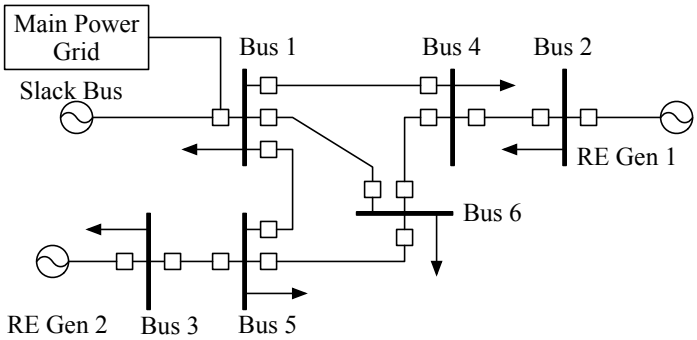


Fig. 5.5: Overview of the Microgrid structure [51]

The input data for the renewable energy generators is generated by the DOE/NREL alliance [304]. The two renewable energy resources are assumed to be solar photovoltaics;

each with a peak generation of 11.5 MW. Both the power generation and the power consumption are assumed to contain solely the active power. Figure 5.7 shows the energy generation curve of the generators independently and aggregated. The slack bus sustains the power balance in the microgrid and is allowed to be both positive and negative. The slack bus can, for example, be a combination of a connection with the power grid, to offload a generation surplus, and a gas turbine, to provide when there is a power deficit. The gas turbine, in this illustrative example, is assumed to have a heat rate of 14,692.6 BTU/kWh at minimum efficiency, and 11,302 BTU/kWh at maximum efficiency [305, 306].

This example demonstrated the breadth of function in the methodological development (Section 5.3). Future work can add greater redundancy in both the production system as well as the power grid so as to more deeply investigate control and optimization decision making algorithms.

5.5 Results & Discussion

The Microgrid-Enabled Production System model succeeds to integrate the microgrid and the production system in one model. It aggregates both system resources and processes, which results in the shared knowledge base J_{MPS}.

The system transformation resources are M_{MPS} = {Machine 1, Machine 2, Machine 3, Machine 4, Machine 5, Machine 6, Input Buffer, Output Buffer, Shuttle 1, Shuttle 2, Slack Bus, RE Generator 1, RE Generator 2}. The system transportation resources are $H_{MPS} = H_{MG}$. The system transformation processes are $P_{\mu MPS}$ = {Lathe Tab, Lathe Slot, Mill Hole, Laminate, Paint Yellow, Paint Green, Paint Blue, $m_i m_j, m_i b_k, b_k m_i, b_k b_l$, Generate Power} $\forall i, j = 1, 2, 3, 4, 5, 6; k, l = 1, 2$. The system transportation processes $P_{\eta MPS} = P_{\eta MG}$.

Table 5.3 shows the transitions of the tokens in the product Petri net over time. By inspection it can be concluded that the products evolve correctly through the product net.

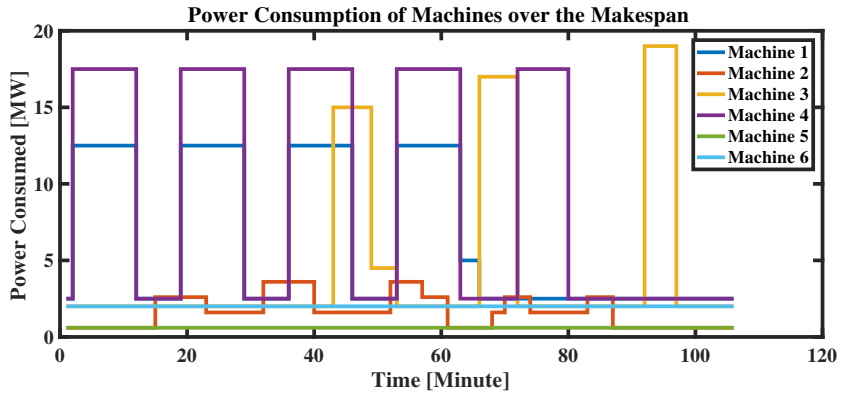


Fig. 5.6: Energy Consumption of the Machines over Time

The transitions in the product net are coupled to the transitions in the production Petri net. Figure 5.6 shows the energy consumption of the production system's machine over the duration of their production schedule of 110 minutes. The energy consumption of the machines is presented in Figure 5.6. This figure can be compared with the product net transitions of Table 5.3 and by inspection the energy consumption of the machines corresponds with the evolutions of the product parts. Additionally, the transitions have different levels of energy consumption in different machines. This explains the difference in height between the curve of Machine 1 and Machine 4. Machine 5 and Machine 6 consume a constant level of energy, which is the result of standing idle.

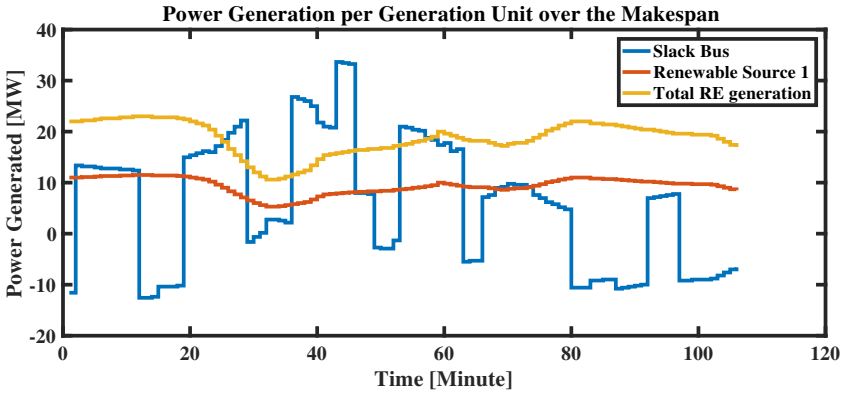


Fig. 5.7: Energy Generation over Time

Figure 5.7 displays the energy generation of the three different energy sources over time.

Two renewable energy sources (solar photovoltaics) are included. These generators have the same pattern and the two lines represent the individual pattern (RE. Gen. 1) and the aggregated pattern (RE. Gen. Total).

Both the renewable energy generation and the production system consumption are highly variable. The slack bus needs to match the netload of the system. This clearly shows the challenge of islanded operation of a microgrid-enabled production system, as the ramping speed of the slack bus needs to be large. To resolve this control problem, there is a clear need to decrease the variability of the production system energy consumption.

Under the assumption that the gas turbine produces the power demand that is not provided by the renewable energy resources, the CO_2 emissions can be calculated with the provided heat rate data. The heat rate decreases with more efficient operation of the gas turbine and is thus modeled as a linear function of the relative load on the turbine. The CO_2 emissions of the gas turbine are 53.07 Kilogram per 1 million BTU natural gas fed into the turbine. The CO_2 are calculated as:

$$CO_2 \text{ emissions} = \begin{pmatrix} \text{Carbon Intensity} \\ \text{of Fuel } [CO_2/BTU] \end{pmatrix} \begin{pmatrix} \text{Heat Rate} \\ [BTU/MWh] \end{pmatrix} \begin{pmatrix} \text{Electrical Energy} \\ \text{Generated } [MWh] \end{pmatrix} \tag{5.44}$$

The resulting rate of CO_2 emissions is presented in Figure 5.8.

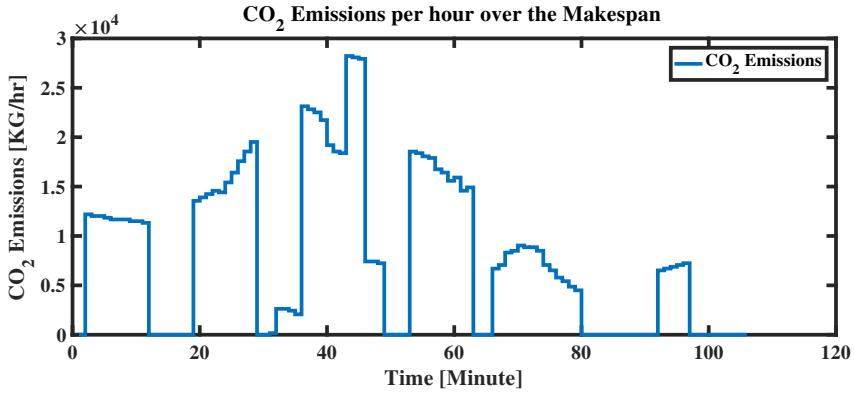


Fig. 5.8: CO_2 emissions from the Gas Turbine over Time

Generally speaking, the microgrid-enabled production system model shows that the

production activities can be directly coupled as a load to the microgrid. The two unlike systems come together in a model that addresses the characteristics of each of the systems. The model shows that a production system that is optimized solely for production, may impose an unwanted load on the microgrid. Consequently, grid balancing activities would need to address this variable behavior; as demonstrated in Figure 5.7. The determination of the firing vectors can therefore be viewed as an opportunity for decision-making that dynamically manages production activities with the energy management of the microgrid. Finally, the model can be used to create a better insight in the detailed dynamics of the microgrid (e.g. transient stability analysis) as has been demonstrated in [27].

5.6 Conclusion & Future Work

This chapter defines a dynamic model for the energy management of microgrid-enabled production systems. It succeeds in integrating a production system model with a power grid model, using Axiomatic Design for Large Flexible Engineering Systems. A mutually exclusive and collectively exhaustive description of processes and resources is introduced, which form the basis of precise energy accounting and accurate grid operations. The power flow analysis analyzes the static state of the power system. The outcomes of the example show that the variable electricity demand imposed by production activities is ill accommodated by the renewable energy sources alone. The slack bus is necessary to maintain the balance on the microgrid. Additionally, this model can function to calculate requirements for microgrid-enabled production facilities, more specifically energy storage and demand response. In future work, this model may be used for energy management decisions using the firing vectors that control the production system processes. Different system set ups can be compared on their electricity consumption and their microgrid impacts.

Chapter Summary:

This chapter leverages the hetero-functional graph theory structural model as a foundation for the development of a dynamic microgrid-enabled production system model. The dynamic model is constructed through imposing the device models on their associated degrees of freedom. This model is able to describe the product and power flow through a microgrid-enabled production system of arbitrary size, topology, and coupling. Finally, the modeling approach is demonstrated on a test case. The test case is simulated and calculates carbon emissions of the system as a proxy of sustainability. ■

Chapter 6

Optimization of Dynamic Systems with Hetero-functional Graph Theory

Chapter Abstract:

This chapter develops the first hetero-functional network minimum cost flow optimization program for engineering systems. This chapter builds on the hetero-functional graph theory structural model, as developed in Chapter 3, and the hetero-functional graph theory-based dynamic model, as developed in Chapter 5. The work in this chapter has yet to be submitted for publication.

The chapter starts by establishing two new elements in support of dynamic system models based on hetero-functional graph theory. First, the relationship between the hetero-functional incidence tensor and the field of Petri nets is established. Second, the device model matrix is defined that describes the device models for the system processes.

The chapter then defines the transformation of the Petri net-based dynamic model to the hetero-functional network minimum cost flow program. This program adheres to the quadratic program canonical form. As a result of the Petri net and hetero-functional graph theory foundations, the program successfully accommodates: the explicit definition of time, the optimization of flow for multiple operands, the transformation of one operand into

another, the storage of operands, and the description of the state of both the engineering system and the operands. The program enables optimization of systems of arbitrary size, topology, and operand types, as long as the device models can be approximated as linear relationships.

The chapter demonstrates the use of the optimization program by modeling and simulation of a hydrogen-natural gas test case. This is the first hydrogen-natural gas test case to the knowledge of the author. The optimization is performed such that the trade-offs and interdependencies of the two systems become apparent.

6.1 Introduction

Over the past decades, engineering systems have developed as networks of systems that deliver multiple services across multiple domains [307]. Examples of such socio-technical systems are the electrified transportation system [30, 308, 309], the energy-water nexus [296, 310–312], and the multi-modal energy system [313]. These systems have become increasingly interdependent across domains as a result of market forces and the associated pursuit of efficiency and cost reductions [5]. For example, the New England electric power grid relies more than ever on natural gas for its electricity generation, whereas the same natural gas is also needed to heat homes in the winter.

The interdependence of engineering system services has lead to a need for a better understanding of the holistic dynamics and trade-offs in these systems [241, 307]. Modeling tools can support the pursuit for more insight into engineering system and their optimal control. These tools need to be quantitative, represent the heterogeneity of the modeled system, and be generalizable across domains [48].

Existing optimization methods are generally based on conventional graph theoretic approaches, or on discipline and application specific dynamic models. Minimum cost

flow programs, for example, are based on networks [93] and consequently fail to address heterogeneity of function. The multilayer networks community has aimed to expand graph theory to accommodate heterogeneity of function [314], but Kivelä et. al. have identified eight modeling limitations to the types of systems that can be modeled with multi-layer networks [95]. Consequently, optimization programs based on those foundations inherently impose those same limitations. A graph-based approach was also used in the multi-commodity network flow optimization programs [315–317]. This approach does implement a notion of heterogeneity of function, but it does not integrate a specific description of operand state or storage in its program. Finally, approaches that optimize discipline or application specific programs lack generalizability [48].

Hetero-functional Graph Theory provides a rigorous modeling method that does not impose the previously mentioned modeling limitations of multilayer networks [5]. Furthermore, hetero-functional graph theory has been used in a variety of engineering system applications, to define both structural [2, 5, 14, 15, 22, 28, 44] and dynamic models [30–35]. However, hetero-functional graph theory has not been used as a foundation to an optimization program. This work proposes the first hetero-functional graph theory-based optimization program, entitled: the Hetero-functional Network Minimum Cost Flow Program.

6.1.1 Original Contribution

This work intends to define the first hetero-functional network minimum cost flow optimization program. This entails that the optimization program balances supply and demand of multiple types of operands at distinct locations over time. The problem is solved as a linearly constrained, convex quadratic program. The program can be applied to a wide variety of unlike application domains, as the operands may be transformed, assembled, and disjoined.

In the process of developing the hetero-functional network minimum cost flow optimization program, this work also establishes the first formal connection between the hetero-functional incidence tensor, arc-constant colored Petri nets, and the engineering

system net. Furthermore, it establishes the first integration of device models to the system service feasibility matrices that couple the engineering system net dynamics to the operand behavior.

Finally, this work demonstrates the hetero-functional network minimum cost flow optimization program by optimizing the first hydrogen-natural gas infrastructure test case.

6.1.2 Outline

The background (Sec. 6.2) provides an introduction to Hetero-functional Graph Theory and Petri nets. The former is used as the structural backbone of the structural model, and the latter is used as a foundation to describe the system's dynamics. Sec. 6.3 introduces the hetero-functional graph based dynamic model that incorporates device models. Sec. 6.4 then defines the hetero-functional network minimum cost flow optimization program. Sec. 6.5 introduces a hydrogen-natural gas networked infrastructure test case as an example engineering system. This test case is modeled and optimized in Sec. 6.6. Sec. 6.6 presents the hetero-functional graph model, the minimum cost flow optimization program, and the outcomes of the optimization program for the specified test case. Finally, Sec. 6.7 concludes the work and recaps the main contributions of the work to the literature.

6.2 Background

Hetero-functional Graph Theory (HFGT) was introduced over a decade ago for the study of reconfigurability of manufacturing systems [2, 6, 7, 20] and has since been applied to a number of large flexible engineering systems including electric power grids, water systems, transportation systems, healthcare, and interdependent infrastructures. Schoonenberg et al. [5] have produced a consolidating text on Hetero-functional Graph Theory, which has been further extended to include a tensor-based formulation [52]. Hetero-functional graph theory introduces a large number of modeling constructs that are not found in "traditional"

graph theory [5, 52]. Therefore, in order to maintain the self-contained nature of this paper many of the prerequisite terms are defined here for the reader's convenience and will serve as the basis for developing the hetero-functional network dynamics in Sec. 6.3 and the hetero-functional network minimum cost flow in Sec. 6.4. This section also introduces several relevant definitions from the Petri net literature [249, 318]. More specifically timed arc-constant colored Petri nets serve as an intermediate modeling vehicle that facilitates the transformation of a hetero-functional graph into hetero-functional network minimum cost flow optimization program.

This section starts with an overview of the System Concept in Hetero-functional Graph Theory in Sec. 6.2.1. After which, it continues to discuss the hetero-functional incidence tensor in Sec. 6.2.2. Sec. 6.2.3 then covers Timed Petri nets that are used in Sec. 6.2.4 as a foundation for the Hetero-functional Graph Theory Service Model. Sec. 6.2.5 introduces mathematical foundations for multi-sets (i.e. bags) which is required for the introduction of Arc-Constant Colored Petri nets in Sec. 6.2.6.

6.2.1 Hetero-functional Graph Theory: System Concept

The first hetero-functional graph theory modeling construct is the system concept.

Definition 6.1 – System Concept [2, 6, 14, 17, 20, 98]: A binary matrix A_S of size $\sigma(P) \times \sigma(R)$ whose element $A_S(w, v) \in \{0, 1\}$ is equal to one when action $e_{wv} \in \mathcal{E}_S$ (in the SysML sense) is available as a system process $p_w \in P$ being executed by a resource $r_v \in R$ The $\sigma()$ notation is used return the size of a set. ■

In other words, the system concept forms a bipartite graph between the set of system processes and the set of system resources [14]. The definition of the system concept relies on several other definitions: system resource, system process, and system operand.

Definition 6.2 – System Resource [104]: An asset or object $r_v \in R$ that is utilized during the execution of a process. ■

Definition 6.3 – System Process [104, 239]: An activity $p_w \in P$ that transforms a predefined

set of input operands into a predefined set of outputs. ■

Definition 6.4 – System Operand [104]: An asset or object $l_i \in L$ that is operated on or consumed during the execution of a process. They are the inputs and outputs of systems processes and “move" through the system. ■

It is important to recognize the system resources are classified into three categories. $R = M \cup B \cup H$, where M is the set of transformation resources, B is the set of independent buffers, and H is the set of transportation processes. Furthermore, the system buffers $B_S = M \cup B$ are introduced as well. Fig. 6.1 shows this classification as a SysML block diagram. Similarly, the system processes are classified as well. $P = P_\mu \cup P_{\bar{\eta}}$, where P_μ is the set of transformation processes, and $P_{\bar{\eta}} = P_\gamma \times P_\eta$ is the set of refined transportation processes, and where $\times$ is the Cartesian product. Fig. 6.2 shows the flow of system processes as an activity diagram [5].

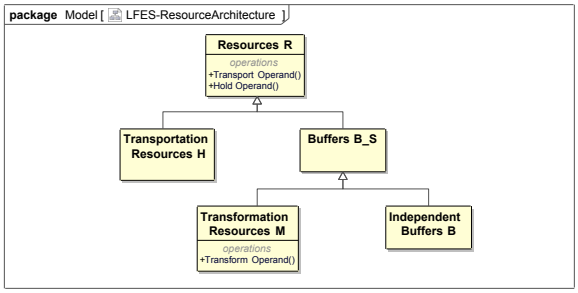


Fig. 6.1: A SysML Block Diagram: the meta-architecture of the allocated architecture of an LFES from a system form perspective [5].

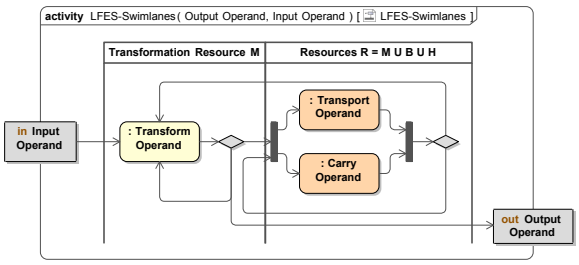


Fig. 6.2: A SysML Activity Diagram with swim lanes: the meta-architecture of the allocated architecture of an LFES from a system function perspective [5].

Finally, HFGT makes extensive use of the total number of degrees of freedom (or system

capabilities) DOF_S.

$$DOF_S = \sigma(\mathcal{E}_S) = \sum_{w}^{\sigma(P)} \sum_{v}^{\sigma(R)} A_S(w, v) \tag{6.1}$$

6.2.2 Hetero-functional Graph Theory: Incidence Tensor

The second hetero-functional graph theory modeling construct is the hetero-functional incidence tensor $\widetilde{\mathcal{M}_\rho}$ [52]. It defines the structural relationship between the system capabilities $\mathcal{E}_S$, the system operands L, and the system buffers B_S.

$$\widetilde{\mathcal{M}_\rho} = \widetilde{\mathcal{M}_\rho^+} - \widetilde{\mathcal{M}_\rho^-} \tag{6.2}$$

Definition 6.5 – The Negative 3^{rd} Order Hetero-functional Incidence Tensor $\widetilde{\mathcal{M}_\rho^-}$ [52]: The negative hetero-functional incidence tensor $\widetilde{\mathcal{M}_\rho}^- \in \{0,1\}^{\sigma(L)\times\sigma(B_S)\times\sigma(\mathcal{E}_S)}$ is a third-order tensor whose element $\widetilde{\mathcal{M}_\rho^-}(i, y, \psi) = 1$ when the system capability $\epsilon_\psi \in \mathcal{E}_S$ pulls operand $l_i \in L$ from buffer $b_{s_y} \in B_S$. ■

Definition 6.6 – The Positive 3^{rd} Order Hetero-functional Incidence Tensor $\widetilde{\mathcal{M}_\rho^+}$ [52]: The positive hetero-functional incidence tensor $\widetilde{\mathcal{M}_\rho^+} \in \{0,1\}^{\sigma(L)\times\sigma(B_S)\times\sigma(\mathcal{E}_S)}$ is a third-order tensor whose element $\widetilde{\mathcal{M}_\rho^+}(i, y, \psi) = 1$ when the system capability $\epsilon_\psi \in \mathcal{E}_S$ injects operand $l_i \in L$ into buffer $b_{s_y} \in B_S$. ■

These definitions can be used directly to determine the non-zero elements of the respective incidence tensor. Alternatively, Farid et. al. have provided a method for their calculation from more fundamental hetero-functional graph theory concepts [52].

The development of the hetero-functional network minimum cost flow optimization program requires the matricization (or “flattening") of the hetero-functional incidence tensor into a hetero-functional incidence tensor where the operand (i.e. first), and the buffer (i.e.

second) dimension are combined. The matricization function $\mathcal{F}_M()$ is adopted from [52].

$$\widetilde{M}_\rho = \mathcal{F}_M\left(\widetilde{\mathcal{M}}_\rho, [1,2], [3]\right) \tag{6.3}$$

$$\widetilde{M}^-_\rho = \mathcal{F}_M\left(\widetilde{\mathcal{M}}^-_\rho, [1,2], [3]\right) \tag{6.4}$$

$$\widetilde{M}^+_\rho = \mathcal{F}_M\left(\widetilde{\mathcal{M}}^+_\rho, [1,2], [3]\right) \tag{6.5}$$

6.2.3 Timed Petri nets

As mentioned previously, timed Petri nets serve as an intermediate modeling vehicle that facilitates the transformation of a hetero-functional graph into a hetero-functional network minimum cost flow optimization program.

