A FRAMEWORK FOR SENSE-MAKING OF COMPLEX SOCIOTECHNICAL SYSTEMS

By

Lucie Reymondet

Ingénieur diplômé de l'Ecole Polytechnique, Palaiseau, France, 2016

Submitted to the Department of Aeronautics and Astronautics
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Aeronautics and Astronautics

at the
Massachusetts Institute of Technology
June 2016

Signature of Author……
Department of Aeronautics and Astronautics
May 19, 2016

Certified by..……..……
Donna H. Rhodes
Principal Research Scientist, Sociotechnical Systems Research Center
Thesis Supervisor

Certified by..……..……
Daniel E. Hastings
Cecil and Ida Green Education Professor of Engineering Systems and Aeronautics and Astronautics
Academic Advisor

Accepted by...……
Paulo Lozano
Associate Professor of Aeronautics and Astronautics
Graduate Program Chair

A FRAMEWORK FOR SENSE-MAKING OF COMPLEX SOCIOTECHNICAL SYSTEMS

By

Lucie Reymondet

Submitted to the Department of Aeronautics and Astronautics on May 19, 2016 in Partial Fulfillment of the Requirements for the Degree of Master of Science in Aeronautics and Astronautics

Abstract

Contemporary engineering systems are often highly complex, large-scale, open sociotechnical systems, with strongly interwoven technical systems and social players within a dynamic environmental context that influences both social and technical spheres. Examples of such systems include online social networks, air and maritime traffic control systems, urban transportation systems, net-centric warfare systems and humanitarian disaster relief supply chains. Growing complexity in sociotechnical systems is often blamed for unanticipated (emergent) behaviors in the system or its development, such as accidents, cascading failures, operational bottlenecks, development cost overruns and delays.

A literature review highlighted the types of complexity usually tackled by systems architecting researchers and practitioners. The scope of this thesis is sociotechnical systems, and it is apparent from literature and from the examination of three example systems that types of complexity are diverse among complex sociotechnical systems. A taxonomy is proposed that compiles major threads of this review. Essentially, complexity and emergence increase the difficulty of sense-making, which can be briefly defined as the ability to foresee system-level impacts of architectural decisions. Complex sociotechnical systems present challenges for systems architects: they are difficult to understand, study, predict, control or change, and often display emergent behavior, which may or may not be desirable. This thesis goes a step beyond classifying complexity to propose a framework for guiding sense-making of complexity in sociotechnical systems architecting.

The systems community has developed and instantiated many modeling approaches, practices, formal languages and toolsets, to help system architects and engineers with sense-making and decision-making. This work proposes a phenomena-based framework to leverage established modeling practices and guide the selection and composition of model-centric approaches for complex sociotechnical systems architecting. The framework is applied to the Collaborative Decision Making System at Paris Charles De Gaulle airport.

Thesis Supervisor: Donna Rhodes
Title: Principal Research Scientist, Sociotechnical Systems Research Center

A mes parents et mon frère, qui ont nourri ma détermination jusqu’ici de leur amour.

Acknowledgments

I would like to extend my thanks to so many people, who over the past two years at MIT and the past year at SEAri, have contributed to my personal and professional development. I have learnt that nothing matters like the people you work with…

First, I would like to thank **Dr. Donna Rhodes**, for betting on me a year ago and giving me the opportunity to jump onto SEAri's research effort when systems engineering was a new field to me. I would like to thank my academic advisor **Prof. Daniel Hastings**, for sharing stories that never fail to plant a seed of thought in your mind. My thanks also go to **Dr. Adam Ross**, who strived to make the SEAri's research legacy available to present students.

SEAri students have been an extraordinary group to work with. I would like to thank **Parker Vascik,** with whom I shared a memorable year as GA^3 chair, for introducing me to SEARi in the first place: I shall cherish our friendship! I'd like to thank **Paul La Tour**, for enlightening chats and debates about research, video games, and politics. I was happy to share the perks of thesis writing and presenting at a conference with **Sarah Rovito**. Additional thanks go to **Matthew Fitzgerald**, **Amaya Arcelus** (adoptive SEAri), **Shane German**, **Mike Curry** and **Jack Reid**. What an amazing group of people to work with!

I would like to thank **Didier Lucas**, **Jaufre Planchons**, **Kamal Amri**, and the control tower team from Paris Charles de Gaulle airport, for the time and wealth of information they contributed to the test case in the thesis. The control tower shift immersion was a memorable experience!

I extend my heartfelt thanks to **Beth Marois**, for her invaluable advice and support, and a record entry in my email inbox over the past two years!

I am grateful to my **family** for their patience, support and unconditional love: I promise to call home more often now! **Friends**, old and new, you have made my graduate experience unforgettable: MERCI!

The author gratefully acknowledges the financial support for this research from the Systems Engineering Advancement Research Initiative (SEAri), a part of the MIT Sociotechnical Systems Research Center (SSRC).

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Systems Architecting

Architecture: 1.The art or science of building or constructing edifices of any kind for human use; 2.The action or process of building; 3.Architectural work, structure, building; 4. (Computing) The conceptual structure and overall logical organization of a computer or computer-based system from the point of view of its use or design; a particular realization of this[1]

Broadly speaking, a system is a collection of things that together produce results unachievable by themselves alone. The value added by systems is in the interrelationships of their elements. This definition is valid across system disciplines, like biology, informatics, engineering, social sciences etc. Architecture is concerned with the selection of elements, their interactions, and the constraints on those elements and interactions necessary to satisfy the requirements and serve as a basis for the design (Whitcomb, Auguston, & Giammarco, 2015). Systems architecting differs from systems engineering, both in terms of the problems they tackle and in terms of the tools used to answer them. On one hand, engineering usually employs analytical tools, derived from mathematics, physics or other hard sciences to answer measurable problems or questions (e.g. optimize airfoil shape). Architecting on the other hand usually employs qualitative tools and guidelines to answer non-measurable, soft problems (e.g. stakeholder satisfaction). Engineering is concerned with quantifiable costs, architecting with qualitative worth. Engineering aims for technical optimization, architecting for client satisfaction. "Engineering is more of a science, architecting more of an art" (Maier, 2009). For example, in building a car, a systems architect is concerned with posing a problem, determining a set of tradeoffs and impactful decisions to be made (e.g. esthetics, aerodynamic performance, safety, cost), whereas a systems engineer is concerned with analyzing the tradeoff problem posed to propose mathematically grounded solutions (e.g. cost analysis, aerodynamic drag calculations, optimal mirror shape, hull stress calculations).

[1] Oxford English Dictionary (http://www.oed.com) accessed 03/09/16

Systems architecting is informed by systems engineering. For example, some architectural decisions impact system cost or practical feasibility. In the field of computing, Brooks (1962) states that architecture is "the art of determining the needs of the user of a structure, and then designing to meet those needs as effectively as possible within economic and technological constraints. Architecture must include engineering considerations, so that the design will be economical and feasible; but the emphasis in architecture is upon the needs of the user, whereas in engineering the emphasis is upon the needs of the fabricator" (Brooks, 1962). A system's structure, or architecture, also affects system complexity and life cycle properties. Structure refers to the way in which the elements of the system are interconnected, whereas architecture (see below) is a broader concept that also includes system functionalities and behavior in relation to user requirements (De Weck, Roos, & Magee, 2011).

Systems **architecture**

- System **structure**, i.e. the way in which the elements of the system are interconnected (static)
- System **functionalities & behavior**, in relation to user requirements (dynamic)

Systems architecting can be normative (describing architecture, as it "should" be), or method-based (following principles to arrive to a solution for well-known types of systems). Both these methods are analytic, experiment based, well understood and widely spread throughout academia and industry. However, for architecting systems with no precedent, data or best practices, where experiments are impossible or too costly, where there are many stakeholders, unknowns and alternative architectures, or little time for detailed analysis, or where there are soft and hard problems, such methodologies are limited. In these cases, stakeholder participative and heuristic methodologies are useful to assess architectural worthiness.