Definition 6.7 – Continuous Marked Place-Transition Net (Graph [249, 319]): A bipartite directed graph represented as a 5-tuple $\mathcal{N} = \{S, \mathcal{E}, \mathbf{M}, W, Q\}$, where

- $\mathcal{N}$ is the place-transition net.
- S is a finite set of places.
- $\mathcal{E}$ is a finite set of (instantaneous) transitions, such that $B \cap \mathcal{E} = \emptyset$ and $S \cup \mathcal{E} \neq \emptyset$.
- $\mathbf{M} \subseteq (S \times \mathcal{E}) \cup (\mathcal{E} \times S)$ is a set of arcs of size $\sigma(\mathbf{M})$ from places to transitions and from transitions to places in the graph. Furthermore, defined are the associated incidence matrix $M = M^+ - M^-$ where the positive incidence matrix has element $M^+(s,e) \in \{0,1\}$ and the negative incidence matrix has element $M^-(s,e) \in \{0,1\}$ for all $(s,e) \in S \times \mathcal{E}$.
- $W : \mathbf{M} \to \mathbb{R}$, is the set of weights on the arcs.
- $Q : S \cup \mathcal{E} \to \mathbb{R}$ is the marking of the place-transition net states.

The definition of the weights W and the markings Q over the real numbers gives the Petri net its continuous rather than discrete nature. ■

Definition 6.8 – Timed Place-Transition Net Dynamics [249]: Given a binary input firing vector $U^-[k]$ and a binary output firing vector $U^+[k]$ both of size $\sigma(\mathcal{E}) \times 1$, and the positive and negative components M^+ and M^- of the Petri net incidence matrix of size $\sigma(S) \times \sigma(\mathcal{E})$, the evolution of the marking vector $Q \in \mathbb{R}^{\sigma(S)+\sigma(\mathcal{E})}$ is given by the state transition function $\Phi_T(Q[k], U^-[k], U^+[k])$:

$$Q[k+1] = \Phi_T(Q[k], U^-[k], U^+[k]) \tag{6.6}$$

where $Q = [Q_B; Q_\mathcal{E}]$ and

$$Q_B[k+1] = Q_B[k] + M^+U^+[k] - M^-U^-[k] \tag{6.7}$$

$$Q_\mathcal{E}[k+1] = Q_\mathcal{E}[k] - U^+[k] + U^-[k] \tag{6.8}$$

$$U^+_\psi[k + k_{d\psi}] = U^-_\psi[k] \tag{6.9}$$

and where $U^-_\psi[k]$ indicates the ψ^{th} element of the $U^-[k]$ vector and Eq. 6.9 allows for a transition duration of $k_{d\psi}$ between the negative and positive firing vectors. ■

6.2.4 Hetero-functional Graph Theory: Service Model

The third hetero-functional graph theory modeling construct utilizes Defn. 6.7 and is called the service model. It describes the collective behavior of operands in an engineering system. It is composed of one service Petri net and one service feasibility matrix for each operand.

Definition 6.9 – Service Petri Net [10, 11, 14, 35, 44]: Given service l_i, a service net $\mathcal{N}_{l_i}$ is marked place-transition net where

$$\mathcal{N}_{l_i} = \{S_{l_i}, \mathcal{E}_{l_i}, \mathbf{M}_{l_i}, W_{l_i}, Q_{l_i}\} \tag{6.10}$$

where

- S_{l_i} is the set of places describing a set of service states.
- $\mathcal{E}_{l_i}$ is the set of transitions describing service activities.
- $\mathbf{M}_{l_i} \subseteq (S_{l_i} \times \mathcal{E}_{l_i}) \cup (\mathcal{E}_{l_i} \times S_{l_i})$ is the set of arcs describing the relations of (service states to service activities) and (service activities to service states). Furthermore, defined are the associated incidence matrix $M_{l_i} = M^+_{l_i} - M^-_{l_i}$ where the positive incidence matrix has element $M^+_{l_i}(s_{\zeta l_i}, e_{xl_i}) \in \{0, 1\}$ and the negative incidence matrix has element $M^-_{l_i}(s_{\zeta l_i}, e_{xl_i}) \in \{0, 1\}$ for all $(s_{\zeta l_i}, e_{xl_i}) \in S_{l_i} \times \mathcal{E}_{l_i}$.
- $W_{l_i} : \mathbf{M}_{l_i} \rightarrow [0 \ldots 1]$ is the set of weights on the arcs describing the service transition probabilities for the arcs.
- Q_{l_i} is the Petri net marking representing the set of service states.

■

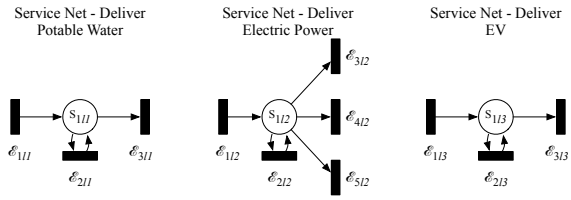


Fig. 6.3: Three service nets. One for each operand (a) Water, (b) Power, and (c) Electric Vehicle [5].

Fig. 6.3 displays a service net for three operands. The places track the operand state, and the transitions evolve the state of the operand. Furthermore, the transitions can “create" or “destroy" operands, by transitions that do not have an origin or destination respectively.

Service Petri nets have the following dynamics:

Definition 6.10 – Service Net Dynamics [249]: Given a binary input firing vector $U^+_{l_i}[k]$ and a binary output firing vector $U^-_{l_i}[k]$ both of size $\sigma(\mathcal{E}_{l_i}) \times 1$, and the positive and negative components $M^+_{l_i}$ and $M^-_{l_i}$ of the Petri net incidence matrix of size $\sigma(S_{l_i}) \times \sigma(\mathcal{E}_{l_i})$, the evolution

of the marking vector Q_{l_i} is given by the state transition function $\Phi_{l_i}(Q_{l_i}[k], U_{l_i}^-[k], U_{l_i}^+[k])$:

$$Q_{l_i}[k+1] = \Phi_{l_i}(Q_{l_i}[k], U_{l_i}^-[k], U_{l_i}^+[k]) \tag{6.11}$$

where $Q_{l_i} = [Q_{Sl_i}; Q_{\mathcal{E}l_i}]$ and

$$Q_{Sl_i}[k+1] = Q_{Sl_i}[k] + M_{l_i}^+ U_{l_i}^+[k] - M_{l_i}^- U_{l_i}^-[k] \tag{6.12}$$

$$Q_{\mathcal{E}l_i}[k+1] = Q_{\mathcal{E}l_i}[k] - U_{l_i}^+[k] + U_{l_i}^-[k] \tag{6.13}$$

The duration of the service net transitions is discussed specifically in Sec. 6.3.3. ■

In addition to the service petri net, the hetero-functional graph theory service model includes the service feasibility matrix.

Definition 6.11 – Service-Capability Feasibility Matrix [52]: For a given service l_i, a binary matrix of size $\sigma(\mathcal{E}_{l_i}) \times \sigma(\mathcal{E}_S)$ whose value $\widetilde{\Lambda}_i(x, \psi) = 1$ if e_{xl_i} realizes capability $e_{s\psi}$. Furthermore:

$$\widetilde{\Lambda}_i = \widetilde{\Lambda}_i^+ \oplus \widetilde{\Lambda}_i^- \tag{6.14}$$

such that $\widetilde{\Lambda}_i^+$ enables capability $e_{s\psi}$ to generate the output l_i and $\widetilde{\Lambda}_i^-$ enables capability $e_{s\psi}$ to use l_i as its input. ■

The service feasibility matrix couples the operand behavior to the hetero-functional graph theory incidence tensor.

6.2.5 Multi-sets

In order to discuss arc-constant colored Petri nets in the next subsection, a mathematical foundation for multi-sets is introduced here.

Definition 6.12 – Multi-set or Bag [318]: A multi-set m, over a non-empty set $\mathcal{S}$, is a function of $m \in [\mathcal{S} \to \mathbb{N}]$. The non-negative integer $m(s) \in \mathbb{N}$ is the number of appearances

of the element s in the multi-set m. The multi-set m is represented by a formal sum:

$$\sum_{s \in \mathcal{S}} m(s)'s \tag{6.15}$$

$\mathcal{S}_{MS}$ denotes the set of all multi-sets over $\mathcal{S}$. The non-negative integers $\{m(s) \mid s \in \mathcal{S}\}$ are called the coefficients of the multi-set m, and $m(s)$ is called the coefficient of s. An element $s \in \mathcal{S}$ is said to belong to the multi-set m iff $m(s) \neq 0$, and thus $s \in m$. ■

In this work, this multi-set definition is relaxed so that $m(s) \in \mathbb{R}^+$ to allow for fractional members of a set. Finally, multi-sets admit arithmetic operations as expected.

$$m_1 + m_2 = \sum_{s \in \mathcal{S}} (m_1(s) + m_2(s))'s \tag{6.16}$$

$$m_2 - m_1 = \sum_{s \in \mathcal{S}} (m_2(s) - m_1(s))'s \tag{6.17}$$

$$n * m = \sum_{s \in \mathcal{S}} (n * m(s))'s \tag{6.18}$$

where $m_1, m_2 \in S_{MS}$ and all $n \in \mathbb{R}^+$.

6.2.6 Arc-Constant Colored Petri Nets

In addition to timed place-transition nets, arc-constant colored Petri nets (ac-CPN) serve as an intermediate modeling vehicle that facilitates the transformation of a hetero-functional graph into a hetero-functional network minimum cost flow optimization program. More specifically, ac-CPNs are used to introduce operand heterogeneity to the Petri net logic.

Definition 6.13 – Arc-constant colored Petri net (ac-CPN [249, 319]): An arc-constant colored Petri net $\mathcal{N}_\mathcal{C}$ is defined by a tuple $\mathcal{N}_\mathcal{C} = \{S_\mathcal{C}, \mathcal{E}_\mathcal{C}, \mathbf{M}_\mathcal{C}, \mathcal{C}, cd, Q_\mathcal{C}\}$, where

- $S_\mathcal{C}$ is a finite set of places,
- $\mathcal{E}_\mathcal{C}$ is a finite set of transitions disjoint from $S_\mathcal{C}$,

- $\mathbf{M}_{\mathcal{C}} \subseteq (S_{\mathcal{C}} \times \mathcal{E}_{\mathcal{C}}) \cup (\mathcal{E}_{\mathcal{C}} \times S_{\mathcal{C}})$. The associated incidence matrix $M_{\mathcal{C}} = M_{\mathcal{C}}^{+} - M_{\mathcal{C}}^{-}$ where the positive incidence matrix $M_{\mathcal{C}}^{+} \in \mathcal{B}^{|S_{\mathcal{C}}| \times |\mathcal{E}_{\mathcal{C}}|}$ has element $M_{\mathcal{C}}^{+}(s_c, e_c) \in \text{Bag}(cd(s_c))$ and the negative incidence matrix $M_{\mathcal{C}}^{-} \in \mathcal{B}^{|S_{\mathcal{C}}| \times |\mathcal{E}_{\mathcal{C}}|}$ has element $M_{\mathcal{C}}^{-}(s_c, e_c) \in \text{Bag}(cd(s_c))$ for all $(s_c, e_c) \in S_{\mathcal{C}} \times \mathcal{E}_{\mathcal{C}}$.
- $\mathcal{C}$ is the set of color classes.
- $cd : S_{\mathcal{C}} \rightarrow \mathcal{C}$ is the color domain mapping.
- $Q_{\mathcal{C}} \in \text{Bag}(cd(s_c))$ is the marking vector of the arc-constant Colored Petri Net states. It is equal in size to the number of places.

Note that $\mathcal{B} = \text{Bag}(A)$, where A is the union of all color sets $\mathcal{C}$. Furthermore, the difference operator in $M_{\mathcal{C}} = M_{\mathcal{C}}^{+} - M_{\mathcal{C}}^{-}$ follows Eq. 6.17. Finally, in comparison to the Place-Transition Net, the arc weights of an ac-CPN are integrated into the incidence matrices directly and the marking of the net is now over $\text{Bag}(cd(s_c))$ instead of over the set of positive real numbers. ■

Definition 6.14 – Arc-Constant Colored Petri Net State Transition Function $\Phi_{\mathcal{C}}()$:

$$Q_{\mathcal{C}}[k+1] = \Phi_{\mathcal{C}}(Q_{\mathcal{C}}[k], U_{\mathcal{C}}^{-}[k], U_{\mathcal{C}}^{+}[k]) \quad \forall k \in \{1, \ldots, K\} \tag{6.19}$$

where $Q_{\mathcal{C}} = [Q_{B\mathcal{C}}; Q_{\mathcal{E}\mathcal{C}}]$ and

$$Q_{B\mathcal{C}}[k+1] = Q_{B\mathcal{C}}[k] + M_{\mathcal{C}}^{+} U_{\mathcal{C}}^{+}[k] - M_{\mathcal{C}}^{-} U_{\mathcal{C}}^{-}[k] \tag{6.20}$$

$$Q_{\mathcal{E}\mathcal{C}}[k+1] = Q_{\mathcal{E}\mathcal{C}}[k] - U_{\mathcal{C}}^{+}[k] + U_{\mathcal{C}}^{-}[k] \tag{6.21}$$

$$U_{\mathcal{C}\psi}^{+}[k + k_{d\psi}] = U_{\mathcal{C}\psi}^{-}[k] \tag{6.22}$$

$U_{\mathcal{C}\psi}^{-}[k]$ indicates the ψ^{th} element of the $U_{\mathcal{C}}^{-}[k]$ vector and Eq. 6.22 allows for a transition duration of $k_{d\psi}$ between the negative and positive firing vectors. ■

While ac-CPNs are valuable tool for modeling, verification, and visualization, they must be transformed into place-transition nets prior to their use in an optimization setting. Jensen has defined the steps necessary for such a transformation [318]; which is summarized here using a tensor-based treatment.

Algorithm 6.1 – Conversion from an ac-CPN to a PN:

Input: $\mathcal{N}_{\mathcal{C}} = \{S_C, \mathcal{E}_C, \mathbf{M}_{\mathcal{C}}, \mathcal{C}, cd, Q_{\mathcal{C}}\}$

Output: $\mathcal{N} = \{S, \mathcal{E}, \mathbf{M}, W, Q\}$

1. Split the places of the ac-CPN for each color set. $S = \mathcal{C} \bigtimes S_C$.

2. Retain the transitions of the ac-CPN. $\mathcal{E} = \mathcal{E}_C$.

3. Redefine the multi-set negative incidence matrix $M^-_{\mathcal{C}}$ as a third-order negative incidence tensor $\mathcal{M}^-_{\mathcal{C}}$ where $\mathcal{M}^-_{\mathcal{C}}(c, s_c, e_c) = M^-_{\mathcal{C}}(s_c, e_c)'c$. Matricize this tensor along the first two dimensions. $M^- = \mathcal{F}_M\left(\mathcal{M}^-_{\mathcal{C}}, [1,2], [3]\right)$.

4. Redefine the multi-set positive incidence matrix $M^+_{\mathcal{C}}$ as a third-order negative incidence tensor $\mathcal{M}^+_{\mathcal{C}}$ where $\mathcal{M}^+_{\mathcal{C}}(c, s_c, e_c) = M^+_{\mathcal{C}}(s_c, e_c)'c$. Matricize this tensor along the first two dimensions. $M^+ = \mathcal{F}_M\left(\mathcal{M}^+_{\mathcal{C}}, [1,2], [3]\right)$.

5. Redefine the initial multi-set marking vector $Q_{B\mathcal{C}}[0]$ as a matrix $\mathcal{Q}_{B\mathcal{C}}[0]$ where $\mathcal{Q}_{B\mathcal{C}}(c, s_c)[0] = Q_{B\mathcal{C}}(s_c)'c[0]$. The vectorize this matrix. $Q_B[0] = vec(\mathcal{Q}_{B\mathcal{C}}[0])$.

6. Retain the initial conditions of the ac-CPN transitions. $Q_{\mathcal{E}}[0] = \mathcal{Q}_{\mathcal{E}C}[0]$.

■

6.3 Hetero-functional Network Dynamics

Given the foundation of hetero-functional graph theory and Petri-net definitions provided above, this paper now derives the Hetero-functional Network Dynamics. The dynamic model consists of three parts: (1) the Engineering System Net, which represents the dynamics of the engineering system, (2) the Service Net, which represents the dynamics of the system

operands, and (3) the Synchronization Matrix, which couples the operand behavior to the engineering system net behavior. The hetero-functional network dynamics are modeled in discrete time. Continuous time dynamics may be discretized into discrete-time [131] and discrete-event dynamics can be given a system clock and scheduled event list [129] to recover discrete-time dynamics. The three parts of the hetero-functional network dynamics are now discussed in sequence.

6.3.1 Engineering System Net

The engineering system net describes the dynamics of the engineering system.

Definition 6.15 – Engineering System Net: An arc-constant colored Petri net $\mathcal{N}_C = \{B_S, \mathcal{E}_S, \mathbf{M}_C, L, cd, Q\}$, where

- B_S system buffers are the set of places,
- $\mathcal{E}_S$ system capabilities are the set of transitions (disjoint from B_S),
- $\mathbf{M}_C \subseteq (B_S \times \mathcal{E}_S) \cup (\mathcal{E}_S \times B_S)$. The associated incidence matrix $M_C = M_C^+ - M_C^-$ such that

$$M_C^-(y,\psi) = \sum_{l_i \in L} \left(\widetilde{\mathcal{M}}_\rho^-(l_i, y, \psi)\right)' l_i \quad \in \{l_1, \ldots, l_{\sigma(L)}\} \tag{6.23}$$

$$M_C^+(y,\psi) = \sum_{l_i \in L} \left(\widetilde{\mathcal{M}}_\rho^+(l_i, y, \psi)\right)' l_i \quad \in \{l_1, \ldots, l_{\sigma(L)}\} \tag{6.24}$$

- L (system operands) are the set of color classes.
- $cd : B_S \to L$ is the color domain mapping.
- $Q \in \text{Bag}(cd(s))$ is the marking vector of the engineering system net. It represents the state of the engineering system.

■

Here, it is important to recognize that the positive and negative hetero-functional incidence tensors indicate the presence of “colored" arcs in the arc-constant colored Petri net. Consequently, the hetero-functional incidence tensor can be used to straightforwardly recover the engineering systems behavior via the arc-constant colored Petri net state transition function $\Phi_{\mathcal{C}}()$ (Defn. 6.14). Furthermore, from a physics perspective, the engineering system net as defined above imposes continuity laws for all colored-operands at all system buffers. Finally, this engineering system definition provided is a generalization of the one used in prior hetero-functional graph theory work for transportation systems [23–25], electrified transportation systems [30–33], production systems [2, 6, 7, 14, 15, 19, 20, 22], and microgrid-enabled production systems [34, 35].

6.3.2 Device Model Refinement of the Engineering System Net

In addition to the continuity laws imposed by the engineering system net defined in the previous section, a set of device models must be added to describe the behavior of each system capability (or degree of freedom). The nature of the device model depends on 1.) the type of engineering system, 2.) the nature of each capability, and 3.) the resolution (or degree of decomposition) by which the capability has been defined. In time-driven systems with engineering physics and “elemental" capabilities, these device models are constitutive laws (e.g. Ohm’s resistor law, the capacitor law, and the inductor law) and compatibility laws (e.g. Kirchoff’s Voltage law for electrical circuits) [27, 320]. In such cases, the structural degrees of freedom (i.e. system capabilities) are equivalent to the degrees of freedom (i.e. generalized coordinates) in engineering physics [2, 6, 14, 20]. In other cases (e.g. power systems engineering), many elemental capabilities are combined into a single capability with a complex device model expressed as a set of simultaneous differential algebraic equations [27, 321].

Given the tremendous diversity of engineering system device models, for the purposes of the hetero-functional network minimum cost flow optimization, this work restricts itself

to device models that create a fixed ratio between input and output operands (L) for each of the system process (P). These ratios are most easily implemented in a positive and negative device model matrix.

Definition 6.16 – Positive Device Model Matrix : A matrix $D_R^+ \in \mathbb{R}^{+\sigma(L)\times\sigma(P)}$ whose element $D_R^+(i,w)$ describes the relative quantity of operand l_i ejected by process p_w. ■

Definition 6.17 – Negative Device Model Matrix: A matrix $D_R^- \in \mathbb{R}^{+\sigma(L)\times\sigma(P)}$ whose element $D_R^-(i,w)$ describes the relative quantity of operand l_i consumed by process p_w. ■

The primary advantage of using device models of this form is that they can be readily folded into the positive and negative hetero-functional incidence tensors respectively.

$$\widehat{\mathcal{M}_\rho^+} = \left(\mathbb{1}^{\sigma(B_S)} \circ \left(\mathbb{1}^{\sigma(R)T} \otimes D_R^+\right)\mathbb{P}_S^T\right)^T \odot \widetilde{\mathcal{M}_\rho^+} \tag{6.25}$$

$$\widehat{\mathcal{M}_\rho^-} = \left(\mathbb{1}^{\sigma(B_S)} \circ \left(\mathbb{1}^{\sigma(R)T} \otimes D_R^+\right)\mathbb{P}_S^T\right)^T \odot \widetilde{\mathcal{M}_\rho^-} \tag{6.26}$$

where $\circ$ is the third-order outer product [322, 323], and $\widehat{\mathcal{M}_\rho^+}$ and $\widehat{\mathcal{M}_\rho^-}$ are the positive and negative third-order device model refined hetero-functional incidence tensors of size $\sigma(L) \times \sigma(B_S) \times \sigma(\mathcal{E}_S)$. These refined hetero-functional incidence tensors are then reincorporated directly into engineering system net (in Defn. 6.15).

6.3.3 Operand Behavior with the Service Model

The second element in the hetero-functional network dynamics is the system operand behavior through Service Nets (Defn. 6.9) and their dynamics (Defn. 6.10). These definitions are adopted directly into the hetero-functional network dynamics without change.

6.3.4 Synchronization Matrix

In hetero-functional graph theory, the engineering system net and the service nets are coupled through the service feasibility matrices (Defn. 6.11). The coupling of their dynamics is achieved through the synchronization of the engineering system net and service net firing

vectors. The state of the engineering system net is distinct from the state of the service net, but the transitions of both nets are coupled in time. The negative firing vectors indicate the start of transitions, they are synchronized by the negative service feasibility matrix $\widetilde{\Lambda}_i^-$. The positive firing vectors indicate the end of transitions, they are synchronized by the positive service feasibility matrix $\widetilde{\Lambda}_i^+$.

The service synchronization must, however, also reflect the device models as implemented in the engineering system net. Consequently, the service feasibility matrices are first converted to the Synchronization Matrices:

$$\widehat{\Lambda}_i^+ = \widetilde{\Lambda}_i^+ \odot \left(\left[e_i^{\sigma(L)T} \mathbb{P}_S (\mathbb{1}^{\sigma(R)T} \otimes D_R^+) \right] \otimes \mathbb{1}^{\sigma(\mathcal{E}_{l_i})} \right) \tag{6.27}$$

$$\widehat{\Lambda}_i^- = \widetilde{\Lambda}_i^- \odot \left(\left[e_i^{\sigma(L)T} \mathbb{P}_S (\mathbb{1}^{\sigma(R)T} \otimes D_R^-) \right] \otimes \mathbb{1}^{\sigma(\mathcal{E}_{l_i})} \right) \tag{6.28}$$

$$\forall i \in \{1, \ldots, \sigma(L)\}$$

Then, the positive and negative firing vectors of the engineering system net and service nets are synchronized through the service synchronization equations:

$$U_{l_i}^+[k] = \widehat{\Lambda}_i^+ U_C^+[k] \quad \forall i \in \{1, \ldots, \sigma(L)\},\ k \in \{1, \ldots, K\} \tag{6.29}$$

$$U_{l_i}^-[k] = \widehat{\Lambda}_i^- U_C^-[k] \quad \forall i \in \{1, \ldots, \sigma(L)\},\ k \in \{1, \ldots, K\} \tag{6.30}$$

Note that the duration of transitions in the service net is a result of the duration of transitions in the engineering system net.

6.4 Hetero-functional Network Minimum Cost Flow

This section develops the hetero-functional network minimum cost flow optimization program so as to optimize the dynamic system model developed in the previous section (Sec. 6.3). The first four constraints incorporate the engineering system net (Sec. 6.4.1) and

service net dynamics (Sec. 6.4.2), their synchronization (Sec. 6.4.3), and their transition duration (Sec. 6.4.4). The section then defines the boundary constraints (Sec. 6.4.5), the initial and final conditions (Sec. 6.4.6), the capacity constraints (Sec. 6.4.7), and the objective function (Sec. 6.4.8). Finally, Sec. 6.4.9 provides the compiled optimization program.

6.4.1 Engineering System Net

The engineering system net was defined as an ac-CPN in Sec. 6.3.1. The state of the ac-CPN is defined as a multiset, which cannot be optimized with a conventional quadratic program over reals. It is therefore necessary to convert the ac-CPN to a regular Petri net using Algorithm 6.1. As a result of the conversion, the engineering system dynamics are now described by a net with the following properties:

- S is the set of places with length: $\sigma(L)\sigma(B_S)$,
- $\mathcal{E}$ is the set of transitions with length: $\sigma(\mathcal{E}_S)$,
- $\mathbf{M}$ is the set of arcs, with the associated incidence matrices: $M = M^+ - M^-$,
- W is the set of weights on the arcs, as captured in the incidence matrices,
- Q is the marking vector for both the set of places and the set of transitions.

The state transition equations of the engineering system net are:

$$Q[k+1] = \Phi_T(Q[k], U^-[k], U^+[k]) \quad \forall k \in \{1, \ldots, K\} \tag{6.31}$$

where $Q = [Q_B; Q_\mathcal{E}]$ and

$$Q_B[k+1] = Q_B[k] + M^+U^+[k] - M^-U^-[k] \tag{6.32}$$

$$Q_\mathcal{E}[k+1] = Q_\mathcal{E}[k] - U^+[k] + U^-[k] \tag{6.33}$$

where $U^+ = U^+_{\mathcal{C}}$, $U^- = U^-_{\mathcal{C}}$, Q_B has size $\sigma(L)\sigma(B_S) \times 1$, and $Q_{\mathcal{E}}$ has size $\sigma(\mathcal{E}_S) \times 1$. These state transition functions are incorporated directly into the quadratic program in Sec. 6.4.9.

6.4.2 Service Net

The service net was defined as a Petri net in Sec. 6.2.4. Recall that its dynamics are described by the transition function in Eq. 6.11. The optimization program constraints require the concatenation of the state space equations over all the operands in the system: $\Phi_L(Q_L[k], U^-_L[k], U^+_L[k])$, where $Q_L = [Q_{SL}; Q_{\mathcal{E}L}]$:

$$Q_{SL}[k+1] = Q_{SL}[k] + M^+_L U^+_L[k] - M^-_L U^-_L[k] \quad \forall k \in \{1, \dots, K\} \tag{6.34}$$

$$Q_{\mathcal{E}L}[k+1] = Q_{\mathcal{E}L}[k] - U^+_L[k] + U^-_L[k] \quad \forall k \in \{1, \dots, K\} \tag{6.35}$$

where: Q_{SL} has length $\sigma(Q_{SL}) = \sum_{l_i \in L} \sigma(S_{l_i})$ and is the vertical concatenation of the service net place markings for all operands in L:

$$Q_{SL} = \begin{bmatrix} Q_{Sl_1} \\ \vdots \\ Q_{Sl_{\sigma(L)}} \end{bmatrix} \tag{6.36}$$

$Q_{\mathcal{E}L}$ has length $\sigma(Q_{\mathcal{E}L}) = \sum_{l_i \in L} \sigma(\mathcal{E}_{l_i})$ and is the vertical concatenation of the service net transition markings for all operands in L:

$$Q_{\mathcal{E}L} = \begin{bmatrix} Q_{\mathcal{E}l_1} \\ \vdots \\ Q_{\mathcal{E}l_{\sigma(L)}} \end{bmatrix} \tag{6.37}$$

U_L^+ and U_L^+ are the vertical concatenations of the service net positive and negative firing vectors for all operands in L:

$$U_L^+ = \begin{bmatrix} U_{l_1}^+ \\ \vdots \\ U_{l_{\sigma(L)}}^+ \end{bmatrix}, \qquad U_L^- = \begin{bmatrix} U_{l_1}^- \\ \vdots \\ U_{l_{\sigma(L)}}^- \end{bmatrix} \tag{6.38}$$

where U_L^+ and U_L^- have size $\sigma(Q_{\mathcal{E}L}) \times 1$. Finally M_L^+ and M_L^- are the block-diagonal positive and negative system service net incidence matrices:

$$M_L^+ = \begin{bmatrix} M_{l_1}^+ & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & M_{l_{\sigma(L)}}^+ \end{bmatrix}, \quad M_L^- = \begin{bmatrix} M_{l_1}^- & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & M_{l_{\sigma(L)}}^- \end{bmatrix} \tag{6.39}$$

where M_L^+ and M_L^+ have size $\sigma(Q_{SL}) \times \sigma(Q_{\mathcal{E}L})$.

6.4.3 Synchronization Constraint

The synchronization of the engineering system net and the service nets was defined in Sec. 6.3.4. The conversion from the engineering system firing vector $U_{\mathcal{C}}$ to the Petri net firing vector U requires the conversion of Eqs. 6.29 and 6.30 to:

$$U_L^+[k] = \widehat{\Lambda}^+ U^+[k] \qquad \forall k \in \{1, \dots, K\} \tag{6.40}$$

$$U_L^-[k] = \widehat{\Lambda}^- U^-[k] \qquad \forall k \in \{1, \dots, K\} \tag{6.41}$$

where:

$$\widehat{\Lambda}^+ = \begin{bmatrix} \widehat{\Lambda}_1^+ \\ \vdots \\ \widehat{\Lambda}_{\sigma(L)}^+ \end{bmatrix}, \quad \widehat{\Lambda}^- = \begin{bmatrix} \widehat{\Lambda}_1^- \\ \vdots \\ \widehat{\Lambda}_{\sigma(L)}^- \end{bmatrix} \tag{6.42}$$

6.4.4 Duration Constraints

The duration constraints are adopted from Eq. 6.22. As the Engineering System Net firing vector is converted to a Petri net firing vector, the equation is defined as:

$$U_{\psi}^{+}[k + k_{d\psi}] = U_{\psi}^{-}[k] \qquad \forall k \in \{1, \ldots, K\} \tag{6.43}$$

where $U_{\psi}^{-}[k]$ indicates the ψ^{th} element of the $U^{-}[k]$ vector and where $k_{d\psi}$ is the duration of engineering system net transition ψ.

6.4.5 Boundary Constraints

The boundary constraints are the fifth element in the program. They define the interaction between the dynamic system and the context. These constraints are specifically used when modeling an *open* system. The boundary constraints consist of two types: 1) demand constraints that control output transitions and 2) supply constraints that control input transitions. The demand constraints are imposed on $U^{-}[k]$:

$$D_{Bn} U^{-}[k] = C_{Bn}[k] \quad \forall k \in \{1, \ldots, K\} \tag{6.44}$$

where D_{Bn} is a transition selector matrix of size: $\sigma(\mathcal{E}_{\text{Out}}) \times \sigma(\mathcal{E}_S)$, with one filled element per row in the column of the selected transition, where $\sigma(\mathcal{E}_{\text{Out}})$ is the *number of output transitions*. Vector $C_{Bn}[k]$ contains the demand data for each time step k.

The supply constraints are imposed on $U^{+}[k]$:

$$D_{Bp} U^{+}[k] = C_{Bp}[k] \quad \forall k \in \{1, \ldots, K\} \tag{6.45}$$

where D_{Bp} is a transition selector matrix of size: $\sigma(\mathcal{E}_{\text{In}}) \times \sigma(\mathcal{E}_S)$, with one filled element per row in the column of the selected transition, where $\sigma(\mathcal{E}_{\text{In}})$ is the *number of input transitions*. Vector $C_{Bp}[k]$ contains the supply data for each time step k. The boundary constraints are

combined in a single equation:

$$\begin{bmatrix} D_{Bp} & \mathbf{0} \\ \mathbf{0} & D_{Bn} \end{bmatrix} \begin{bmatrix} U^{+} \\ U^{-} \end{bmatrix} [k] = \begin{bmatrix} C_{Bp} \\ C_{Bn} \end{bmatrix} [k] \qquad \forall k \in \{1, \ldots, K\} \tag{6.46}$$

6.4.6 Initial and Final Conditions

The initial conditions constrain the system at the initial time step: $k = 1$. This allows the program to be used with a pre-populated system (also called a “hot-start"). The initial conditions of the *input* transitions should be left undetermined when modeling an *open* system – the optimization program will determine the quantities of the operands that need to enter the system in order to satisfy the demand. The initial condition constraints are:

$$\left[Q_B; Q_{\mathcal{E}}; Q_{SL} \right] [k = 1] = \left[C_{B1}; C_{\mathcal{E}1}; C_{SL1} \right] \tag{6.47}$$

where *";"* is the MATLAB operator to define a vertically concatenated matrix.

The final conditions constrain the system at the final time step: $k = K + 1$. The final conditions of the *output* transitions should be left open when modeling an *open* system. The state of those transitions in the last time step contains the cumulative outputs of that specific transition. Finally, in order to ensure that all tokens are accounted for, the negative firing vectors of the engineering system net and the system service net are set to zero.

$$\left[Q_B; Q_{\mathcal{E}}; Q_{SL}; U^{-}; U_L^{-} \right] [k = K + 1] = \left[C_{BK}; C_{\mathcal{E}K}; C_{SLK}; \mathbf{0}; \mathbf{0} \right] \tag{6.48}$$

6.4.7 Capacity Constraints

The capacity constraints impose limits on the engineering system net. The capacity constraints limit the amount of each operand that can be fired at any point in time:

$$U^{-}[k] \leq C_U \qquad \forall k \in \{1, \ldots, K\} \tag{6.49}$$

This equation is modified to account for system input transitions: transitions that input operands to the system without a predetermined value. These transitions are constrained specifically on the positive firing vectors.

$$\begin{bmatrix} D_{Cp} & \mathbf{0} \\ \mathbf{0} & I^{\sigma(\mathcal{E}_S)} \end{bmatrix} \begin{bmatrix} U^{+} \\ U^{-} \end{bmatrix} [k] \leq C_U \qquad \forall k \in \{1, \ldots, K+1\} \tag{6.50}$$

where D_{Cp} selects the system input transitions without a predetermined value.

6.4.8 Objective Function

Finally, the objective function motivates the objective of the optimization program. It contains the cost or benefit of the execution of the decision variables. For the hetero-functional network minimum cost flow program, the cost is related to the execution of engineering system net transitions. However, when desired, cost can be imposed on other elements of the set of decision variables. The set of decision variables (as defined piece-wise in the previous sections) is defined as:

$$x[k] = \left[Q_B; Q_{\mathcal{E}}; Q_{SL}; Q_{\mathcal{E}L}; U^{+}; U^{-}; U_L^{+}; U_L^{-} \right] [k] \qquad \forall k \in \{1, \ldots, K+1\} \tag{6.51}$$

where the size of the set of decision variables is:

$$\sigma(x) = (K+1)\left(\sigma(B_S) + 3\sigma(\mathcal{E}_S) + \sigma(Q_{SL}) + 3\sigma(Q_{\mathcal{E}L})\right) \tag{6.52}$$

The cost function is imposed on the decision variables as either a linear or a quadratic function. This work introduces a quadratic objective function. The resulting objective function has the following form:

$$\text{minimize } Z = x^T F_{QP} x + f_{QP}^T x \tag{6.53}$$

where $F_{QP} \geq 0$ is the quadratic cost coefficient (a matrix of size $\sigma(x) \times \sigma(x)$), and where $f_{QP} \geq 0$ is the linear cost coefficient (a vector of size $\sigma(x) \times 1$). Note that the quadratic cost matrix F_{QP} is assumed to be diagonal. Furthermore, for all zero-valued elements on the diagonal, an infinitesimally small value may be added to ensure that the quadratic cost matrix is positive definite ($F_{QP} \succ 0$). This guarantees convexity of the quadratic program.

6.4.9 Optimization Program Compilation

Finally, this section compiles the elements of the optimization program to define the hetero-functional network minimum cost flow program. The canonical form of a linearly constrained quadratic program is presented below:

$$\text{minimize } Z = x^T F_{QP} x + f_{QP}^T x \tag{6.54}$$

$$\text{s.t. } A_{QP} x = B_{QP} \tag{6.55}$$

$$D_{QP} x \leq E_{QP} \tag{6.56}$$

$$x \geq 0, \quad x \in \mathbb{R} \tag{6.57}$$

where:

- x has size $\sigma(x) \times 1$, as defined in Eq. 6.52,
- F_{QP} has size: $\sigma(x) \times \sigma(x)$,
- f_{QP} has size: $\sigma(x) \times 1$,
- A_{QP} has size: $\sigma(A_{QP}) \times \sigma(x)$
- B_{QP} has size: $\sigma(A_{QP}) \times 1$,
- D_{QP} has size: $\sigma(D_{QP}) \times \sigma(x)$
- E_{QP} has size: $\sigma(D_{QP}) \times 1$.

Matrix A_{QP} and vector B_{QP} are constructed by concatenating eight constraints (Eqs. 6.58 through 6.65) over all time steps K with the initial and final condition constraints (Eqs. 6.66 and 6.67):

$$-Q_B[k+1] + Q_B[k] + M^+ U^+[k] - M^- U^-[k] = 0 \tag{6.58}$$

$$-Q_{\mathcal{E}}[k+1] + Q_{\mathcal{E}}[k] - U^+[k] + U^-[k] = 0 \tag{6.59}$$

$$-U^+[k+k_{d\psi}] + U^-[k] = 0 \tag{6.60}$$

$$-Q_{SL}[k+1] + Q_{SL}[k] + M_L^+ U_L^+[k] - M_L^- U_L^-[k] = 0 \tag{6.61}$$

$$-Q_{\mathcal{E}L}[k+1] + Q_{\mathcal{E}L}[k] - U_L^+[k] + U_L^-[k] = 0 \tag{6.62}$$

$$U_L^+[k] - \widehat{\Lambda}^+ U^+[k] = 0 \tag{6.63}$$

$$U_L^-[k] - \widehat{\Lambda}^- U^-[k] = 0 \tag{6.64}$$

$$\begin{bmatrix} D_{Bp} & \mathbf{0} \\ \mathbf{0} & D_{Bn} \end{bmatrix} \begin{bmatrix} U^+ \\ U^- \end{bmatrix}[k] = \begin{bmatrix} C_{Bp} \\ C_{Bn} \end{bmatrix}[k] \tag{6.65}$$

where Eqs. 6.58 through 6.65 defined for all $k \in \{1, \ldots, K\}$. The initial and final condition

constraints are:

$$\Big[Q_B; Q_{\mathcal{E}}; Q_{SL}\Big][k=1] = \Big[C_{B1}; C_{\mathcal{E}1}; C_{SL1}\Big] \tag{6.66}$$

$$\Big[Q_B; Q_{\mathcal{E}}; Q_{SL}; U^-; U_L^-\Big][k=K+1] = \Big[C_{BK}; C_{\mathcal{E}K}; C_{SLK}; \mathbf{0}; \mathbf{0}\Big] \tag{6.67}$$

Consequently, the number of rows in the A_{QP} matrix is defined as:

$$\begin{aligned}\sigma(A_{QP}) = K\Big[&\sigma(Q_B) + 2\sigma(Q_{\mathcal{E}}) + \sigma(Q_{SL}) + 3\sigma(Q_{\mathcal{E}L}) \; + \\ &\sigma(\mathcal{E}_{\text{Out}}) + \sigma(\mathcal{E}_{\text{In}})\Big] \; + \sigma(Q_B) + \sigma(Q_{\mathcal{E}}) + \sigma(Q_{SL}) \; + \\ &\sigma(Q_B) + 2\sigma(Q_{\mathcal{E}}) + \sigma(Q_{SL}) + \sigma(Q_{\mathcal{E}L})\end{aligned} \tag{6.68}$$

Note that the number of decision variables is defined over $K+1$ time steps to accommodate the mathematical structure of the state transition equations.

The inequality constraints, $Dx \leq E$, contain the capacity constraints:

$$\begin{bmatrix} D_{Cp} & \mathbf{0} \\ \mathbf{0} & I^{\sigma(\mathcal{E}_S)} \end{bmatrix} \begin{bmatrix} U^+ \\ U^- \end{bmatrix} [k] \leq C_U \qquad \forall k \in \{1, \ldots, K+1\} \tag{6.69}$$

which is defined over the time steps $K+1$ to maintain consistency with the number of decision variables. The number of rows of the inequality matrix D_{QP} is defined as:

$$\sigma(D_{QP}) = (K+1)\Big[\sigma(\mathcal{E}_{\text{In}}) + \sigma(Q_{\mathcal{E}})\Big] \tag{6.70}$$

6.5 Illustrative Example: Hydrogen-Natural Gas System

This section introduces a test case to demonstrate the application of the hetero-functional network minimum cost-flow program. The section first introduces the context of the test case, then it provides the test case data and finally, the it introduces four optimization

scenarios.

6.5.1 Introduction

Test cases enable the study of modeling, simulation, and optimization methods of complex critical (infrastructure) systems [50,251,254]. The test case in this work is the first hydrogen-natural gas infrastructure test case to the knowledge of the authors. The test case is inspired by the Dutch natural gas system and the plans for a European hydrogen pipeline network [324] and it does not aim to represent the current or future system.

The plans to develop hydrogen infrastructure are driven by the need for the reduction of carbon emissions. Electrolysis enables carbon-free generation of hydrogen from electric power and water. Consequently, hydrogen may serve as an intermediate mode of energy storage. A secondary benefit is that some industrial processes require a high-heat energy source. This is challenging to achieve through electric power, but hydrogen provides a (still expensive) alternative to natural gas and coal. Finally, natural gas is currently used as the energy source for the production of hydrogen. As a consequence, the hydrogen and natural gas system have interdependencies and overlap of their services. This interdependent system is especially challenging to operate and optimize.

6.5.2 Test Case Data

This subsection first introduces the physical lay-out of the test case. Then, it discusses the device models for the processes.

6.5.2.1 Test Case Physical Lay-out

The lay-out of the test case is derived from the topology of the Dutch industrial and infrastructure clusters (see Fig. 6.4 on Page 278):

The *south-west area* of The Netherlands accommodates critical energy infrastructure: a hydrogen electrolysis facility (Node 1), a steam-methane reformation facility (Node 2), a

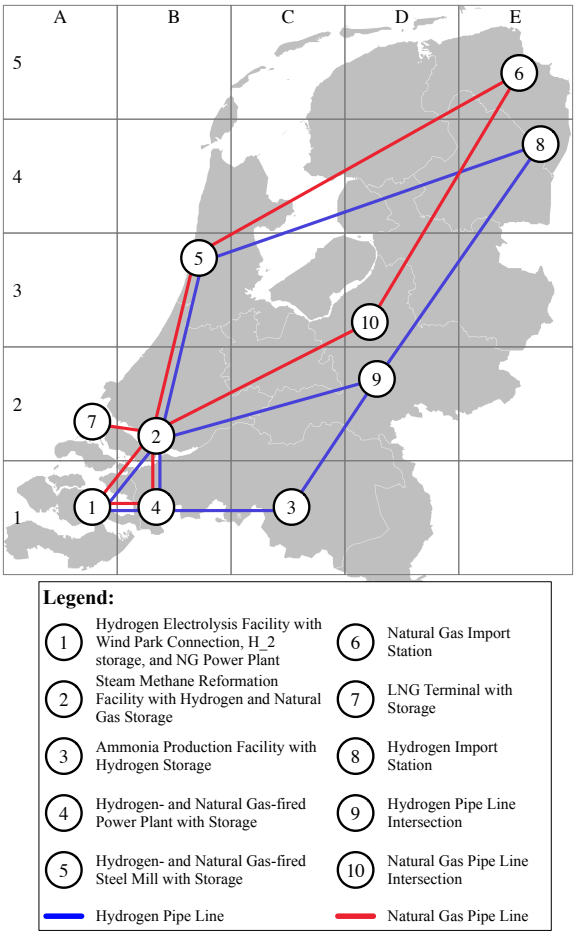


Fig. 6.4: Hydrogen Natural Gas Test Case.

power generation cluster (Node 4), and an LNG terminal (Node 7). The *north-west region* of the test case contains heavy industry: a steel mill that uses a combination of natural gas and hydrogen as its fuel (Node 5). The *north-east* contains infrastructure that imports natural gas (Node 6) and hydrogen (Node 8) to the system. Finally, the *mid- and south-east* region contains two pipeline junctions (Nodes 9 and 10) and an ammonia factory (Node 3).

The system consists of industrial clusters, connected through dedicated pipelines for hydrogen and natural gas. Table 6.1 provides an overview of the clusters with the associated processes, the cost, the capacities of the processes, and the processing time. Note that most process capacities are not intended to be a binding constraint, however, the capacities of hydrogen pipe lines 4 and 6 are likely to be binding in some scenarios.

Table. 6.1: Overview of the test case resources, processes, cost, capacity, and processing time.