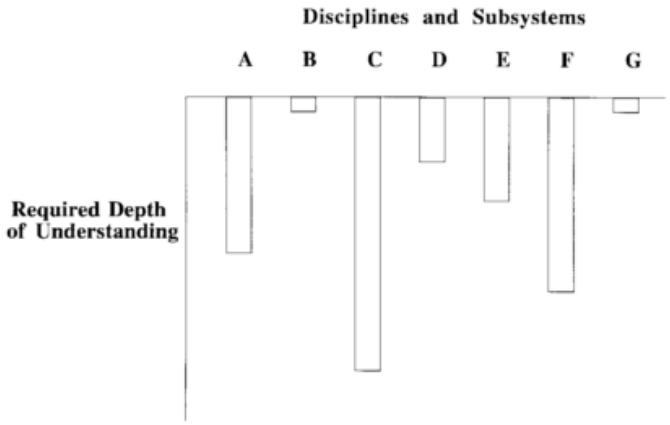


Figure 1-1: Architecting Calls for Different Depths of Understanding of Disciplines/Subsystems, from (Maier, 2009)**, p. 30**

Systems architecture should proceed from client, user, operator, or customer purpose. Architecture is not just about the structure of hardware, software and other engineered components: It encompasses the conceptual structure and functional behavior of a system. Although a systems architect's task usually does not call for detailed analysis (e.g. data flow, controls, logical design, algorithm coding, physical implementation of a mechanical structure etc.), some systems and user requirements may require that systems architects delve to some depth of understanding and analysis into selected disciplines/subsystems, into the consequences of architectural decisions made in these disciplines on the whole system, into the behavior of these subsystems in relation with the whole system. Consider for example a simple autonomous system for operating a gate: disciplines may involve mechanical engineering, software engineering and optical engineering; subsystems may be the computer and timer, the optical sensors, and the mechanical structure and actuators. User concern about system lifetime might emphasize mechanical structure considerations, whereas security concerns might emphasize computer and software behavior instead.

1.2 Sociotechnical Systems Architecting

Social: Of or relating to people or society in general

Technical: Relating to the practical use of machines or science in industry, medicine, etc.[2]

Sociotechnical systems are technical systems involving the participation of groups of people in ways that significantly affect the architecture and design of those systems. Trist (1981) reviews the evolution of the concept from its origination in the coal industry in the fifties. Initial sociotechnical theories understood that work organizations that best fit people with equipment yielded higher economic performance and job satisfaction. Systems theory formalized this understanding in terms of two systems of different nature, one technical and the other social: "the former follows the laws of the natural sciences while the latter follows the laws of human sciences and is a purposeful system." The two are coupled: the technical system requires a social system for its operation (e.g. work tasks, processes). The social system is dependent on the technical system for delivering functionality and creating value to clients (e.g. services, products). The healthcare system, online social networks, manufacturing systems, air transportation systems, urban transportation systems, and power grids are some examples of sociotechnical systems. The purposes of the technical and social systems are intertwined: a healthcare data management system might be developed in order to keep track of patients, an online social networks might include a messaging application in order for users to be able to communicate, a manufacturing production line is designed so as to be operated productively by employees etc. This coupling requires joint architecting of the social and technical systems. Making decisions for either the technical or the social system alone will result in a sub optimization whole: contradictions emerge from sociotechnical integration, synergies are untapped, resources are inefficiently allocated etc. For example, architecting airport infrastructure and processes requires accounting for the diversity of aircraft operators in the platform. Architecting for a "single-airline" airport would be a simplified version of the real situation, where airlines compete for resources and efficiency, form alliances, protect their data, negotiate slots with one another etc. This entanglement of social and technical systems is a factor of complexity in architecting sociotechnical systems.