Node #	Node Name	Processes	Quadratic Cost	Linear Cost	Capacity	Processing Time
1	Hydrogen Electrolysis Facility	Electrolyze Water to Hydrogen and Oxygen	-	\$1000 / ton H_2	3,000 ton H_2 / day	2 days
		Burn Natural Gas to Generate Electric Power	0.01 $\$^2$/ton CH_4	\$145 / ton CH_4	3,000 ton CH_4 / day	1 day
		Import Electric Power	-	\$10 MWh	100,000 MWh / day	0 days
		Import Water	-	-	30,000 ton H_2O / day	0 days
		Export Water	-	-	30,000 ton H_2O / day	0 days
		Import Oxygen	-	-	30,000 ton O_2 / day	0 days
		Export Oxygen	-	-	30,000 ton O_2 / day	0 days
		Export CO2	-	See Scenarios	30,000 ton CO_2 / day	0 days
		Export Heat Loss	-	-	30,000 MMBTU / day	0 days
		Store Hydrogen	-	\$ 0.1 / ton H_2	21,000 ton H_2 / day	1 day
		Store Natural Gas	-	\$ 0.1 / ton CH_4	100,000 ton CH_4 / day	1 day
2	Steam Methane Reformation Facility	Reform Steam and Methane to Hydrogen and CO2	-	\$ 1000 / ton H_2	3,000 ton H_2 / day	2 days
		Burn Natural Gas to Generate Industrial Heat	-	\$ 100 / ton CH_4	1,000 ton CH_4 / day	1 day
		Import Water	-	-	30,000 ton H_2O / day	0 days
		Export Water	-	-	30,000 ton H_2O / day	0 days
		Import Oxygen	-	-	30,000 ton O_2 / day	0 days
		Export CO2	-	See Scenarios	30,000 ton CO_2 / day	0 days
		Store Hydrogen	-	\$ 0.1 / ton H_2	21,000 ton H_2 / day	1 day
		Store Natural Gas	-	\$ 0.1 / ton CH_4	100,000 ton CH_4 / day	1 day
3	Ammonia Production Facility	Manufacture Ammonia	-	\$ 100 / ton H_2	2,000 ton H_2 / day	0 days
		Store Hydrogen	-	\$ 0.1 / ton H_2	21,000 ton H_2 / day	1 day
4	Hydrogen- and Natural Gas-fired Power Plant	Burn Hydrogen to Generate Electric Power	0.01 $\$^2$/ton H_2	\$ 1000 / ton H_2	1,000 ton H_2 / day	1 day
		Burn Natural Gas to Generate Electric Power	0.01 $\$^2$/ton CH_4	\$ 145 / ton CH_4	3,000 ton CH_4 / day	1 day
		Consume Electric Power	-	-	10,000 MWh / day	0 days
		Import Oxygen	-	-	30,000 ton O_2 / day	0 days
		Export Water	-	-	30,000 ton H_2O / day	0 days
		Export Heat Loss	-	-	30,000 MMBTU / day	0 days
		Export CO2	-	See Scenarios	30,000 ton CO_2 / day	0 days
		Store Hydrogen	-	\$ 0.1 / ton H_2	21,000 ton H_2 / day	1 day
		Store Natural Gas	-	\$ 0.1 / ton CH_4	100,000 ton CH_4 / day	1 day
5	Hydrogen- and Natural Gas-fired Steel Mill	Burn Hydrogen to Generate Industrial Heat	-	\$ 300 / ton H_2	1,000 ton H_2 / day	1 day
		Burn Natural Gas to Generate Industrial Heat	-	\$ 100 / ton CH_4	1,000 ton CH_4 / day	1 day
		Consume Industrial Heat	-	-	5,000 MMBTU / day	0 days
		Export Water	-	-	30,000 ton H_2O / day	0 days
		Export CO2	-	See Scenarios	30,000 ton CO_2 / day	0 days
		Import Oxygen	-	-	30,000 ton O_2 / day	0 days
		Store Hydrogen	-	\$ 0.1 / ton H_2	21,000 ton H_2 / day	1 day
		Store Natural Gas	-	\$ 0.1 / ton CH_4	100,000 ton CH_4 / day	1 day
6	Natural Gas Import Station	Import Natural Gas	-	\$ 130 / ton CH_4	100,000 ton CH_4 / day	0 days
7	LNG Terminal	Regasify Natural Gas	-	\$ 210 / ton CH_4	100,000 ton CH_4 / day	0 days
		Store Natural Gas	-	\$ 0.1 / ton CH_4	100,000 ton CH_4 / day	1 day
8	Hydrogen Import Station	Import Hydrogen	-	\$ 3000 / ton H_2	100,000 ton H_2 / day	0 days
9	Hydrogen Pipe Line Intersection	Store Hydrogen	-	\$ 0.1 / ton H_2	21,000 ton H_2 / day	1 day
10	Natural Gas Pipe Line Intersection	Store Natural Gas	-	\$ 0.1 / ton CH_4	100,000 ton CH_4 / day	1 day
	Hydrogen Pipe Line	Transport Hydrogen	-	\$ 0.01 / ton H_2	10,000 ton H_2 / day	1 day
	Hydrogen Pipe Line 4	Transport Hydrogen	-	\$ 0.01 / ton H_2	260 ton H_2 / day	1 day
	Hydrogen Pipe Line 6	Transport Hydrogen	-	\$ 0.01 / ton H_2	260 ton H_2 / day	1 day
	Natural Gas Pipe Line	Transport Natural Gas	-	\$ 0.01 / ton CH_4	10,000 ton CH_4 / day	1 day

Table. 6.2: Overview of the test case supply and demand data.

Day	Electric Power Supply at Node 1 [MWh / day]		Hydrogen Consumption for Ammonia Production at Node 3 [ton / day]	Electric Power Consumption at Node 4 [MWh / day]	Industrial Heat Consumption at Node 5 [MMBTU / day]
	Scenarios 1 & 3	Scenarios 2 & 4			
1	0	6000	0	0	0
2	0	6000	0	0	0
3	0	6000	0	0	0
4	0	6000	0	0	0
5	0	6000	126	1435	35000
6	0	6000	126	1459	35000
7	0	6000	126	1312	35000
8	0	6000	126	1189	35000
9	0	6000	126	1402	35000
10	0	6000	126	1404	35000
11	0	6000	126	1363	35000
12	0	6000	126	1416	35000
13	0	6000	126	1479	35000
14	0	6000	126	1288	35000
15	0	6000	126	1281	35000
16	0	0	126	1455	35000
17	0	0	126	1480	35000
18	0	0	126	1476	35000
19	0	0	126	1275	35000
20	0	0	0	0	0

6.5.2.2 Device Models

The dynamics of the test case processes are described through their device models. These device models are (mass-based) ratios between input and output operands derived from their stoichiometry. All weights are in metric ton (1000 kg). The device models of the transformative processes are derived from the relevant literature:

1. Electrolyze Water to Hydrogen and Oxygen [325]:

$$2H_2O + \text{ Electric Power } \rightarrow 2H_2 + O_2 \tag{6.71}$$

   This process consumes 40 - 50 MWh / ton H_2 [326]. The associated mass-based ratio is:

$$8.936\ ton\ H_2O\ +\ 40\ MWh\ \rightarrow 1\ ton\ H_2\ +\ 7.936\ ton\ O_2 \tag{6.72}$$

2. Reform Steam and Methane to Hydrogen and CO_2: The stoichiometric equation

combines the steam reformation process and the shift reaction [327, 328]:

$$CH_4 + 2H_2O + \text{Industrial Heat} \rightarrow 4H_2 + CO_2 \tag{6.73}$$

This process consumes around 19.4 MMBTU per ton H_2 produced. The associated mass-based stochiometric equation for steam methane reformation is:

$$1.989\ ton\ CH_4\ +\ 4.468\ ton\ H_2O\ +\ 19.4\ \text{MMBTU} \rightarrow$$
$$1\ ton\ H_2\ +\ 5.457\ ton\ CO_2 \tag{6.74}$$

3. Burn Natural Gas to Generate Industrial Heat:

$$CH_4 + 2O_2 \rightarrow 2H_2O + CO_2 + \text{ Industrial Heat} \tag{6.75}$$

For this ratio, it is assumed that all generated industrial heat is used productively (with a HHV of CH_4 of 891 kJ / mol) [329]. The associated mass-based ratio is:

$$1\ ton\ CH_4\ +\ 3.989\ ton\ O_2\ \rightarrow$$
$$2.246\ ton\ H_2O\ +\ 2.743\ ton\ CO_2\ +\ 52.6\ \text{MMBTU} \tag{6.76}$$

4. Burn Natural Gas to Generate Electric Power:

$$CH_4 + 2O_2 \rightarrow 2H_2O + CO_2 + \text{ Electric Power} + \text{ Heat Loss} \tag{6.77}$$

The heat rate is assumed at 7633 BTU / kWh [329, 330]. The associated mass-based

ratio is:

$$1\ ton\ CH_4\ +\ 3.989\ ton\ O_2\ \rightarrow$$
$$2.246\ ton\ H_2O\ +\ 2.743\ ton\ CO_2\ +\ 6.897\ \text{MWh}\ +\ 29.1\ \text{MMBTU} \quad (6.78)$$

5. Burn Hydrogen to Generate Industrial Heat:

$$2H_2 + O_2 \rightarrow 2H_2O + \text{ Industrial Heat} \quad (6.79)$$

Where all generated heat is used productively [329]. The associated mass-based ratio is:

$$1\ ton\ H_2\ +\ 7.936\ ton\ O_2\ \rightarrow 8.936\ ton\ H_2O\ +\ 134.5\ \text{MMBTU} \quad (6.80)$$

6. Burn Hydrogen to Generate Electric Power:

$$2H_2 + O_2 \rightarrow 2H_2O + \text{ Electric Power} + \text{ Heat Loss} \quad (6.81)$$

For this ratio, the heat rate of the hydrogen-fired turbine is assumed to be 7633 BTU / kWh. As a result, the mass-based ratio is:

$$1\ ton\ H_2\ +\ 7.936\ ton\ O_2\ \rightarrow$$
$$8.936\ ton\ H_2O\ + 17.616\ \text{MWh}\ +\ 74.3\ \text{MMBTU} \quad (6.82)$$

The remaining transformation processes import or consume operands and are defined only what they bring into or take out of the system, as displayed in Table 6.1.

Finally, the test case assumes that all transportation processes are lossless:

- Transport Natural Gas, expressed in ton per day.

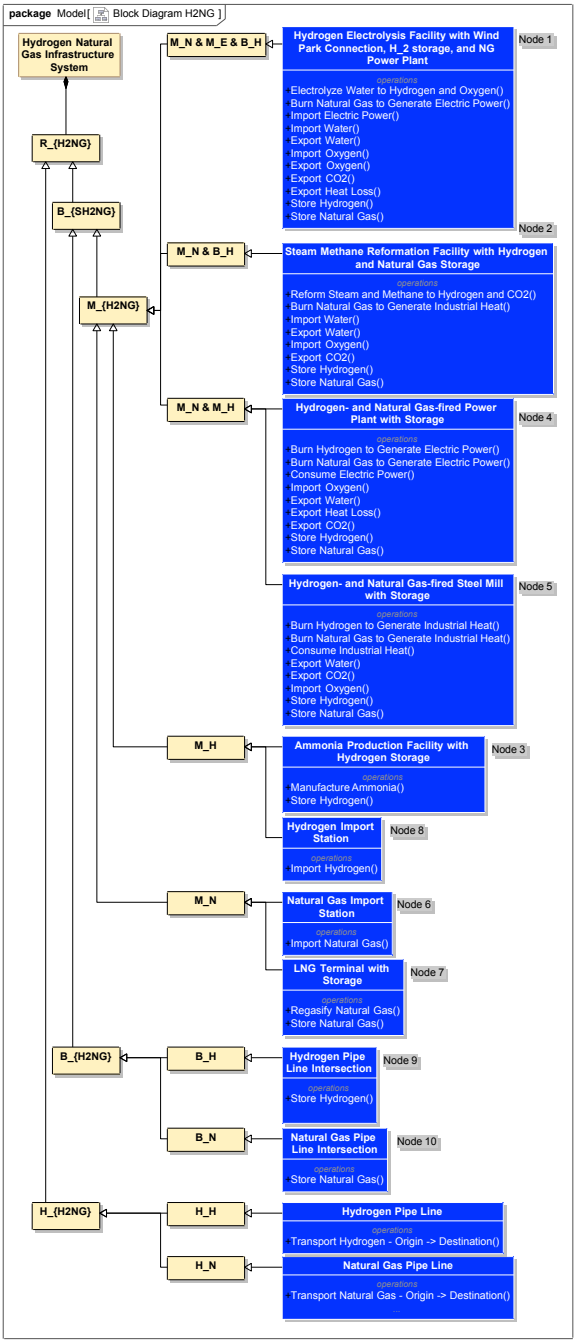


Fig. 6.5: SysML Block Diagram of the Hydrogen-Natural Gas System Resources.

- Transport Hydrogen, expressed in ton per day.

Table 6.2 presents the four *supply and demand* curves. Note that in this test case, electric power cannot be stored and needs to be used immediately.

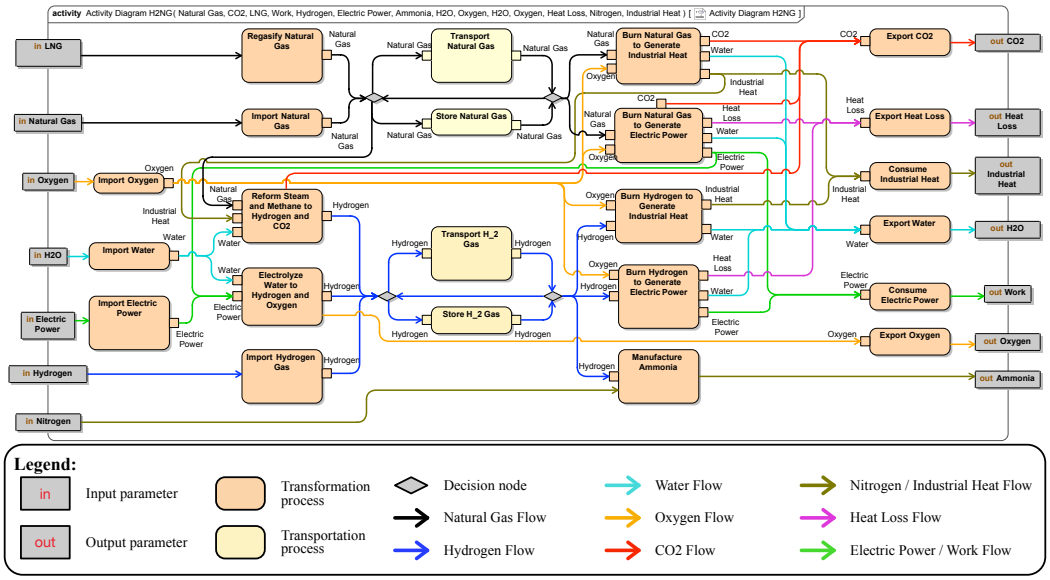


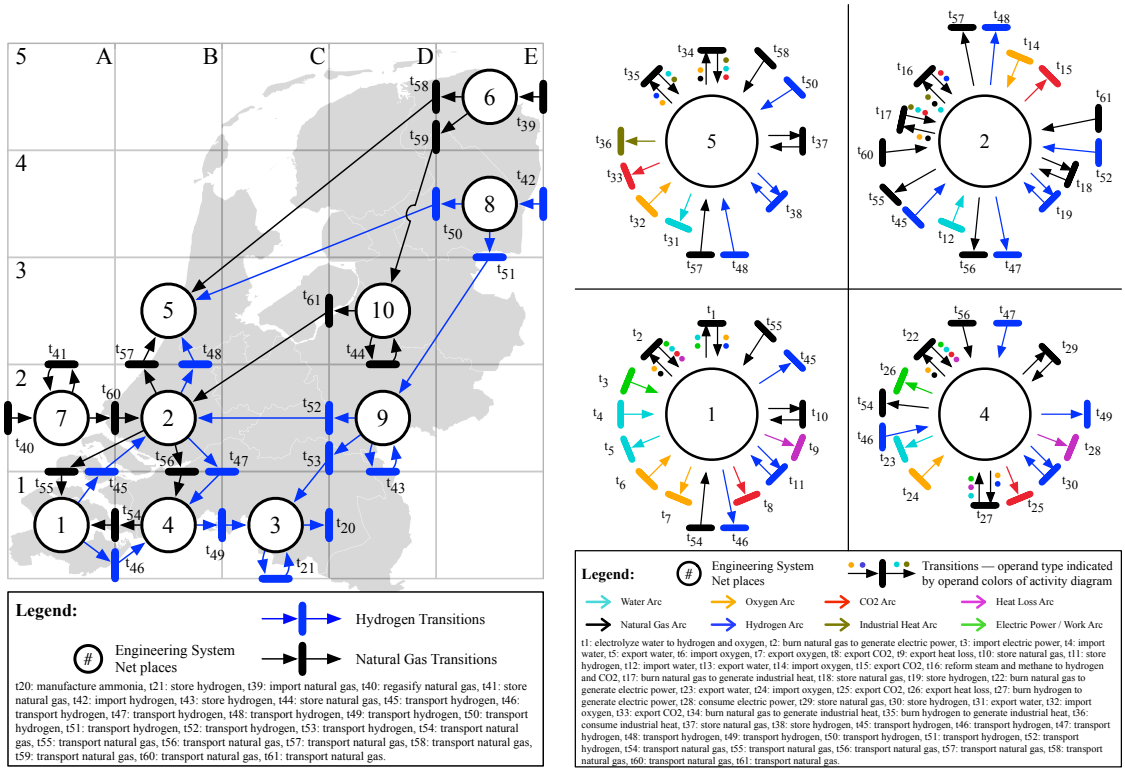
Fig. 6.6: SysML Activity Diagram of the Hydrogen-Natural Gas System Processes.

6.5.3 Scenario Data

The test case optimizes four scenarios:

- Scenario 1: the base case scenario without carbon pricing or a fixed renewable electricity supply.
- Scenario 2: incorporates carbon pricing of $250 per ton for carbon emissions at the steel mill. It does not include a fixed renewable electricity supply.
- Scenario 3: introduces the fixed renewable electricity supply. It does not include carbon pricing.
- Scenario 4: incorporates carbon pricing of $500 per ton for all resources and the fixed renewable electricity supply.

For each of these scenarios, the goal is to have the lowest fulfillment cost for the three demand operands over the 20 day time horizon ($K = 20$).



(a) Full system view. The transformation and storage transitions in Nodes 1, 2, 4, and 5 have been left out for clarity.

(b) Detailed view of Nodes 1, 2, 4, and 5.

Fig. 6.7: The engineering system net presented as an arc-constant Colored Petri net.

6.6 Results and Discussion

This section applies the hetero-functional network minimum cost flow program to the hydrogen-natural gas test case. Sec. 6.6.1 first covers the hetero-functional graph theory structural model. Sec. 6.6.2 then develops the dynamic model. Sec. 6.6.3 defines the optimization program. Finally, Sec. 6.6.4 discusses the results of the optimization program.

6.6.1 Hetero-functional Graph Theory Structural Model

The Hetero-functional Graph Theory structural model provides the foundation for the development of a dynamic model and an optimal control program. It contains the System Concept (Sec. 6.2.1), the Hetero-functional Incidence Tensor (Sec. 6.2.2), and the Service

Model (Sec. 6.2.4), which includes the Service Nets and the Service Feasibility Matrices.

Fig. 6.5 describes the system resources with a SysML block definition diagram. The test case contains 27 resources of which 8 are transformation resources, 2 are independent buffers, and 17 are transportation resources. The diagram also shows the processes allocated to each of the resources.

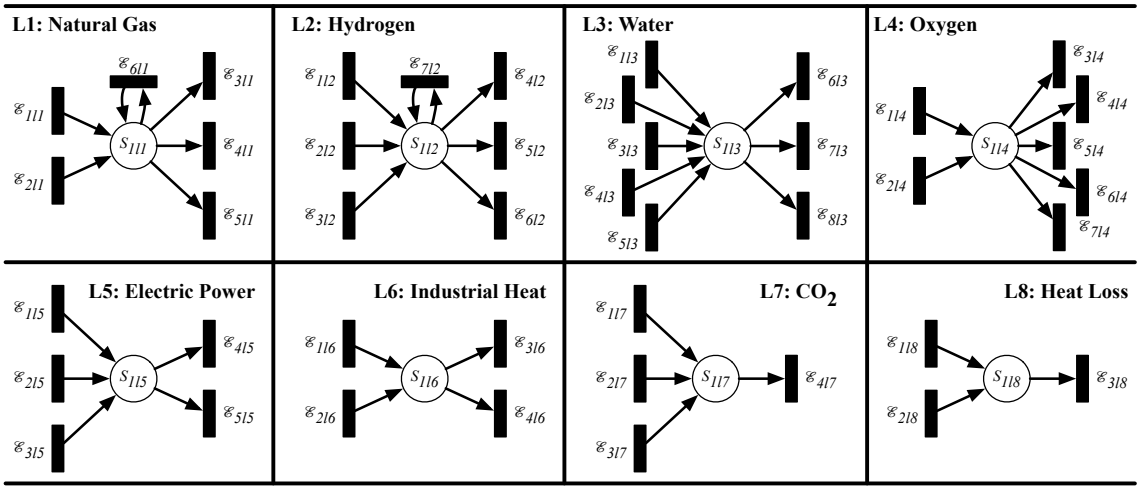
Fig. 6.6 describes the system processes with a SysML activity diagram. The activity diagram shows the functional reference architecture and the feasible system process sequences.

The system concept, or the allocated architecture, maps the system processes onto the system resources with the knowledge base. As expected, the knowledge base has size 219×27, with 61 filled elements.

The hetero-functional incidence tensor describes the association of the system buffers with the capabilities and the system operands. It is defined in Definitions 6.5 and 6.6. For this test case, the projected Hetero-functional Incidence Tensor has size: $8 \times 10 \times 61$ (operands by buffers by capabilities) and it has 98 filled elements. The associated Engineering System Net is presented in Figs. 6.7(a) and 6.7(b), where the latter provides a detailed look at Nodes 1, 2, 4, and 5.

Fig. 6.8 describes the service nets for all eight operands in the system. The service net incidence matrices are defined for each of the operands:

- Natural Gas, size: $\sigma(S_{l1}) \times \sigma(\mathcal{E}_{l1})$: 1×6.
- Hydrogen, size: $\sigma(S_{l2}) \times \sigma(\mathcal{E}_{l2})$: 1×7.
- Water, size: $\sigma(S_{l3}) \times \sigma(\mathcal{E}_{l3})$: 1×8.
- Oxygen, size: $\sigma(S_{l4}) \times \sigma(\mathcal{E}_{l4})$: 1×7.
- Electric Power, size: $\sigma(S_{l5}) \times \sigma(\mathcal{E}_{l5})$: 1×5.
- Industrial Heat, size: $\sigma(S_{l6}) \times \sigma(\mathcal{E}_{l6})$: 1×4.



Legend											
s_{111}	Natural Gas	e_{411}	Burn Natural Gas to Generate Electric Power	e_{213}	Burn Natural Gas to Generate Electric Power	e_{414}	Burn Natural Gas to Generate Electric Power	e_{216}	Burn Hydrogen to Generate Industrial Heat
s_{112}	Hydrogen	e_{511}	Reform Steam and Methane to Hydrogen and CO2	e_{313}	Import Water	e_{514}	Burn Hydrogen to Generate Industrial Heat	e_{316}	Consume Industrial Heat
s_{113}	Water	e_{611}	Transport Natural Gas	e_{413}	Burn Hydrogen to Generate Industrial Heat	e_{614}	Burn Hydrogen to Generate Electric Power	e_{416}	Reform Steam and Methane to Hydrogen and CO2
s_{114}	Oxygen	e_{112}	Reform Steam and Methane to Hydrogen and CO2	e_{513}	Burn Hydrogen to Generate Electric Power	e_{714}	Export Oxygen	e_{117}	Reform Steam and Methane to Hydrogen and CO2
s_{115}	Electric Power	e_{212}	Electrolyze Water to Hydrogen and Oxygen	e_{613}	Reform Steam and Methane to Hydrogen and CO2	e_{115}	Burn Natural Gas to Generate Electric Power	e_{217}	Burn Natural Gas to Generate Industrial Heat
s_{116}	Industrial Heat	e_{312}	Import Hydrogen Gas	e_{713}	Electrolyze Water to Hydrogen and Oxygen	e_{215}	Import Electric Power	e_{317}	Burn Natural Gas to Generate Electric Power
s_{117}	CO_2	e_{412}	Burn Hydrogen to Generate Industrial Heat	e_{813}	Export Water	e_{315}	Burn Hydrogen to Generate Electric Power	e_{417}	Export CO2
s_{118}	Heat Loss	e_{512}	Burn Hydrogen to Generate Electric Power	e_{114}	Import Oxygen	e_{415}	Electrolyze Water to Hydrogen and Oxygen	e_{118}	Burn Natural Gas to Generate Electric Power
e_{111}	Regasify Natural Gas	e_{612}	Manufacture Ammonia	e_{214}	Electrolyze Water to Hydrogen and Oxygen	e_{515}	Consume Electric Power	e_{218}	Burn Hydrogen to Generate Electric Power
e_{211}	Import Natural Gas	e_{712}	Transport Hydrogen	e_{314}	Burn Natural Gas to Generate Industrial Heat	e_{116}	Burn Natural Gas to Generate Industrial Heat	e_{318}	Export Heat Loss
e_{311}	Burn Natural Gas to Generate Industrial Heat	e_{113}	Burn Natural Gas to Generate Industrial Heat						

Fig. 6.8: Service nets for the Hydrogen Natural Gas test case.

- Carbon Dioxide, size: $\sigma(S_{l7}) \times \sigma(\mathcal{E}_{l7})$: 1×4.
- Heat Loss, size: $\sigma(S_{l8}) \times \sigma(\mathcal{E}_{l8})$: 1×3.

The services are synchronized with the engineering system capabilities through the service feasibility matrix. The service feasibility matrices are defined for each of the operands:

- Natural Gas, size: $\sigma(\mathcal{E}_{l1}) \times \sigma(\mathcal{E}_S)$: 6×61.
- Hydrogen, size: $\sigma(\mathcal{E}_{l2}) \times \sigma(\mathcal{E}_S)$: 7×61.
- Water, size: $\sigma(\mathcal{E}_{l3}) \times \sigma(\mathcal{E}_S)$: 8×61.
- Oxygen, size: $\sigma(\mathcal{E}_{l4}) \times \sigma(\mathcal{E}_S)$: 7×61.
- Electric Power, size: $\sigma(\mathcal{E}_{l5}) \times \sigma(\mathcal{E}_S)$: 5×61.

- Industrial Heat, size: $\sigma(\mathcal{E}_{l6}) \times \sigma(\mathcal{E}_S)$: 4×61.
- Carbon Dioxide, size: $\sigma(\mathcal{E}_{l7}) \times \sigma(\mathcal{E}_S)$: 4×61.
- Heat Loss, size: $\sigma(\mathcal{E}_{l8}) \times \sigma(\mathcal{E}_S)$: 3×61.

6.6.2 Hetero-functional Graph Theory Dynamic Model

The hetero-functional network dynamics model was introduced in Sec. 6.3. The first element of the dynamic model is the engineering system net, modified to incorporate the device models. The device model matrices D_R^+ and D_R^- have size: $\sigma(L) \times \sigma(P) = 8 \times 219$. The incidence matrices in the engineering system net are modified as noted Eqs. 6.25 and 6.26. The second element of the dynamic model contains the service nets. These are directly adopted from the structural model. The final element of the dynamic model describes the synchronization equations for the coupling of the engineering system net and the service nets. The synchronization matrices $\widehat{\Lambda}_i^+$ and $\widehat{\Lambda}_i^-$ are defined by incorporation of the device models in Eqs. 6.27 and 6.28 and have the same size as the service feasibility matrices as defined in the previous section.

6.6.3 Hetero-functional Network Minimum Cost Flow Program

The definition of the quadratic program follows the description in Sec. 6.4. For this specific test case, the program is defined as follows:

$$\text{minimize } Z = x^T F_{QP} x + f_{QP}^T x \tag{6.83}$$

$$\text{s.t. } A_{QP} x = B_{QP} \tag{6.84}$$

$$D_{QP} x \leq E_{QP} \tag{6.85}$$

$$x \geq 0, \quad x \in \mathbb{R} \tag{6.86}$$

where: x has size: $8,463 \times 1$, F_{QP} has size: $8,463 \times 8,463$, f_{QP} has size: $8,463 \times 1$, A_{QP} has size: $7,323 \times 8,463$, B_{QP} has size: $7,323 \times 1$, D_{QP} has size: $1,281 \times 8,463$, and E_{QP} has size: $1,281 \times 1$. The quadratic cost function, the F_{QP}-matrix, has positive eigenvalues. In combination with the linear constraints, that results in a convex quadratic program. The linear equality constraints matrix A_{QP} consists of block rows that reflect the equality constraints (as introduced in Sec. 6.4.9). These block rows are now discussed in order.

State transition functions for engineering system net are introduced in Sec. 6.4.1. The state transition function for the engineering system net buffers is:

$$-Q_B[k+1] + Q_B[k] + M^+U^+[k] - M^-U^-[k] = 0 \qquad \forall k \in \{1,\ldots,K\} \tag{6.87}$$

The M^+ and M^- are the positive and negative engineering system net incidence matrices, with size 80 by 61, with 62 and 60 filled elements respectively. The state transition function for the engineering system net transitions is:

$$-Q_{\mathcal{E}}[k+1] + Q_{\mathcal{E}}[k] - U^+[k] + U^-[k] = 0 \qquad \forall k \in \{1,\ldots,K\} \tag{6.88}$$

The duration constraint for engineering system net transitions is (as introduced in Sec. 6.4.4):

$$-U^+[k+k_{d\psi}] + U^-[k] = 0 \qquad \forall k \in \{1,\ldots,K\} \tag{6.89}$$

The state transition functions for the system service net is introduced in Sec. 6.4.2. The state transition function for the system service net places is:

$$-Q_{SL}[k+1] + Q_{SL}[k] + M_L^+U_L^+[k] - M_L^-U_L^-[k] = 0 \qquad \forall k \in \{1,\ldots,K\} \tag{6.90}$$

The M_L^+ and M_L^- are the positive and negative system service net incidence matrices, with size: 8 by 44, with 24 and 22 filled elements respectively. The state transition function for the system service net transitions is:

$$-Q_{\mathcal{E}L}[k+1] + Q_{\mathcal{E}L}[k] - U_L^+[k] + U_L^-[k] = 0 \qquad \forall k \in \{1,\ldots,K\} \tag{6.91}$$

The synchronization of the engineering system net and the system services net is achieved through the system services feasibility constraints, as introduced in Sec. 6.4.3. The system services feasibility constraints are:

$$U_L^+[k] - \widehat{\Lambda}^+ U^+[k] = 0 \qquad \forall k \in \{1,\ldots,K\} \tag{6.92}$$

$$U_L^-[k] - \widehat{\Lambda}^- U^-[k] = 0 \qquad \forall k \in \{1,\ldots,K\} \tag{6.93}$$

The matrices $\widehat{\Lambda}^+$ and $\widehat{\Lambda}^-$ are the positive and negative system services feasibility matrices, of size 43 by 61, with 62 and 60 filled elements respectively.

The demand and supply function is introduced in Sec. 6.4.5:

$$\begin{bmatrix} D_{Bp} & \mathbf{0} \\ \mathbf{0} & D_{Bn} \end{bmatrix} \begin{bmatrix} U^+ \\ U^- \end{bmatrix} [k] = \begin{bmatrix} C_{Bp} \\ C_{Bn} \end{bmatrix} [k] \qquad \forall k \in \{1,\ldots,K\} \tag{6.94}$$

Where D_{Bp} has size $1 \times \sigma(Q_{\mathcal{E}})$ with a single filled element, as the program contains a single predetermined import transition (*Import Electric Power* at Node 1). D_{Bn} has size $3 \times \sigma(Q_{\mathcal{E}})$ with three filled elements, as the program contains three predetermined export transitions (*Manufacture Ammonia* at Node 3, *Consume Electric Power* at Node 4, and *Consume Industrial Heat* at Node 5).

The initial and final condition constraints are introduced in Sec. 6.4.6. The initial

condition constraint is defined as:

$$\mathbf{I}^{\sigma(Q_B)+\sigma(Q_{\mathcal{E}})+\sigma(Q_{SL})}\begin{bmatrix} Q_B \\ Q_{\mathcal{E}} \\ Q_{SL} \end{bmatrix}[k=1] = \begin{bmatrix} C_{B1} \\ C_{\mathcal{E}1} \\ C_{SL1} \end{bmatrix} \tag{6.95}$$

Where the identity matrix is adjusted to have a zero value on the diagonal at the import transition.

The final condition constraint is defined as:

$$\mathbf{I}^{\sigma(Q_B)+2\sigma(Q_{\mathcal{E}})+\sigma(Q_{SL})+\sigma(Q_{\mathcal{E}L})}\begin{bmatrix} Q_B \\ Q_{\mathcal{E}} \\ Q_{SL} \\ U^- \\ U_L^- \end{bmatrix}[k=K+1] = \begin{bmatrix} C_{BK} \\ C_{\mathcal{E}K} \\ C_{SLK} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{6.96}$$

Where the identity matrix is adjusted to have zero values on the diagonal at the export transitions.

The linear inequality constraints matrix C_{QP} consists of block rows that constrain the capacity of the transitions in the engineering system net (as introduced in Sec. 6.4.7).

$$\begin{bmatrix} D_{Cp} & \mathbf{0} \\ \mathbf{0} & I^{\sigma(\mathcal{E}_S)} \end{bmatrix}\begin{bmatrix} U^+ \\ U^- \end{bmatrix}[k] \leq C_U \qquad \forall k \in \{1,\ldots,K+1\} \tag{6.97}$$

Finally, the objective function follows the definition in Sec. 6.4.8. In this test case, there is a quadratic cost component for three transitions (*Burn Natural Gas to Generate Electric Power* at Node 1, *Burn Hydrogen to Generate Electric Power* at Node 4, and *Burn Natural Gas to Generate Electric Power* at Node 4) as is common in the power systems literature [331, 332]. Furthermore, for this test case, a value of $1e^{-10}$ is added to all diagonal elements in the quadratic cost matrix F_{QP} to ensure that the matrix is positive definite. All

Table. 6.3: Overview of cost and carbon emissions per scenario

	Scenario 1	**Scenario 2**	**Scenario 3**	**Scenario 4**
Total Cost	$6,092,627.17	$10,452,421.24	$11,777,395.62	$25,244,985.80
Total CO_2 Emissions	47,026.74 ton	45,041.97 ton	33,091.17 ton	0 ton

transitions incur a linear cost as defined in Table 6.1. All costs, both linear and quadratic, are imposed on the negative firing vectors except for the import transitions. Those costs are imposed on the positive firing vectors. The objective function is:

$$Z = x^T F_{QP} x + f_{QP}^T x \tag{6.98}$$

6.6.4 Scenario Results

The final results of this work encompass the optimization of the test case program for the four different scenarios. The optimization program matrices were defined in MATLAB 2019a and solved as a quadratic program using the CONOPT 3 solver in GAMS. All programs were found to be locally optimal in less than 2 seconds when running the program on a MacBook Pro (15-inch, 2017) with a 3.1 GHz Quad-Core Intel Core i7 and 16 GB RAM.

Table 6.3 provides an overview of the total cost and the carbon emissions of each of the four scenarios. Fig. 6.9.a) shows a breakdown of the carbon emissions per resource, Fig. 6.9.b) the natural gas balance (the generation and consumption for each of the resources), and Fig. 6.9.c) the hydrogen balance for the system as a whole. The results of the scenarios are now compared.

Scenario 1 is the least expensive scenario, but it emits the highest level of carbon dioxide. Since there is no renewable energy input to the system, electrolysis is only used to replace steam reformation in time step 3. Steam reformation requires an extra time step to ramp up from a cold start and cannot fulfill the hydrogen demand in time step 4. The demand for industrial heat in the steel mill is satisfied by natural gas as the least cost option.

Scenario 2 imposes a carbon tax of $ 250 per ton CO_2 emitted by the steel mill. Scenario

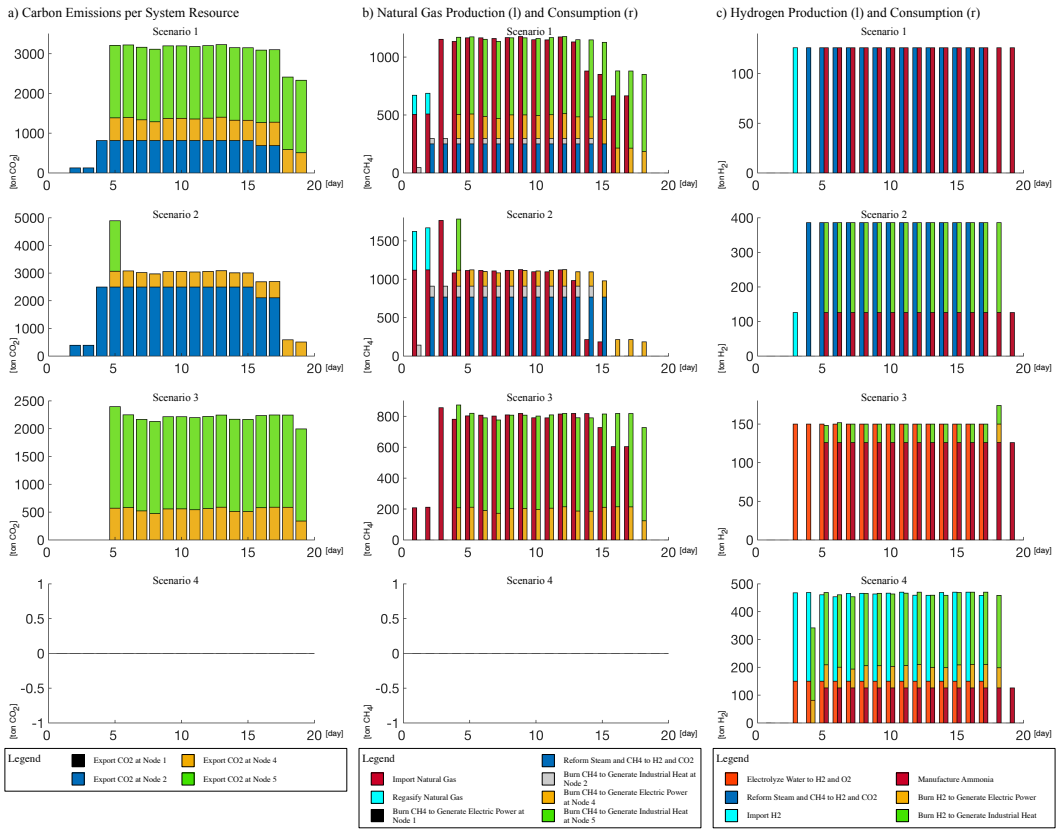


Fig. 6.9: a) Carbon emissions per system resource for all time steps. b) Natural Gas Production (left column) vs. Natural Gas Consumption (right column) for all time steps. c) Hydrogen Production (left column) vs. Hydrogen Consumption (right column) for all time steps.

2 is 72% more expensive than scenario 1, while emitting 4% less carbon dioxide. The steel mill sources almost all of its industrial heat from hydrogen to avoid the carbon tax. Its hydrogen supply is produced by the SMR process and causes a substantial increase of carbon emissions at the SMR facility relative to scenario 1. Not all industrial heat is satisfied by hydrogen, as the capacity of Hydrogen Pipe Line 4 is insufficient. The remainder of the industrial heat is supplied by natural gas combustion, as the imported hydrogen is more expensive than the combination of imported natural gas and a carbon tax.

Scenario 3 incorporates a predetermined supply of renewable electricity to the system. The electricity cannot be transported and forces the production of hydrogen through electrolysis. The total carbon dioxide emissions are 30% lower than in scenario 1. The total cost

of scenario 3 is 93% higher than scenario 1 and 13% higher than scenario 2. The hydrogen through electrolysis is predominantly used to supply the ammonia facility and the left-overs are used to produce industrial heat in the steel mill. Natural gas is used to provide the bulk of the industrial heat in the steel mill as the least cost option.

Scenario 4 combines the renewable electricity supply with a carbon tax of $500 per ton CO_2 at all locations. This results in a cost increase of 314% over scenario 1 and zero carbon emissions (within the boundaries of this system). As the use of natural gas is clearly too expensive in this scenario, the supply of hydrogen is satisfied by the least cost routing of the hydrogen. The steel mill is a single transportation process removed from the hydrogen import facility and therefore, it receives predominantly imported hydrogen. Hydrogen Pipe Line 6 reaches its capacity limit as a result.

From the optimization results of these four scenarios, it is clear that the hetero-functional network minimum cost flow program enables the optimization of a continuous flow multi-operand system over time with storage of operands, transformation of operands, and the explicit description of the state of operands. This holistic program enables the user to study trade-offs and synergies in the behavior of interdependent systems.

6.7 Conclusion

This work set out to define a hetero-functional network minimum cost flow optimization program that enables the optimization of large flexible engineering systems across multiple types of operands. This program is the first of its kind, as it is the first hetero-functional graph theory-based optimization program.

In the process of developing the first hetero-functional network minimum cost flow optimization program, this work has established the first formal connection between the Hetero-functional Incidence Tensor, arc-constant Colored Petri nets, and the Engineering System Net. Furthermore, it has defined the first integration of device models to the

feasibility matrices that couple the engineering system net and the system services net. Additionally, the implementation of the hetero-functional network minimum cost flow optimization program accommodates the explicit definition of time and therefore storage. Moreover, the program accommodates both linear and quadratic optimization of such a dynamic, hetero-functional network model. Finally, the demonstration of the hetero-functional network minimum cost flow program in this paper has lead to the definition of the first hydrogen-natural gas infrastructure test case.

Chapter Summary:

This chapter successfully developed a hetero-functional network minimum cost flow program for engineering systems. The chapter leveraged the hetero-functional graph theory structural model and Petri net dynamics to develop a continuous flow dynamic system model. The chapter then translated this dynamic system model to the quadratic program canonical form. The application of the program to various policy scenarios in the hydrogen-natural gas test case demonstrated that the program may be implemented for engineering systems of arbitrary topology and that the program specifically includes the synergies and trade-offs of interdependent engineering systems. ■

Chapter 7

Conclusion and Future Work

This dissertation set out to advance a Hetero-functional Graph Theory for engineering systems to study their structure, behavior, and optimization. This chapter provides the concluding remarks to the dissertation. Section 7.1 first discusses the outcomes to each of the research questions and the thesis statement, as introduced in Section 1.5. Then, Section 7.2 provides an overview of the contributions by the work in this dissertation to the engineering systems and hetero-functional graph theory literature. Finally, Section 7.3 seeks to provide insight in both the limitations and the future work related to this dissertation.

7.1 Conclusion

The thesis statement of this dissertation is defined in Section 1.5 as:

Thesis Statement: *A Hetero-functional Graph Theory provides a novel approach to modeling the structure of large flexible engineering systems such that it enables simulation and optimization of the behavior of such systems.*

In order to investigate this statement, four research questions were posed. This section addresses each of these questions and provides a number of concluding remarks.

7.1.1 Research Question 1

Why are existing modeling frameworks of engineering system structure inadequate?
Existing modeling frameworks of engineering system structure cannot describe all engineering systems adequately as a result of their ontological insufficiencies. This conclusion is reached based on a review of the literature in Chapter 2.

The context to this conclusion is a comparison of the experimental, data-driven, and model-based approach for engineering system analysis. Though these approaches are complementary, it was concluded that the model-based approach deserves specific attention as it provides insight towards both behavioral and structural interventions while it is relatively affordable (Section 2.1.3). Within the set of model-based approaches, quantitative structural models were found to be especially useful towards the study of engineering system structure, as they provide quantitative insight and are potentially generalizable across engineering disciplines to provide holistic models of engineering systems (Section 2.2.1).

As existing multilayer networks approaches have ontological insufficiencies that result in limitations as to which engineering systems can be modeled, there is a need for an alternative quantitative structural modeling approach (Section 2.2.2). Multilayer network approaches impose at least one of eight constraints, as identified by Kivelä, that limit the type of system that can be modeled [95]. The theory of ontology was then used to establish that multilayer networks are neither *"Complete"* nor *"Lucid"*: multilayer networks neither have enough modeled elements to represent all conceptual elements, nor do their conceptual elements have unique representations in the modeling elements (Section 2.3.1).

The work to answer this research question draws upon Chapter 23 in the *"Handbook of Engineering System Design"* [48] and Chapters 2 and 3 in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5].

7.1.2 Research Question 2

What is an ontologically clear structural model of a hetero-functional engineering system?

Hetero-functional graph theory provides an ontologically clear structural model of a hetero-functional engineering system.

This conclusion is reached through a theoretical exposition (Chapter 3) and two demonstrations of hetero-functional graph theory (Chapter 4). The theoretical exposition develops hetero-functional graph theory as an intellectual fusion of model-based systems engineering and network science. Hetero-functional graph theory consists of seven constituent mathematical models that are all related to counterparts in SysML. The definition of this relationship assumes SysML to be an accurate representation of the Domain Conceptualization and establishes hetero-functional graph theory as the Language that represents it. Consequently, hetero-functional graph theory is established as an ontologically clear structural model of a hetero-functional engineering system.

The demonstration of hetero-functional graph theory for two hetero-functional engineering system test cases serves two goals: first, it seeks to illustrate the use of hetero-functional graph theory and second, it shows that hetero-functional graph theory is able to describe a system that violates all eight limitations on the modeled system as imposed by multilayer networks.

The theoretical exposition of hetero-functional graph theory draws upon Chapter 4 in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5]. The demonstration of hetero-functional graph theory draws upon Chapter 5 and Appendix A in the book *"A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure"* [5].

7.1.3 Research Question 3

How can a Hetero-functional Graph Theory structural model be revised with dynamic behavior so as to simulate an engineering system?

A hetero-functional graph theory structural model provides the foundation for the development of a dynamic model (Chapter 5). Device models describe the behavior of the degrees of freedom. Subsequently, the device models are coupled based on the sequence-dependent degrees of freedom (represented in Chapter 5 with incidence matrices). This coupled system of device models describes the holistic, dynamic behavior of the engineering system.

The development of the dynamic model for engineering systems is applied specifically to a microgrid-enabled production system. The model leverages petri net dynamics to describe the dynamics of the production system degrees of freedom. For the microgrid dynamics, the work finds that the hetero-functional graph theory structural model results in a dynamic model that is entirely consistent with traditional power flow analysis. Furthermore, the work simulates a test case of a microgrid-enabled production system that tracks carbon emissions as a proxy for sustainability (a *life-cycle property*). The simulations show that the integration of renewable energy resources into the microgrid-enabled production system have the potential to reduce carbon emissions, but that care should be taken as the generation of renewable electricity does not necessarily correspond to the load imposed by the production system dynamics.

This work draws from a 2017 journal article entitled *"A Dynamic Model for the Energy Management of Microgrid-Enabled Production Systems"* in the *Journal of Cleaner Production* [35].

7.1.4 Research Question 4

How can a Hetero-functional Graph Theory dynamic model be optimized?

The definition of an optimization program for a hetero-functional graph theory-based dynamic model requires the reformulation of the system of dynamic equations (Chapter

6). First, the state transition functions are reformulated as equality constraints. Second, physical limitations and other external parameters to the dynamic model are reformulated as constraints: boundary conditions, demand and supply constraints, and capacity constraints. Finally, an objective function is formulated such that the program minimizes cost over time while satisfying all the constraints.

This work relates back to the development of the hetero-functional graph theory-based dynamic model and work that has been developed in parallel to this dissertation on the topic of the hetero-functional incidence tensor. This work establishes the relationship between arc-constant colored Petri nets and the hetero-functional incidence tensor to solidify the relationship between existing theory for dynamic models (Petri nets) and a novel structural modeling approach (hetero-functional graph theory). The first integration of device models to the system service feasibility matrices couple the engineering system net dynamics to the operand behavior.

The demonstration of the hetero-functional network minimum cost flow optimization program to a hydrogen-natural gas infrastructure system finds that the program explicitly defines time and enables storage of operands. This demonstration also shows that the hetero-functional graph theory foundation to the optimization program enables the optimization of an integrated, multi-disciplinary system.

7.1.5 Concluding Remarks

In conclusion, the dissertation has advanced a hetero-functional graph theory. First, it established a need for a novel approach to modeling engineering systems as the inadequacies of other methods were explored. Then, hetero-functional graph theory was found to be a novel, ontologically clear quantitative structural modeling framework for representing engineering systems. Thereafter, it was shown that hetero-functional graph theory serves as a foundation for the development of a dynamic system model for engineering systems. Finally, the dissertation has shown how to develop an optimization program for hetero-functional

graph theory based dynamic models. Hetero-functional graph theory is shown to provide a novel approach to modeling the structure of large flexible engineering systems such that it enables simulation and optimization of the behavior of such systems.

7.2 Contribution

This section provides an overview of the research contributions that have resulted from answering the four research questions discussed above. The contributions of this thesis are classified in three main areas: contributions towards 1) ... the context of modeling for engineering systems, 2) ... the advancement of a Hetero-functional Graph Theory, and 3) ... the application domains of a Hetero-functional Graph Theory. The contributions of the thesis are:

1. **Contextual contributions:** this dissertation provides ...
    - ... the first high level overview of evaluation methods for engineering systems in Section 2.2.1.
    - ... the first demonstration that modeling limitations by the multilayer network community are restricting which engineering systems can be modeled in Section 2.2.2.
    - ... the first discussion as to why existing multilayer network models are inadequate to model engineering systems, with the argument rooted in ontological sciences in Section 2.3.1.
2. **Contributions towards the advancement of a Hetero-functional Graph Theory:** this dissertation provides ...
    - ... the first argument that Hetero-functional Graph Theory provides an ontologically clear structural model of a hetero-functional engineering system in Chapter 3.

- ... the first argument that explicitly couples systems engineering modeling concepts to Hetero-functional Graph Theory in Chapter 3.
- ... the first complete and consistent overview of Hetero-functional Graph Theory in Chapter 3.
- ... the first introduction of the following mathematical elements:
  - Transportation processes of transformative nature in Section 3.1.
  - The controller adjacency matrix in Section 3.4.
  - The system adjacency matrix in Section 3.7.
- ... first definition of an optimization program for a Hetero-functional Graph Theory based dynamic engineering system model in Chapter 6.

3. **Contributions towards the application domains of a Hetero-functional Graph Theory:** this dissertation provides ...
   - ... the first Hetero-functional Graph Theory based model for three coupled infrastructure systems through the Trimetrica test case in Chapter 4.
   - ... the first Hetero-functional Graph Theory based model for a four layer system with three infrastructures and a control layer in Section 4.11.
   - ... the first Hetero-functional Graph Theory based model for a dynamic energy management of a microgrid-enabled production system in Chapter 5.
   - ... the first definition of a dynamic model for a Hydrogen-Natural Gas network in Chapter 6.

7.3 Limitations and Future Work

This dissertation has consolidated and advanced hetero-functional graph theory as a novel methodology for modeling engineering system structure, behavior, and optimal control.

This dissertation may therefore be used as a starting point for further research and the development of more advanced tools in the areas of engineering system structure, behavior, and optimization. Based on this work and other recent advances in hetero-functional graph theory, this thesis identifies three main areas of future development.

The first area of future work is the development of more and better measures of life-cycle properties. As noted in Chapter 2, life-cycle properties are essential to understand the socio-technical nature of engineering systems. Hetero-functional graph theory may serve to support further quantitative understanding of life-cycle properties as its description of engineering systems has a more expansive feature set than existing modeling approaches.

The initial focus of hetero-functional graph theory was the development of measures for reconfigurability and modularity of manufacturing systems [6, 7, 20]. More recently, there has been a push to understanding resilience of energy systems [8, 10, 11, 14, 28, 29]. Furthermore, Prof. Khayal's healthcare system research has leveraged hetero-functional graph theory in combination with statistics to create a deeper understanding of a patient's health state and enable personalized healthcare delivery [39, 44, 333]. It therefore appears that hetero-functional graph theory is generalizable beyond the disciplines in this thesis (as conventional graph theory is). The use of hetero-functional graph theory in other fields may lead to the development of new measures for life-cycle properties and the conversion of those measures to create insight in multiple other disciplines.

The second area of future work entails the development of better and more extensive toolchains for hetero-functional graph theory. Currently, the use of hetero-functional graph theory requires an extensive and lengthy modeling process. This is a direct result of the need to accommodate heterogeneity in the described systems. However, this complex process may be prohibitive to some users and it is desirable to lower the barrier of entry to using hetero-functional graph theory. To that end, the hetero-functional graph theory toolbox has been published on GitHub, but this is only a small part of process [334].

Examples of such novel tools are an integrated SysML and hetero-functional graph

theory software package, where SysML diagrams can be directly translated to hetero-functional graphs. Furthermore, software exists to develop extensive dynamic models (for example Modelica). An avenue of future work could thus be the integration of such dynamic modeling software such that is leverages a hetero-functional graph theory structural model. A natural extension of that is the development of an optimization package that integrates the hetero-functional graph theory-based dynamic model.

A final aspect of increasing the adoption of hetero-functional graph theory is the need for more expansive pedagogical tools. Currently, hetero-functional graph theory is taught at a graduate or advanced undergraduate level, or through a book. At some schools, it may be possible to integrate the fundamentals of hetero-functional graph theory into introductory courses such as “systems thinking." One could also leverage alternative or new media channels, such as videos or interactive online tutorials.

The third area of future work focuses on the construction of bridges to other engineering systems analysis approaches. The inspiration for this third area of work is drawn from Kivelä et. al. [95], who, for the work in multilayer networks, emphasize the need *“to unify the various disparate threads and to discern their similarities and differences in as precise a manner as possible."* There is a clear need to converge the approaches to analyzing engineering systems, while respecting the unique perspectives that they all offer. The construction of bridges to other fields enables researchers to value the inputs and contributions of those respective fields.

This dissertation has established bridges to several other fields. Chapters 2 and 3 emphasize the connections between hetero-functional graph theory, model-based systems engineering, and graph theory. Recent work by Farid et. al. [52] further reinforces the similarities and differences between hetero-functional graph theory and multilayer networks through a comparison of tensor-based formulations. Chapter 6 has established the relationship between the discrete-event dynamics community through Petri nets and the Operations Research community through a quadratic program. However, more bridges need to be build.

Examples include bridges to the fields of System Dynamics, Bond Graphs, and Linear Graphs as distinct approaches to developing dynamic models of engineering systems.

This last avenue of future work reinforces the need for a convergence of theory to describe engineering systems. Hopefully this dissertation contributes to building bridges and converging the theory.

Bibliography

[1] Olivier L De Weck, Daniel Roos, and Christopher L Magee. *Engineering systems: meeting human needs in a complex technological world*. MIT Press, Cambridge, Mass., 2011.

[2] A M Farid. *Reconfigurability Measurement in Automated Manufacturing Systems*. Ph.d. dissertation, University of Cambridge Engineering Department Institute for Manufacturing, 2007.

[3] N P Suh. *Axiomatic Design: Advances and Applications*. Oxford University Press, 2001.

[4] Amro M. Farid and Nam P Suh. *Axiomatic Design in Large Systems: Complex Products, Buildings and Manufacturing Systems*. Springer, Berlin, Heidelberg, 2016.

[5] Wester C.H. Schoonenberg, Inas S. Khayal, and Amro M. Farid. *A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure*. Springer, Berlin, Heidelberg, 2018.

[6] Amro M. Farid. Product Degrees of Freedom as Manufacturing System Reconfiguration Potential Measures. *International Transactions on Systems Science and Applications – invited paper*, 4(3):227–242, 2008.

[7] A M Farid. Facilitating ease of system reconfiguration through measures of manufacturing modularity. *Proceedings of the Institution of Mechanical Engineers, Part B (Journal of Engineering Manufacture) – invited paper*, 222(B10):1275–1288, 2008.

[8] Amro M. Farid. An Axiomatic Design Approach to Non-Assembled Production Path Enumeration in Reconfigurable Manufacturing Systems. In *2013 IEEE International Conference on Systems Man and Cybernetics*, pages 1–8, Manchester, UK, 2013.

[9] Amro M Farid. Axiomatic Design and Design Structure Matrix Measures for Reconfigurability and Its Key Characteristics in Automated Manufacturing Systems. In *International Conference on Axiomatic Design*, pages 1–8, Campus de Caparica, Portugal, 2014.

[10] Amro M Farid. Static Resilience of Large Flexible Engineering Systems: Part I – Axiomatic Design Model. In *4th International Engineering Systems Symposium*, pages 1–8, Hoboken, N.J., 2014. Stevens Institute of Technology.

[11] Amro M Farid. Static Resilience of Large Flexible Engineering Systems: Part II – Axiomatic Design Measures. In *4th International Engineering Systems Symposium*, pages 1–8, Hoboken, N.J., 2014. Stevens Institute of Technology.

[12] Amro M. Farid. Designing multi-agent systems for resilient engineering systems (invited paper). In *Holomas 2015 7th International Conference on Industrial Applications of Holonic and Multi-Agent Systems*, pages 1–6, Valencia, Spain, 2015.

[13] A M Farid and Wutthiphat Covanich. Measuring the Effort of a Reconfiguration Process. In *Emerging Technologies and Factory Automation, 2008. ETFA 2008. IEEE International Conference on*, pages 1137–1144, Hamburg, Germany, 2008.

[14] Amro M Farid. Static Resilience of Large Flexible Engineering Systems: Axiomatic Design Model and Measures. *IEEE Systems Journal*, PP(99):1–12, 2015.

[15] Amro M Farid. Measures of Reconfigurability and Its Key Characteristics in Intelligent Manufacturing Systems. *Journal of Intelligent Manufacturing*, 28(2):353–369, 2017.

[16] A M Farid and D C McFarlane. An Approach to the Application of the Design Structure Matrix for Assessing Reconfigurability of Distributed Manufacturing Systems. In *Proceedings of the IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications (DIS'06)*, pages 1–6, Prague, Czech republic, 2006.

[17] Amro M. Farid and Duncan C McFarlane. A Development of Degrees of Freedom for Manufacturing Systems. In *IMS'2006: 5th International Symposium on Intelligent Manufacturing Systems: Agents and Virtual Worlds*, pages 1–6, Sakarya, Turkey, 2006.

[18] Amro M Farid and Duncan C McFarlane. A tool for assessing reconfigurability of distributed manufacturing systems. In *12th IFAC Symposium on Information Control Problems in Manufacturing,*, volume 12, pages 1–6, Saint Etienne, France, 2006.

[19] A M Farid and D C McFarlane. A Design Structure Matrix Based Method for Reconfigurability Measurement of Distributed Manufacturing Systems. *International Journal of Intelligent Control and Systems Special Issue – invited paper*, 12(2):118–129, 2007.

[20] A M Farid and D C McFarlane. Production degrees of freedom as manufacturing system reconfiguration potential measures. *Proceedings of the Institution of Mechanical Engineers, Part B (Journal of Engineering Manufacture) – invited paper*, 222(B10):1301–1314, 2008.

[21] Amro M Farid and Luis Ribeiro. An Axiomatic Design of a Multi-Agent Reconfigurable Manufacturing System Architecture. In *International Conference on Axiomatic Design*, pages 1–8, Lisbon, Portogal, 2014.

[22] Amro M Farid and Luis Ribeiro. An Axiomatic Design of a Multi-Agent Reconfigurable Mechatronic System Architecture. *IEEE Transactions on Industrial Informatics*, 11(5):1142–1155, 2015.

[23] Edgar Eugenio Samano Baca and Amro M Farid. An Axiomatic Design Approach to Reconfigurable Transportation Systems Planning and Operations (invited paper). In *DCEE 2013: 2nd International Workshop on Design in Civil & Environmental Engineering*, pages 22–29, Worcester, MA, USA, 2013.

[24] Edgar Eugenio Samano Baca, Amro M. Farid, I-Tsung Tsai, and Asha Viswanath. An Axiomatic Design Approach to Passenger Itinerary Enumeration in Reconfigurable Transportation Systems. In *Proceedings of ICAD2013 The Seventh International Conference on Axiomatic Design*, volume PP, pages 1–10, Worcester, MA, USA, 2013.

[25] Asha Viswanath, Edgar Eugenio Samano Baca, and Amro M Farid. An Axiomatic Design Approach to Passenger Itinerary Enumeration in Reconfigurable Transportation Systems. *IEEE Transactions on Intelligent Transportation Systems*, 15(3):915 – 924, 2014.

[26] Amro M. Farid. Multi-Agent System Design Principles for Resilient Operation of Future Power Systems. In *IEEE International Workshop on Intelligent Energy Systems*, pages 1–7, San Diego, CA, 2014.

[27] Amro M Farid. Multi-Agent System Design Principles for Resilient Coordination and Control of Future Power Systems. *Intelligent Industrial Systems*, 1(3):255–269, 2015.

[28] Dakota Thompson, Wester C.H. Schoonenberg, and Amro M. Farid. A Heterofunctional Graph Analysis of Electric Power System Structural Resilience. In *IEEE Innovative Smart Grid Technologies Conference North America*, pages 1–5, Washington, DC, United states, 2020.

[29] Dakota Thompson, Wester C.H. Schoonenberg, and Amro M. Farid. A Heterofunctional Graph Resilience Analysis of the Future American Electric Power System. *submitted to: IEEE Access*, 1(1):11, 2020.

[30] Amro M. Farid. Electrified transportation system performance: Conventional vs. online electric vehicles. In Nam P Suh and Dong Ho Cho, editors, *The On-line Electric Vehicle: Wireless Electric Ground Transportation Systems*, chapter 20, pages 279–313. Springer, Berlin, Heidelberg, 2017.

[31] Amro M. Farid. A Hybrid Dynamic System Model for Multi-Modal Transportation Electrification. *IEEE Transactions on Control System Technology*, PP(99):1–12, 2016.

[32] Thomas JT van der Wardt and Amro M Farid. A hybrid dynamic system assessment methodology for multi-modal transportation-electrification. *Energies*, 10(5):653, 2017.

[33] Asha Viswanath and Amro M. Farid. A Hybrid Dynamic System Model for the Assessment of Transportation Electrification. In *American Control Conference 2014*, pages 1–7, Portland, Oregon, 2014. IEEE.

[34] Wester C.H. Schoonenberg and Amro M. Farid. A dynamic production model for industrial systems energy management. In *2015 IEEE International Conference on Systems Man and Cybernetics*, pages 1–7, Hong Kong, 2015.

[35] Wester C.H. Schoonenberg and Amro M. Farid. A Dynamic Model for the Energy Management of Microgrid-Enabled Production Systems. *Journal of Cleaner Production*, 1(1):1–10, 2017.

[36] Amro M. Farid and Inas S Khayal. Axiomatic Design Based Volatility Assessment of the Abu Dhabi Healthcare Labor Market: Part I-Theory. In *Proceedings of the ICAD 2013: The Seventh International Conference on Axiomatic Design*, pages 1–8, Worcester, MA, USA, 2013.

[37] Inas S. Khayal and Amro M. Farid. Axiomatic Design Based Volatility Assessment of the Abu Dhabi Healthcare Labor Market: Part II – Case Study. In *Proceedings of the ICAD 2013: The Seventh International Conference on Axiomatic Design*, pages 1–8, Worcester, MA, USA, 2013.

[38] Inas S Khayal and Amro M. Farid. Axiomatic Design Based Volatility Assessment of the Abu Dhabi Healthcare Labor Market. *Journal of Enterprise Transformation*, 5(3):162–191, 2015.

[39] Inas S. Khayal and Amro M. Farid. The Need for Systems Tools in the Practice of Clinical Medicine. *Systems Engineering*, 20(1):3–20, 2016.

[40] IS Khayal and Amro M. Farid. The Application of Model-Based Systems Engineering to the Practice of Clinical Medicine. In *Proceedings of the 11th Annual IEEE International Systems Conference*, pages 1–5, Montreal, Quebec, Canada, April 2017.

[41] Inas Khayal and Amro M. Farid. A Dynamic Model for a Cyber-Physcal Healthcare Delivery System with Human Agents. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC2017), Intelligent Industrial System Special Session*, pages 1–6, Banff, Canada, October 2017.

[42] Inas Khayal and Amro Farid. An Architecture for a Cyber-Physical Healthcare Delivery System with Human Agents. In *Proceedings of the 2017 IEEE International Summer School on Smart Cities (IEEE S3C)*, pages 1–8, Natal, Brazil, August 2017.

[43] Inas Khayal and Amro Farid. Designing Smart Cities for Citizen Health and Wellbeing. In *Proceedings of the 2017 IEEE International Summer School on Smart Cities (IEEE S3C)*, pages 1–6, Natal, Brazil, August 2017.

[44] Inas S. Khayal and Amro M. Farid. Architecting a System Model for Personalized Healthcare Delivery and Managed Individual Health Outcomes. *Complexity*, 1(1):1–25, 2018.

[45] Inas Khayal and Amro Farid. Designing Patient-Oriented Healthcare Services as Systems of Systems. In *2018 IEEE Systems of Systems Conference*, pages 1–7, Paris, France, 2018.

[46] Inas S. Khayal and Amro M. Farid. Healthcare system deisgn. In *Design Engineering and Science*, pages 1–14. Springer, Berlin, Heidelberg, 2019.

[47] Amro M. Farid. LIINES Website. In *Laboratory for Intelligent Integrated Networks of Engineering Systems*, page 1, Hanover, NH, USA, 2017.

[48] Wester C.H. Schoonenberg and Amro M. Farid. Evaluating engineering system interventions (in press). In *Handbook of Engineering System Design*, pages 1–20. Springer, Berlin, Heidelberg, 2020.

[49] Giancarlo Guizzardi. On ontology, ontologies, conceptualizations, modeling languages, and (meta) models. *Frontiers in artificial intelligence and applications*, 155:18, 2007.

[50] Amro M. Farid. Symmetrica: Test Case for Transportation Electrification Research. *Infrastructure Complexity*, 2(9):1–10, 2015.

[51] Hadi Saadat. *Saadat - 1999 - Power System Analysis.pdf*. McGraw-Hill, 1999.

[52] Amro M. Farid, Dakota Thompson, Prabhat Hegde, and Wester C.H. Schoonenberg. A Tensor-Based Formulation of Hetero-functional Graph Theory. *submitted to: Physics Review X*, 1(1):25, 2020.

[53] Michael Batty. The size, scale, and shape of cities. *science*, 319(5864):769–771, 2008.

[54] Jie Cheng, Qishuai Liu, Qing Hui, and Fred Choobineh. The joint optimization of critical interdependent infrastructure of an electricity-water-gas system. *arXiv preprint arXiv:1803.02692*, 2018.

[55] Stefano De Porcellinis, Roberto Setola, Stefano Panzieri, and Giovanni Ulivi. Simulation of heterogeneous and interdependent critical infrastructures. *International Journal of Critical Infrastructures*, 4(1-2):110–128, 2008.

[56] Irene Eusgeld, Cen Nan, and Sven Dietz. "system-of-systems" approach for interdependent critical infrastructures. *Reliability Engineering & System Safety*, 96(6):679–686, 2011.

[57] Constantinos Heracleous, Panayiotis Kolios, Christos G Panayiotou, Georgios Ellinas, and Marios M Polycarpou. Hybrid systems modeling for critical infrastructures interdependency analysis. *Reliability Engineering & System Safety*, 165:89–101, 2017.

[58] Giuliano Andrea Pagani and Marco Aiello. The power grid as a complex network: a survey. *Physica A: Statistical Mechanics and its Applications*, 392(11):2688–2700, 2013.

[59] Steven M Rinaldi, James P Peerenboom, and Terrence K Kelly. Identifying, understanding, and analyzing critical infrastructure interdependencies. *IEEE Control Systems*, 21(6):11–25, 2001.

[60] S. M. Rinaldi. Modeling and simulating critical infrastructures and their interdependencies. In *System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on*, pages 8 pp.–, Jan 2004.

[61] V. Rosato, L. Issacharoff, F. Tiriticco, S. Meloni, S. Porcellinis, and R. Setola. Modelling interdependent infrastructures using interacting dynamical models. *International Journal of Critical Infrastructures*, 4(1-2):63–79, 2008.

[62] Saeid Saidi, Lina Kattan, Poornima Jayasinghe, Patrick Hettiaratchi, and Joshua Taron. Integrated infrastructure systems—a review. *Sustainable Cities and Society*, 2017.

[63] Igor Linkov and José Manuel Palma-Oliveira. *Resilience and risk: Methods and application in environment, cyber and social domains*. Springer, 2017.

[64] Priscilla P. Nelson and Raymond L. Sterling. Sustainability and resilience of underground urban infrastructure: New approaches to metrics and formalism. *GeoCongress 2012*, pages 3199–3208, Mar 2012.

[65] Milan Janić. *Advanced Transport Systems*. Springer, 2014.

[66] Sarah M Kaufman, Carson Qing, Nolan Levenson, Melinda Hanson, et al. Transportation during and after hurricane sandy. Technical report, Rudin Center for Transportation Policy & Management, 2012.

[67] Dennis M Buede. *The engineering design of systems: models and methods*. John Wiley & Sons, Hoboken, N.J., 2nd edition, 2009.

[68] Sanford Friedenthal, Alan Moore, and Rick Steiner. *A Practical Guide to SysML: The Systems Modeling Language*. Morgan Kaufmann, Burlington, MA, 2nd edition, 2011.

[69] Tim. Weilkiens. *Systems engineering with SysML/UML modeling, analysis, design*. Morgan Kaufmann, Burlington, Mass., 2007.

[70] OMG. Omg sysml specification. *OMG Systems Modeling Language*, 2006.

[71] Jovica V Milanovic and Wentao Zhu. Modelling of interconnected critical infrastructure systems using complex network theory. *IEEE Transactions on Smart Grid*, 2017.

[72] Jesús Gómez-Gardeñes, Irene Reinares, Alex Arenas, and Luis Mario Floría. Evolution of cooperation in multiplex networks. *Scientific reports*, 2, 2012.

[73] Richard G Morris and Marc Barthelemy. Interdependent networks: the fragility of control. *Scientific reports*, 3, 2013.

[74] Sergey V Buldyrev, Roni Parshani, Gerald Paul, H Eugene Stanley, and Shlomo Havlin. Catastrophic cascade of failures in interdependent networks. *Nature*, 464(7291):1025, 2010.

[75] Walid Najjar and Jean-Luc Gaudiot. Network resilience: A measure of network fault tolerance. *Computers, IEEE Transactions on*, 39(2):174–181, 1990.

[76] John C Whitson and Jose Emmanuel Ramirez-Marquez. Resiliency as a component importance measure in network reliability. *Reliability Engineering & System Safety*, 94(10):1685–1693, October 2009.

[77] Petter Holme, Beom Jun Kim, Chang No Yoon, and Seung Kee Han. Attack vulnerability of complex networks. *Phys. Rev. E*, 65(5):1–14, May 2002.

[78] W. H. Ip and Dingwei Wang. Resilience and friability of transportation networks: Evaluation, analysis and optimization. *IEEE Systems Journal*, 5(2):189–198, Jun 2011.

[79] Réka Albert, Hawoong Jeong, and Albert-László Barabási. Error and attack tolerance of complex networks. *Nature*, 406(6794):378–382, Jul 2000.

[80] F K Hwang, W Najjar, and J L Gaudiot. Comments on Network resilience: a measure of network fault tolerance [and reply]. *IEEE Transactions on Computers*, 43(12):1451–1453, December 1994.

[81] F Harary and J P Hayes. Edge fault tolerance in graphs. *Networks*, 23(2):135–142, mar 1993.

[82] Daniel J. Rosenkrantz, Sanjay Goel, S. S. Ravi, and Jagdish Gangolly. Resilience metrics for service-oriented networks: A service allocation approach. *IEEE Transactions on Services Computing*, 2(3):183–196, Jul 2009.

[83] R. M. Salles and D. A. Marino. Strategies and metric for resilience in computer networks. *The Computer Journal*, 55(6):728–739, Oct 2011.

[84] Royce Francis and Behailu Bekera. A metric and frameworks for resilience analysis of engineered and infrastructure systems. *Reliability Engineering and System Safety*, 121:90–103, Jan 2014.

[85] Devanandham Henry and Jose Emmanuel Ramirez-Marquez. Generic metrics and quantitative approaches for system resilience as a function of time. *Reliability Engineering and System Safety*, 99:114–122, Mar 2012.

[86] Kash Barker, Jose Emmanuel Ramirez-Marquez, and Claudio M. Rocco. Resilience-based network component importance measures. *Reliability Engineering and System Safety*, 117:89–97, Sep 2013.

[87] Benny Sudakov and V H Vu. Local resilience of graphs *. *Random Structures and Algorithms*, pages 409–433, 2008.

[88] Charles J. Colbourn. Network resilience. *SIAM. J. on Algebraic and Discrete Methods*, 8(3):404–409, Jul 1987.

[89] J. Ash and D. Newth. Optimizing complex networks for resilience against cascading failure. *Physica A: Statistical Mechanics and its Applications*, 380:673–683, July 2007.

[90] Duncan S. Callaway, M. E. J. Newman, Steven H. Strogatz, and Duncan J. Watts. Network robustness and fragility: Percolation on random graphs. *Phys. Rev. Lett.*, 85:5468–5471, Dec 2000.

[91] Toni R Farley and Charles J Colbourn. Multiterminal measures for network reliability and resilience. *7th International Workshop on Design of Reliable Communication Networks*, pages 107–114, 2009.

[92] F.K. Hwang, W. Najjar, and J.L. Gaudiot. Comments on "network resilience: a measure of network fault tolerance" [with reply]. *IEEE Transactions on Computers*, 43(12):1451–1453, Dec 1994.

[93] M Newman. *Networks: An Introduction*. Oxford University Press, Oxford, United Kingdom, 2009.

[94] Manlio De Domenico, Albert Solé-Ribalta, Emanuele Cozzo, Mikko Kivelä, Yamir Moreno, Mason A Porter, Sergio Gómez, and Alex Arenas. Mathematical formulation of multilayer networks. *Physical Review X*, 3(4):041022, 2013.

[95] Mikko Kivelä, Alex Arenas, Marc Barthelemy, James P Gleeson, Yamir Moreno, and Mason A Porter. Multilayer networks. *Journal of complex networks*, 2(3):203–271, 2014.

[96] Charles Henry Edwards, David E Penney, and David Calvis. *Differential equations and linear algebra*. Pearson, 2018.

[97] Bohdan T Kulakowski, John F Gardner, and J Lowen Shearer. *Dynamic modeling and control of engineering systems*. Cambridge University Press, 2007.

[98] Amro M. Farid. An engineering systems introduction to axiomatic design. In Amro M. Farid and Nam P Suh, editors, *Axiomatic Design in Large Systems: Complex Products, Buildings & Manufacturing Systems*, chapter 1, pages 1–47. Springer, Berlin, Heidelberg, 2016.

[99] Merriam-Webster Dictionary. Merriam-webster, 2019.

[100] Derek Rowell and D N Wormley. *System dynamics: an introduction*. Prentice Hall, Upper Saddle River, NJ, 1997.

[101] Ludwik Finkelstein. Theory and philosophy of measurement. *Handbook of measurement science*, 1:1–30, 1982.

[102] L Finkelstein. Problems of measurement in soft systems. *Measurement*, 38(4):267–274, 2005.

[103] Eduard Čech and M Katětov. General metric spaces. *Point sets*, 1969.

[104] SE Handbook Working Group. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. International Council on Systems Engineering (INCOSE), 2015.

[105] Alexander J Felson and Steward TA Pickett. Designed experiments: new approaches to studying urban ecosystems. *Frontiers in Ecology and the Environment*, 3(10):549–556, 2005.

[106] Simon Washington, Matthew G Karlaftis, Fred Mannering, and Panagiotis Anastasopoulos. *Statistical and econometric methods for transportation data analysis*. CRC press, 2003.

[107] Vanesa Castán Broto and Harriet Bulkeley. A survey of urban climate change experiments in 100 cities. *Global environmental change*, 23(1):92–102, 2013.

[108] Andrew Karvonen and Bas Van Heur. Urban laboratories: Experiments in reworking cities. *International Journal of Urban and Regional Research*, 38(2):379–392, 2014.

[109] Federico Caprotti and Robert Cowley. Interrogating urban experiments. *Urban Geography*, 38(9):1441–1450, 2017.

[110] Jeffery T Leek and Roger D Peng. What is the question? *Science*, 347(6228):1314–1315, 2015.

[111] William Cyrus Navidi. *Statistics for engineers and scientists*. McGraw-Hill Higher Education New York, NY, USA, 2008.

[112] John T Behrens. Principles and procedures of exploratory data analysis. *Psychological Methods*, 2(2):131, 1997.

[113] Richard Lowry. *Concepts and applications of inferential statistics*. Online, 2014.

[114] Larry B Christensen, Burke Johnson, Lisa Anne Turner, and Larry B Christensen. *Research methods, design, and analysis*. Pearson New York, 2011.

[115] Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*, pages 161–168. ACM, 2006.

[116] Claude Sammut and Geoffrey I Webb. *Encyclopedia of machine learning and data mining*. Springer Publishing Company, Incorporated, 2017.

[117] Carlos Caminha, Vasco Furtado, Vládia Pinheiro, and Caio Silva. Micro-interventions in urban transportation from pattern discovery on the flow of passengers and on the bus network. In *2016 IEEE International Smart Cities Conference (ISC2)*, pages 1–6. IEEE, 2016.

[118] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*, 2017.

[119] Chris Anderson. The end of theory: The data deluge makes the scientific method obsolete. *Wired magazine*, 16(7):16–07, 2008.

[120] Paul Smaldino. Better methods can't make up for mediocre theory. *Nature*, 575(7781):9–9, Nov 2019.

[121] Fulvio Mazzocchi. Could big data be the end of theory in science? a few remarks on the epistemology of data-driven science. *EMBO reports*, 16(10):1250–1255, 2015.

[122] Sauro Succi and Peter V Coveney. Big data: the end of the scientific method? *Philosophical Transactions of the Royal Society A*, 377(2142):20180145, 2019.

[123] Anonymous. *Systems Engineering Fundamentals*. Number January. Defense Acquisition University Press, Fort Belvoir, Virginia, USA, 2001.

[124] J Rumbaugh, I Jacobson, and G Booch. *The Unified Modeling Language Reference Manual*. Addison-Wesley, Reading, Mass., 2005.

[125] Dov Dori. *Object-process methodology : a holistics systems paradigm*. Springer, Berlin ; New York, 2002.

[126] OMG. Notation (bpmn) version 2.0. Technical report, Object Management Group, 2011.

[127] Jay W Forrester. System dynamics, systems thinking, and soft or. *System dynamics review*, 10(2-3):245–256, 1994.

[128] Steven D Eppinger and Tyson R Browning. *Design structure matrix methods and applications*. MIT Press, Cambridge, Mass., 2012.

[129] C G Cassandras and S Lafortune. *Introduction to Discrete Event Systems*. Springer, New York, NY, USA, 2nd edition, 2007.

[130] Eric Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the national academy of sciences*, 99(suppl 3):7280–7287, 2002.

[131] Katsuhiko Ogata. *Discrete-time control systems*. Prentice Hall, Englewood Cliffs, N.J., 2nd edition, 1994.

[132] Frank A Sonnenberg and J Robert Beck. Markov models in medical decision making: a practical guide. *Medical decision making*, 13(4):322–338, 1993.

[133] Arjan J Van Der Schaft and Johannes Maria Schumacher. *An introduction to hybrid dynamical systems*, volume 251. Springer London, 2000.

[134] Deema Fathi Allan and Amro M. Farid. A benchmark analysis of open source transportation-electrification simulation tools. In *2015 IEEE Conference on Intelligent Transportation Systems*, pages 1–7, Las Palmas de Gran Canaria Canary Islands, Spain, 2015.

[135] Gregorio D'Agostino and Antonio Scala. *Networks of networks: the last frontier of complexity*, volume 340. Springer, Berlin, Heidelberg, 2014.

[136] S Havlin, DY Kenett, A Bashan, J Gao, and HE Stanley. Vulnerability of network of networks. *The European Physical Journal Special Topics*, 223(11):2087–2106, 2014.

[137] Manlio De Domenico, Albert Solé-Ribalta, Sergio Gómez, and Alex Arenas. Navigability of interconnected networks under random failures. *Proceedings of the National Academy of Sciences*, 111(23):8351–8356, 2014.

[138] Osman Yağan and Virgil Gligor. Analysis of complex contagions in random multiplex networks. *Phys. Rev. E*, 86:036103, Sep 2012.

[139] Vincenzo Nicosia, Ginestra Bianconi, Vito Latora, and Marc Barthelemy. Growing multiplex networks. *Physical review letters*, 111(5):058701, 2013.

[140] Ginestra Bianconi. Statistical mechanics of multiplex networks: Entropy and overlap. *Physical Review E*, 87(6):062806, 2013.

[141] Federico Battiston, Vincenzo Nicosia, and Vito Latora. Structural measures for multiplex networks. *Physical Review E*, 89(3):032804, 2014.

[142] Emoke-Agnes Horvát and Katharina A Zweig. One-mode projection of multiplex bipartite graphs. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, pages 599–606. IEEE Computer Society, 2012.

[143] Albert Sole-Ribalta, Manlio De Domenico, Nikos E Kouvaris, Albert Díaz-Guilera, Sergio Gómez, and Alex Arenas. Spectral properties of the laplacian of multiplex networks. *Physical Review E*, 88(3):032807, 2013.

[144] Emanuele Cozzo, Raquel A Banos, Sandro Meloni, and Yamir Moreno. Contact-based social contagion in multiplex networks. *Physical Review E*, 88(5):050801, 2013.

[145] Luis Solá, Miguel Romance, Regino Criado, Julio Flores, Alejandro García del Amo, and Stefano Boccaletti. Eigenvector centrality of nodes in multiplex networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 23(3):033131, 2013.

[146] Philippa Pattison and Stanley Wasserman. Logit models and logistic regressions for social networks: Ii. multivariate relations. *British Journal of Mathematical and Statistical Psychology*, 52(2):169–193, 1999.

[147] Matteo Barigozzi, Giorgio Fagiolo, and Giuseppe Mangioni. Identifying the community structure of the international-trade multi-network. *Physica A: statistical mechanics and its applications*, 390(11):2051–2066, 2011.

[148] Deng Cai, Zheng Shao, Xiaofei He, Xifeng Yan, and Jiawei Han. Community mining from multi-relational networks. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 445–452. Springer, 2005.

[149] Andreas Harrer and Alona Schmidt. An approach for the blockmodeling in multi-relational networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on*, pages 591–598. IEEE, 2012.

[150] Victor Stroele, Jonice Oliveira, Geraldo Zimbrao, and Jano M Souza. Mining and analyzing multirelational social networks. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, volume 4, pages 711–716. IEEE, 2009.

[151] Wei Li, Amir Bashan, Sergey V. Buldyrev, H. Eugene Stanley, and Shlomo Havlin. Cascading failures in interdependent lattice networks: The critical role of the length of dependency links. *Phys. Rev. Lett.*, 108:228702, May 2012.

[152] Michaek Kwok-Po Ng, Xutao Li, and Yunming Ye. Multirank: co-ranking for objects and relations in multi-relational data. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1217–1225. ACM, 2011.

[153] Piotr Bródka, Katarzyna Musial, and Przemysław Kazienko. A method for group extraction in complex social networks. *Knowledge Management, Information Systems, E-Learning, and Sustainability Research*, pages 238–247, 2010.

[154] Piotr Brodka, Pawel Stawiak, and Przemyslaw Kazienko. Shortest path discovery in the multi-layered social network. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 497–501. IEEE, 2011.

[155] Piotr Bródka, Przemysław Kazienko, Katarzyna Musiał, and Krzysztof Skibicki. Analysis of neighbourhoods in multi-layered dynamic social networks. *International Journal of Computational Intelligence Systems*, 5(3):582–596, 2012.

[156] Michele Berlingerio, Michele Coscia, Fosca Giannotti, Anna Monreale, and Dino Pedreschi. The pursuit of hubbiness: analysis of hubs in large multidimensional networks. *Journal of Computational Science*, 2(3):223–237, 2011.

[157] Michele Berlingerio, Fabio Pinelli, and Francesco Calabrese. Abacus: frequent pattern mining-based community discovery in multidimensional networks. *Data Mining and Knowledge Discovery*, 27(3):294–320, 2013.

[158] Michele Berlingerio, Michele Coscia, Fosca Giannotti, Anna Monreale, and Dino Pedreschi. Multidimensional networks: foundations of structural analysis. *World Wide Web*, 16(5-6):567–593, 2013.

[159] Lei Tang, Xufei Wang, and Huan Liu. Community detection via heterogeneous interaction analysis. *Data mining and knowledge discovery*, 25(1):1–33, 2012.

[160] Chris Barrett, Karthik Channakeshava, Fei Huang, Junwhan Kim, Achla Marathe, Madhav V. Marathe, Guanhong Pei, Sudip Saha, Balaaji S. P. Subbiah, and Anil Kumar S. Vullikanti. Human initiated cascading failures in societal infrastructures. *PLoS ONE*, 7(10):1–20, 10 2012.

[161] Przemysław Kazienko, Katarzyna Musial, and Tomasz Kajdanowicz. Multidimensional social network in the social recommender system. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 41(4):746–759, 2011.

[162] Michele Coscia, Giulio Rossetti, Diego Pennacchioli, Damiano Ceccarelli, and Fosca Giannotti. "you know because i know": A multidimensional network approach to human resources problem. In *Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on*, pages 434–441. IEEE, 2013.

[163] Przemysław Kazienko, Katarzyna Musial, Elżbieta Kukla, Tomasz Kajdanowicz, and Piotr Bródka. Multidimensional social network: model and analysis. *Computational Collective Intelligence. Technologies and Applications*, pages 378–387, 2011.

[164] Peter J Mucha, Thomas Richardson, Kevin Macon, Mason A Porter, and Jukka-Pekka Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *science*, 328(5980):876–878, 2010.

[165] Vincenza Carchiolo, Alessandro Longheu, Michele Malgeri, and Giuseppe Mangioni. Communities unfolding in multislice networks. In *Complex Networks*, pages 187–195. Springer, 2011.

[166] Danielle S Bassett, Mason A Porter, Nicholas F Wymbs, Scott T Grafton, Jean M Carlson, and Peter J Mucha. Robust detection of dynamic community structure in networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 23(1):013142, 2013.

[167] Daniel Irving and Francesco Sorrentino. Synchronization of dynamical hypernetworks: Dimensionality reduction through simultaneous block-diagonalization of matrices. *Physical Review E*, 86(5):056102, 2012.

[168] Francesco Sorrentino. Synchronization of hypernetworks of coupled dynamical systems. *New Journal of Physics*, 14(3):033035, 2012.

[169] Sebastian Funk and Vincent AA Jansen. Interacting epidemics on overlay networks. *Physical Review E*, 81(3):036118, 2010.

[170] Vincent Marceau, Pierre-André Noël, Laurent Hébert-Dufresne, Antoine Allard, and Louis J Dubé. Modeling the dynamical interaction between epidemics on overlay networks. *Physical Review E*, 84(2):026105, 2011.

[171] Xuetao Wei, Nicholas Valler, B Aditya Prakash, Iulian Neamtiu, Michalis Faloutsos, and Christos Faloutsos. Competing memes propagation on networks: a case study of composite networks. *ACM SIGCOMM Computer Communication Review*, 42(5):5–12, 2012.

[172] Matthew Rocklin and Ali Pinar. On clustering on graphs with multiple edge types. *Internet Mathematics*, 9(1):82–112, 2013.

[173] Jason Hindes, Sarabjeet Singh, Christopher R Myers, and David J Schneider. Epidemic fronts in complex networks with metapopulation structure. *Physical Review E*, 88(1):012809, 2013.

[174] GJ Baxter, SN Dorogovtsev, AV Goltsev, and JFF Mendes. Avalanche collapse of interdependent networks. *Physical review letters*, 109(24):248701, 2012.

[175] Matteo Barigozzi, Giorgio Fagiolo, and Diego Garlaschelli. Multinetwork of international trade: A commodity-specific analysis. *Phys. Rev. E*, 81:046104, Apr 2010.

[176] Davide Cellai, Eduardo López, Jie Zhou, James P. Gleeson, and Ginestra Bianconi. Percolation in multiplex networks with overlap. *Phys. Rev. E*, 88:052811, Nov 2013.

[177] Charles D. Brummitt, Kyu-Min Lee, and K.-I. Goh. Multiplexity-facilitated cascades in networks. *Phys. Rev. E*, 85:045102, Apr 2012.

[178] Peter J. Mucha, Thomas Richardson, Kevin Macon, Mason A. Porter, and Jukka-Pekka Onnela. Community structure in time-dependent, multiscale, and multiplex networks. *Science*, 328(5980):876–878, 2010.

[179] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[180] Byungjoon Min and KI Goh. Layer-crossing overhead and information spreading in multiplex social networks. *seed*, 21(T22):T12, 2013.

[181] Kyu-Min Lee, Jung Yeol Kim, Won-kuk Cho, Kwang-Il Goh, and IM Kim. Correlated multiplexity and connectivity of multiplex random networks. *New Journal of Physics*, 14(3):033027, 2012.

[182] Byungjoon Min, Su Do Yi, Kyu-Min Lee, and K-I Goh. Network robustness of multiplex networks with interlayer degree correlations. *Physical Review E*, 89(4):042811, 2014.

[183] Emanuele Cozzo, Raquel A Banos, Sandro Meloni, and Yamir Moreno. Contact-based social contagion in multiplex networks. *Physical Review E*, 88(5):050801, 2013.

[184] Antoine Allard, Pierre-André Noël, Louis J. Dubé, and Babak Pourbohloul. Heterogeneous bond percolation on multitype networks with an application to epidemic dynamics. *Phys. Rev. E*, 79:036113, Mar 2009.

[185] Amir Bashan, Yehiel Berezin, Sergey V. Buldyrev, and Shlomo Havlin. The extreme vulnerability of interdependent spatially embedded networks. *Nat Phys*, 9(10):667–672, 10 2013.

[186] Alessio Cardillo, Massimiliano Zanin, Jesús Gómez-Gardenes, Miguel Romance, Alejandro J García del Amo, and Stefano Boccaletti. Modeling the multi-layer nature of the european air transport network: Resilience and passengers re-scheduling under random failures. *arXiv preprint arXiv:1211.6839*, 2012.

[187] Mark Dickison, Shlomo Havlin, and H Eugene Stanley. Epidemics on interconnected networks. *Physical Review E*, 85(6):066109, 2012.

[188] Jonathan F Donges, Hanna CH Schultz, Norbert Marwan, Yong Zou, and Jürgen Kurths. Investigating the topology of interacting networks. *The European Physical Journal B*, 84(4):635–651, 2011.

[189] Emmanuel Lazega, Marie-Thérèse Jourda, Lise Mounier, and Rafaël Stofer. Catching up with big fish in the big pond? multi-level network analysis through linked design. *Social Networks*, 30(2):159–176, 2008.

[190] E. A. Leicht and R. M. D'Souza. Percolation on interacting networks. *ArXiv e-prints*, July 2009.

[191] VHP Louzada, NAM Araújo, JS Andrade Jr, and HJ Herrmann. Breathing synchronization in interconnected networks. *arXiv preprint arXiv:1304.5177*, 2013.

[192] J Martin-Hernandez, H Wang, P Van Mieghem, and G D'Agostino. On synchronization of interdependent networks. *arXiv preprint arXiv:1304.4731*, 2013.

[193] Roni Parshani, Sergey V. Buldyrev, and Shlomo Havlin. Interdependent networks: Reducing the coupling strength leads to a change from a first to second order percolation transition. *Phys. Rev. Lett.*, 105:048701, Jul 2010.

[194] Faryad Darabi Sahneh, Caterina Scoglio, and Fahmida N Chowdhury. Effect of coupling on the epidemic threshold in interconnected complex networks: A spectral analysis. In *American Control Conference (ACC), 2013*, pages 2307–2312. IEEE, 2013.

[195] Anna Saumell-Mendiola, M. Ángeles Serrano, and Marián Boguñá. Epidemic spreading on interconnected networks. *Phys. Rev. E*, 86:026106, Aug 2012.

[196] Yizhou Sun, Yintao Yu, and Jiawei Han. Ranking-based clustering of heterogeneous information networks with star network schema. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 797–806. ACM, 2009.

[197] Alexei Vazquez. Spreading dynamics on heterogeneous populations: multitype network approach. *Physical Review E*, 74(6):066114, 2006.

[198] Caixia Wang, Zongxiang Lu, and Ying Qiao. A consideration of the wind power benefits in day-ahead scheduling of wind-coal intensive power systems. *IEEE Trans. Power Syst.*, 28(1):236–245, Feb 2013.

[199] Jin Zhou, Lan Xiang, and Zengrong Liu. Global synchronization in general complex delayed dynamical networks and its applications. *Physica A: Statistical Mechanics and its Applications*, 385(2):729–742, November 2007.

[200] Di Zhou, Jianxi Gao, H Eugene Stanley, and Shlomo Havlin. Percolation of partially interdependent scale-free networks. *Physical Review E*, 87(5):052812, 2013.

[201] Lei Gao, Jiahai Yang, Hui Zhang, Bin Zhang, and Donghong Qin. Flowinfra: A fault-resilient scalable infrastructure for network-wide flow level measurement. *2011 13th Asia-Pacific Network Operations and Management Symposium*, page KICS KNOM; IEICE ICM, Sep 2011.

[202] Kyu-Min Lee, Jung Yeol Kim, Won-kuk Cho, Kwang-Il Goh, and IM Kim. Correlated multiplexity and connectivity of multiplex random networks. *New Journal of Physics*, 14(3):033027, 2012.

[203] Emanuele Cozzo, Alex Arenas, and Yamir Moreno. Stability of boolean multilevel networks. *Phys. Rev. E*, 86:036115, Sep 2012.

[204] Regino Criado, Julio Flores, Alejandro García del Amo, Jesús Gómez-Gardeñes, and Miguel Romance. A mathematical model for networks with structures in the mesoscale. *International Journal of Computer Mathematics*, 89(3):291–309, 2012.

[205] Yinliang Xu and Wenxin Liu. Novel Multiagent Based Load Restoration Algorithm for Microgrids. *Smart Grid, IEEE Transactions on*, 2(1):152–161, 2011.

[206] Osman Yagan, Dajun Qian, Junshan Zhang, and Douglas Cochran. Conjoining speeds up information diffusion in overlaying social-physical networks. *IEEE Journal on Selected Areas in Communications*, 31(6):1038–1048, 2013.

[207] Kathleen M Carley and Vanessa Hill. Structural change and learning within organizations. *Dynamics of organizations: Computational modeling and organizational theories*, pages 63–92, 2001.

[208] Kathleen M Carley, Jana Diesner, Jeffrey Reminga, and Maksim Tsvetovat. Toward an interoperable dynamic network analysis toolkit. *Decision Support Systems*, 43(4):1324–1347, 2007.

[209] Darcy Davis, Ryan Lichtenwalter, and Nitesh V Chawla. Multi-relational link prediction in heterogeneous information networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 281–288. IEEE, 2011.

[210] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment*, 4(11):992–1003, 2011.

[211] Yizhou Sun. *Mining heterogeneous information networks*. PhD thesis, University of Illinois at Urbana-Champaign, 2012.

[212] Wei-Qing Sun, Cheng-Min Wang, Ping Song, and Yan Zhang. Flexible load shedding strategy considering real-time dynamic thermal line rating. *IET Generation, Transmission & Distribution*, 7(2):130–137, Feb 2013.

[213] Maksim Tsvetovat, Jeffrey Reminga, and Kathleen M Carley. Dynetml: Interchange format for rich social network data. *SSRN*, 2004.

[214] Giancarlo Guizzardi. *Ontological foundations for structural conceptual models*. CTIT, Centre for Telematics and Information Technology, 2005.

[215] Dov Dori. *Model-based systems engineering with OPM and SysML*. Springer, 2015.

[216] Alexander Kossiakoff, William N Sweet, and Knovel (Firm). *Systems engineering principles and practice*. Wiley-Interscience, Hoboken, N.J., 2003.

[217] N P Suh. *The Principles of Design*. Oxford University Press, 1990.

[218] Paulo Manuel de Oliveira de Jesus. *Remuneration of distributed generation: A holistic approach*. PhD thesis, Faculdade de Engharia Universidade de Porto, 20007.

[219] Ignacio J Ramirez-Rosado and Jose L Bernal-Agustin. Genetic algorithms applied to the design of large power distribution systems. *Power Systems, IEEE Transactions on*, 13(2):696–703, 1998.

[220] Robert Cloutier, Gerrit Muller, Dinesh Verma, Roshanak Nilchiani, Eirik Hole, and Mary Bone. The concept of reference architectures. *Systems Engineering*, 13(1):14–27, 2010.

[221] Ray Daniel Zimmerman, Carlos Edmundo Murillo-Sanchez, and Robert John Thomas. MATPOWER: Steady-State Operations, Planning, and Analysis Tools for Power Systems Research and Education. *IEEE Transactions on Power Systems*, 26(1):12–19, 2011.

[222] Ray Daniel Zimmerman and Carlos Edmundo Murillo-Sanchez. Matpower 4.1 User‘s Manual. Technical report, Power Systems Engineering Research Center, 2011.

[223] Henry M Paynter. *Analysis and design of engineering systems*. MIT press, 1961.

[224] Dean Karnopp, Donald L Margolis, and Ronald C Rosenberg. *System dynamics: a unified approach*. Wiley, New York, 2nd edition, 1990.

[225] Forbes T. Brown. *Engineering System Dynamics*. CRC Press Taylor & Francis Group, Boca Raton, FL, 2nd edition, 2007.

[226] Herman E Koenig, YeLMAZ TOKAD, and Hiremaglur K Kesavan. *ANALYSIS OF DISCRETE PHYSICAL SYSTEMS.* McGraw-Hill, 1967.

[227] William A Blackwell. *Mathematical modeling of physical networks*. Collier-Macmillan, 1968.

[228] J Lowen Shearer, Arthur T Murphy, and Herbert H Richardson. *Introduction to system dynamics*, volume 7017. Addison-Wesley Reading, 1967.

[229] Benjamin C Kuo. *Linear networks and systems*. McGraw-Hill, 1967.

[230] Shu-Park Chan, Shu-Yun Chan, Shu-Gar Chan, et al. *Analysis of linear Networks and Systems*. Mass. Addison-Wesley, 1972.

[231] J W Forrester. Industrial Dynamics: A Major Breakthrough for Decision Makers. *Harvard Business Review*, 36(4):37–66, 1958.

[232] John D Sterman. *Business dynamics: systems thinking and modeling for a complex world*, volume 19. Irwin/McGraw-Hill Boston, 2000.

[233] Maarten van Steen. *Graph Theory and Complex Networks: An Introduction*. Number January. Maarten van Steen, 2010.

[234] J B Pine. *Mass Customization: The New Frontier in Business Competition*. Harvard Business School Press, Cambridge, MA, 1993.

[235] J Pine. Mass Customizing Products and Services. *Planning Review*, July/Augus:6–13, 1993.

[236] Shana Smith, Roger Jiao, and Chih-Hsing Chu. Editorial: advances in mass customization. *Journal of Intelligent Manufacturing*, 24(5):873–876, Sep 2012.

[237] Shana Smith, Gregory C. Smith, Roger Jiao, and Chih-Hsing Chu. Mass customization in the product life cycle. *Journal of Intelligent Manufacturing*, 24(5):877–885, Oct 2012.

[238] Giovani Da Silveira, Denis Borenstein, H S Fogliatto, G Da Silveira, and F S Fogliatto. Mass customization: Literature review and research directions. *International Journal of Production Economics*, 72(1):1–13, 2001.

[239] David Hoyle. *ISO 9000 pocket guide*. Butterworth-Heinemann, Oxford ; Boston, 1998.

[240] Seth Warner. *Modern algebra*. Courier Corporation, 1990.

[241] Edward Crawley, Bruce Cameron, and Daniel Selva. *System Architecture: Strategy and Product Development for Complex Systems*. Prentice Hall Press, Upper Saddle River, N.J., 2015.

[242] Michael J Fischer and Albert R Meyer. Boolean matrix multiplication and transitive closure. In *Switching and Automata Theory, 1971., 12th Annual Symposium on*, pages 129–131. IEEE, 1971.

[243] Mark EJ Newman. The structure and function of complex networks. *SIAM review*, 45(2):167–256, 2003.

[244] Petter Holme and Jari Saramäki. Temporal networks. *Physics reports*, 519(3):97–125, 2012.

[245] A A Shabana. *Dynamics of Multibody Systems*. Cambridge University Press, 2 edition, 1998.

[246] Wolfgang Borutzky. *Bond Graph Modelling of Engineering Systems*. Springer, 2011.

[247] Mania Pavella, Damien Ernst, and Daniel Ruiz-Vega. *Transient Stability of Power Systems A Unified Approach to Assessment and Control*. Kluwer Academic Publishers, Boston, 2000.

[248] Seth Lloyd. Ultimate physical limits to computation. *Nature*, 406(6799):1047, 2000.

[249] Louchka Popova-Zeugmann. *Time Petri Nets*. Springer, Berlin, Heidelberg, 2013.

[250] René David and Hassane Alla. *Discrete, continuous, and hybrid Petri nets*. Springer, Berlin, Heidelberg, 2010.

[251] P.M. Subcommittee. Ieee reliability test system. *Power Apparatus and Systems, IEEE Transactions on*, PAS-98(6):2047–2054, Nov 1979.

[252] C Grigg, P Wong, P Albrecht, R Allan, M Bhavaraju, R Billinton, Q Chen, C Fong, S Haddad, S Kuruganty, W Li, R Mukerji, D Patton, N Rau, D Reppen, A Schneider, M Shahidehpour, and C Singh. The IEEE Reliability Test System-1996. A report prepared by the Reliability Test System Task Force of the Application of Probability Methods Subcommittee. *Power Systems, IEEE Transactions on*, 14(3):1010–1020, 1999.

[253] R.N. Allan, R. Billinton, and N.M.K. Abdel-Gawad. The ieee reliability test system - extensions to and evaluation of the generating system. *Power Engineering Review, IEEE*, PER-6(11):24–24, Nov 1986.

[254] Center for Water Systems. Anytown water distribution network, 2006.

[255] The White House Office of the Press Secretary. Presidential Policy Directive: Critical Infrastructure Security and Resilience (PPD-21). Technical report, The White House, Washington, D.C. United states, 2013.

[256] Yacov Y Haimes, Kenneth Crowther, and Barry M Horowitz. Homeland security preparedness: Balancing protection with resilience in emergent systems. *Systems Engineering*, 11(4):287–308, 2008.

[257] Department of Homeland Security. National Infrastructure Protection Plan: Partnering for Critical Infrastructure Security and Resilience. Technical report, Department of Homeland Security, Washington, D.C. United states, 2013.

[258] Committee on Increasing National Resilience to Hazards and Disasters and Committee on Science Engineering and Public Policy. *Disaster Resilience: A National Imperative*. The National Academies Press, Washington, DC, United States, 2012.

[259] M. Treiber and A. Kesting. An open-source microscopic traffic simulator. *Intelligent Transportation Systems Magazine, IEEE*, 2(3):6–13, Fall 2010.

[260] Wester C.H. Schoonenberg and Amro M. Farid. Modeling Smart Cities with Heterofunctional Graph Theory. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC2017), Intelligent Industrial System Special Session*, volume 1, pages 1–10, Banff, Canada, 2017.

[261] Deema Fathi Allan, Toufic Mezher, and Amro M. Farid. Enhanced Electric Vehicle Adoption Scenarios for Abu Dhabi Road Transportation. In *UAE Graduate Students Research Conference*, pages 1–2, Al Ain, UAE, 2016.

[262] Mike Schulze, Henrik Nehler, Mikael Ottosson, and Patrik Thollander. Energy management in industry–a systematic review of previous findings and an integrative conceptual framework. *Journal of Cleaner Production*, 112:3692–3708, 2016.

[263] T.A. Napp, A. Gambhir, T.P. Hills, N. Florin, and P.S Fennell. A review of the technologies, economics and policy instruments for decarbonising energy-intensive manufacturing industries. *Renewable and Sustainable Energy Reviews*, 30:616–640, Feb 2014.

[264] Aníbal T de Almeida, Fernando JTE Ferreira, and Dick Both. Technical and economical considerations in the application of variable-speed drives with electric motor systems. *IEEE Transactions on Industry Applications*, 41(1):188–199, 2005.

[265] Max Åhman and Lars J Nilsson. Decarbonizing industry in the eu: Climate, trade and industrial policy strategies. In *Decarbonization in the European Union*, pages 92–114. Springer, 2015.

[266] Max Åhman, Lars J Nilsson, and Bengt Johansson. Global climate policy and deep decarbonization of energy-intensive industries. *Climate Policy*, pages 1–16, 2016.

[267] Michael E Porter and Mark R Kramer. The link between competitive advantage and corporate social responsibility. *Harvard business review*, 84(12):78–92, 2006.

[268] Archie B Carroll. Corporate social responsibility: The centerpiece of competing and complementary frameworks. *Organizational Dynamics*, 44(2):87–96, 2015.

[269] Mohammad Hadi Amini, Behrouz Nabi, and Mahmoud-Reza Haghifam. Load management using multi-agent systems in smart distribution network. In *Power and Energy Society General Meeting (PES), 2013 IEEE*, pages 1–5. IEEE, 2013.

[270] Kianoosh G Boroojeni, M Hadi Amini, Arash Nejadpak, Tomislav Dragicevic, Sitharama S Iyengar, and Frede Blaabjerg. A novel cloud-based platform for implementation of oblivious power routing for clusters of microgrids. *IEEE Access*, 2016.

[271] Nikos Hatziargyriou. *Microgrids: Architectures and Control*. Wiley IEEE Press, West Sussex, England, 2014.

[272] Sebastian Thiede. *Energy efficiency in manufacturing systems*. Springer Science & Business Media, 2012.

[273] Kanako Tanaka. Review of policies and measures for energy efficiency in industry sector. *Energy Policy*, 39(10):6532–6550, 2011.

[274] Hyun Woo Jeon, Marco Taisch, and Vittaldas V Prabhu. Modelling and analysis of energy footprint of manufacturing systems. *International Journal of Production Research*, (ahead-of-print):1–11, 2014.

[275] Gokan May, Marco Taisch, and David Kelly. Enhanced energy management in manufacturing through systems integration. pages 7525 – 7530, Vienna, Austria, 2013. Decision support tools;High energy consumption;Information and Communication Technologies;Key performance indicators;Management concerns;Monitor and control;Monitoring and control;Performance indicators;.

[276] E Giacone and S Mancò. Energy efficiency measurement in industrial processes. *Energy*, 38(1):331–345, 2012.

[277] Vittaldas V Prabhu, Damien Trentesaux, and Marco Taisch. Energy-aware manufacturing operations. *International Journal of Production Research*, pages 1–11, 2015.

[278] Yevgenia Mikhaylidi, Hussein Naseraldin, and Liron Yedidsion. Operations scheduling under electricity time-varying prices. *International Journal of Production Research*, 53(23):7136–7157, 2015.

[279] Kianoosh G Boroojeni, M Hadi Amini, and SS Iyengar. Smart grids: Security and privacy issues, 2016.

[280] Farhad Kamyab, Mohammadhadi Amini, Siamak Sheykhha, Mehrdad Hasanpour, and Mohammad Majid Jalali. Demand response program in smart grid using supply function bidding mechanism. *IEEE Transactions on Smart Grid*, 7(3):1277–1284, 2016.

[281] P. Vrba, V. Marik, P. Siano, P. Leitao, G. Zhabelova, V. Vyatkin, and T. Strasser. A review of agent and service-oriented concepts applied to intelligent energy systems. *Industrial Informatics, IEEE Transactions on*, 10(3):1890–1903, Aug 2014.

[282] Nanfang Yang, D Paire, Fei Gao, and A Miraoui. Power management strategies for microgrid-A short review. In *Industry Applications Society Annual Meeting, 2013 IEEE*, pages 1–9, 2013.

[283] Chun-Xia Dou and Bin Liu. Multi-agent based hierarchical hybrid control for smart microgrid. *IEEE Transactions on Smart Grid*, 4(2):771–778, Jun 2013.

[284] Frank Ibarra Hernandez, Carlos Alberto Canesin, Ramon Zamora, Fransiska Martina, and Anurag K Srivastava. Energy management and control for islanded microgrid using multi-agents. *2013 North American Power Symposium (NAPS)*, pages 1–6, Sep 2013.

[285] Mohammadreza Mazidi, Alireza Zakariazadeh, Shahram Jadid, and Pierluigi Siano. Integrated scheduling of renewable generation and demand response programs in a microgrid. *Energy Conversion and Management*, 86(0):1118 – 1127, 2014.

[286] Zeng Jun, Liu Junfeng, Wu Jie, and H.W. Ngan. A multi-agent solution to energy management in hybrid renewable energy generation system. *Renewable Energy*, 36(5):1352–1363, May 2011.

[287] Lanre Olatomiwa, Saad Mekhilef, MS Ismail, and M Moghavvemi. Energy management strategies in hybrid renewable energy systems: A review. *Renewable and Sustainable Energy Reviews*, 62:821–835, 2016.

[288] Pierluigi Siano. Demand response and smart grids—a survey. *Renewable and Sustainable Energy Reviews*, 30:461–478, Feb 2014.

[289] P Palensky and D Dietrich. Demand Side Management: Demand Response, Intelligent Energy Systems, and Smart Loads. *Industrial Informatics, IEEE Transactions on*, 7(3):381–388, 2011.

[290] Vittaldas V Prabhu, Hyun Woo Jeon, and Marco Taisch. Simulation modelling of energy dynamics in discrete manufacturing systems. In *Service Orientation in Holonic and Multi Agent Manufacturing and Robotics*, pages 293–311. Springer, 2013.

[291] Birgit Vogel-Heuser, Benedikt Weißenberger, Heiko Meyer, and Josef Plössnig. Modeling of power consumption in manufacturing: Gross and detailed planning in consideration of all forms of energy as planning resources including load management during runtime. In *Industrial Technology (ICIT), 2014 IEEE International Conference on*, pages 659–663. IEEE, 2014.

[292] Luoke Hu, Chen Peng, Steve Evans, Tao Peng, Ying Liu, Renzhong Tang, and Ashutosh Tiwari. Minimising the machining energy consumption of a machine tool by sequencing the features of a part. *Energy*, 2017.

[293] C Pach, T Berger, Y Sallez, and D Trentesaux. Reactive control of overall power consumption in flexible manufacturing systems scheduling: A potential fields model. *Control Engineering Practice*, 44:193–208, 2015.

[294] F Tonelli, AAG Bruzzone, M Paolucci, E Carpanzano, G Nicolò, A Giret, MA Salido, and D Trentesaux. Assessment of mathematical programming and agent-based modelling for off-line scheduling: Application to energy aware manufacturing. *CIRP Annals-Manufacturing Technology*, 2016.

[295] Adriana Giret, Damien Trentesaux, and Vittal Prabhu. Sustainability in manufacturing operations scheduling: a state of the art review. *Journal of Manufacturing Systems*, 37:126–140, 2015.

[296] William Naggaga Lubega and Amro M Farid. A Reference System Architecture for the Energy-Water Nexus. *IEEE Systems Journal*, PP(99):1–11, 2014.

[297] Wolfgang Reisig. Understanding petri nets, 2013.

[298] Vito Introna, Vittorio Cesarotti, Miriam Benedetti, Sonia Biagiotti, and Raffaele Rotunno. Energy Management Maturity Model: an organizational tool to foster the continuous reduction of energy consumption in companies. *Journal of Cleaner Production*, 83(0):108–117, November 2014.

[299] MengChu Zhou and Naiqi Wu. *System modeling and control with resource-oriented Petri nets*. CRC Press, 2010.

[300] MengChu Zhou, F Di-C, et al. *Petri net synthesis for discrete event control of manufacturing systems*. Kluwer Academic Publishers, 1993.

[301] Kelwyn A D'Souza and Suresh K Khator. A survey of Petri net applications in modeling controls for automated manufacturing systems. *Computers in Industry*, 24(1):5–16, May 1994.

[302] James L Kirtley. *Electric power principles: sources, conversion, distribution and use*. John Wiley & Sons, 2011.

[303] Federico Milano. *Power system modelling and scripting*. Springer, New York, 1st edition, 2010.

[304] Solar power data for integration studies, April 2014.

[305] U.S.Energy-Information-Administration. Average tested heat rates by prime mover and energy source, 2015.

[306] LD Danny Harvey. Energy efficiency and the demand for energy services, 2010.

[307] O. L. de Weck, D. Roos, C. L. Magee, and C. M. Vest. *Life-Cycle Properties of Engineering Systems: The Ilities*. MITP, 2011.

[308] M Hadi Amini. A panorama of interdependent power systems and electrified transportation networks. In *Sustainable interdependent networks II*, pages 23–41. Springer, 2019.

[309] Berker Bilgin, Pierre Magne, Pawel Malysz, Yinye Yang, Vera Pantelic, Matthias Preindl, Alexandre Korobkine, Weisheng Jiang, Mark Lawford, and Ali Emadi. Making the case for electrified transportation. *IEEE Transactions on Transportation Electrification*, 1(1):4–17, 2015.

[310] Ashlynn S Stillwell, Carey W King, Michael E Webber, Ian J Duncan, and Amy Hardberger. The energy-water nexus in texas. *Ecology And Society*, 16(1):2, 2011.

[311] Karen Hussey and Jamie Pittock. The energy–water nexus: managing the links between energy and water for a sustainable future. *Ecology and Society*, 17(1), 2012.

[312] Morgan Bazilian, Holger Rogner, Mark Howells, Sebastian Hermann, Douglas Arent, Dolf Gielen, Pasquale Steduto, Alexander Mueller, Paul Komor, Richard SJ Tol, et al. Considering the energy, water and food nexus: Towards an integrated modelling approach. *Energy policy*, 39(12):7896–7906, 2011.

[313] Dakota J Thompson and Amro M Farid. A reference architecture for the american multi-modal energy system. *arXiv preprint arXiv:2012.14486*, 2020.

[314] Manlio De Domenico, Albert Solé-Ribalta, Emanuele Cozzo, Mikko Kivelä, Yamir Moreno, Mason A Porter, Sergio Gómez, and Alex Arenas. Mathematical formulation of multilayer networks. *Physical Review X*, 3(4):041022, 2013.

[315] Takuto Ishimatsu, Olivier L de Weck, Jeffrey A Hoffman, Yoshiaki Ohkami, and Robert Shishko. Generalized multicommodity network flow model for the earth–moon–mars logistics system. *Journal of Spacecraft and Rockets*, 53(1):25–38, 2016.

[316] Takuto Ishimatsu, Abdelkrim Doufene, Abdullah Alawad, and Olivier de Weck. Desalination network model driven decision support system: a case study of saudi arabia. *Desalination*, 423:65–78, 2017.

[317] Takuto Ishimatsu, Abdulaziz Alhassan, Abdelkrim Doufene, Olivier de Weck, Adnan Alsaati, Kenneth Strzepek, and Anas Alfaris. Large scale infrastructure design using evolving networks. 2020.

[318] K Jensen. Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use. In *EATCS Monographs on Theoretical Computer Science*, volume 1. Springer Verlag, 1992.

[319] Claude Girault and Rüdiger Valk. *Petri nets for systems engineering: a guide to modeling, verification, and applications*. Springer Science & Business Media, 2013.

[320] Pieter Schavemaker, Lou Van der Sluis, and Books24x7 Inc. *Electrical power system essentials*. Wiley, Chichester, England ; Hoboken, NJ, 2008.

[321] Antonio Gomez-Exposito, Antonio J Conejo, and Claudio Canizares. *Electric Energy Systems: Analysis and Operation*. CRC Press, Boca Raton, FL, 2008.

[322] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[323] Tamara Gibson Kolda. Multilinear operators for higher-order decompositions. Technical report, Sandia National Laboratories, 2006.

[324] Anthony Wang, Kees van der Leun, Daan Peters, and Maud Buseman. European hydrogen backbone: How a dedicated hydrogen infrastructure can be created. Technical report, Enagás, Energinet, Fluxys Belgium, Gasunie, GRTgaz, NET4GAS, OGE, ONTRAS, Snam, Swedegas, Teréga, 2020.

[325] Keith Scott. *Electrochemical Methods for Hydrogen Production*, chapter 1: Introduction to Electrolysis, Electrolysers and Hydrogen Production. Royal Society of Chemistry, 2019.

[326] Luca Bertuccioli, Alvin Chan, David Hart, Franz Lehner, Ben Madden, and Eleanor Standen. Development of water electrolysis in the european union. *Fuel cells and hydrogen joint undertaking*, 83, 2014.

[327] MA Rosen. Thermodynamic investigation of hydrogen production by steam-methane reforming. *International Journal of Hydrogen Energy*, 16(3):207–217, 1991.

[328] XD Peng. Analysis of the thermal efficiency limit of the steam methane reforming process. *Industrial & engineering chemistry research*, 51(50):16385–16392, 2012.

[329] National Academy of Engineering. *The hydrogen economy: opportunities, costs, barriers, and R&D needs*. National Academies Press, 2004.

[330] Energy Information Administration. Average tested heat rates by prime mover and energy source, 2009 - 2019, 2020.

[331] Bo Jiang, Amro M. Farid, and Kamal Youcef-Toumi. Demand side management in a day-ahead wholesale market a comparison of industrial and social welfare approaches. *Applied Energy*, 156(1):642–654, 2015.

[332] Bo Jiang, Aramazd Muzhikyan, Amro M Farid, and Kamal Youcef-Toumi. Demand Side Management in Power Grid Enterprise Control – A Comparison of Industrial and Social Welfare Approaches. *Applied Energy*, 187(1):833–846, 2017.

[333] Inas S. Khayal and Amro M. Farid. A Dynamic System Model for Personalized Healthcare Delivery and Managed Individual Health Outcomes. *submitted to: Health Systems*, 1(1):1–18, 2020.

[334] Dakota Thompson, Prabhat Hegde, Inas Schoonenberg, Wester C.H.and Khayal, and Amro M. Farid. The Hetero-functional Graph Theory Toolbox. *submitted to: Journal of Open Research Software*, 1(1):11, 2020.