[2] (http://www.merriam-webster.com/dictionary, accessed 02/28/16)

The *people* involved can be of different nature. Stakeholders can be operators (function, maintenance), beneficiaries or users of system functionality, regulators, purchasers, investors or financial beneficiaries, developers, or competitors. They can be directly involved (e.g. the general public directly uses an urban mobility system by commuting), or indirectly (e.g. market forces promote rail transit modes because of the increase of fuel price, environmental regulation pushes against private vehicle use). In many cases, there are conflicting interests amongst stakeholders, which makes architectural decision making more complex: more variables and alternatives have to be taken into account, and tradeoffs are non-obvious. Architecting a sociotechnical system is a problem that always involves finding an economically viable solution (Brooks, 1962) that accounts for weighted preferences of multiple stakeholders. The difficulty in accounting for people's preferences is that perceived goodness matters as much as factual goodness (Figure 1-2). This can pull the architecture in counterintuitive directions, or it can make the system resistant to top-down policy-making, for example. A heuristic derived therefrom is that success is in the eyes of the stakeholder, not the architect. For example, legacy systems are difficult to change, not only technically speaking (e.g. compatibility of old and new hardware and software versions), but also socially (e.g. user habits, acceptance of change). "Perhaps more than other complex systems, the design and development of social ones should be amenable to insights and heuristics. Social factors, after all, are notoriously difficult to measure, much less predict" (Maier, 2009).

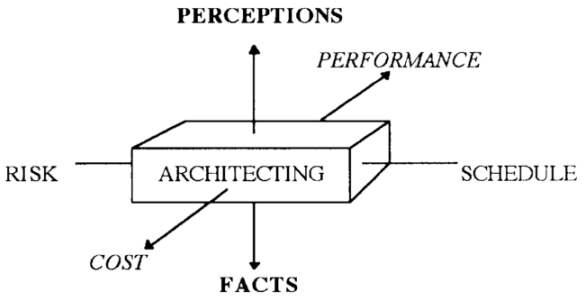


Figure 1-2: Sociotechnical Systems Architecting Adds "Perceptions vs. Facts" to Traditional "Technical" Architecting Tradeoffs, from (Maier, 2009)**, p.103**

Sociotechnical systems of particular interest for this thesis are systems associated with long lifecycles, large technical and/or social scales, partially designed partially evolved architectures, multiple stakeholders, tension between perceived and factual architectural "goodness", and with emergent behavior (to be defined). When system size and lifetime increase, along with the number of participants in system development and operation, and development time, then organization, project and process architecting are as important as engineering design (De Weck et al., 2011).

1.3 Modeling Approaches for Systems Architecting

"Modeling is the creation of abstractions or representations of the system to predict and analyze performance, costs, schedules, and risks, and to provide guidelines for systems research, development, design, manufacture, and management. Modeling is the centerpiece of systems architecting – a mechanism of communication to clients and builders, of design management with engineers and designers, of maintaining system integrity with project management, and of learning for the architect, personally" (Maier, 2009). Modeling is not limited to specifying a concept early in systems development cycles, like a traditional paper-and-wood models used in civil engineering. Architecture models are useful throughout system development and life cycle, for designing, changing, and retiring systems. Systems architects can use many forms of modeling: conceptual or mathematical, static or simulation-enabled, prototypes for testing and communication, flight simulators, optimization engines etc. "Modeling is a multipurpose, progressive activity, evolving and becoming less abstract and more concrete as the system is built and used" (Maier, 2009). These designs are not usually ready to use to build something.

Systems architects deliver a set of system models that together represent system architecture. These can be physical models, gross cost models, toy simulations or animations, textual documentation of assumptions. They would usually not be design-fit, i.e. fit for development and engineering. For example civil architects use multiple modeling methods for different purposes (Table 1-1). Architecture models are a proxy for system functions and structure, understandable across disciplines and among stakeholders. Architecture descriptions belong to a high level of abstraction and ignore many of the implementation details. They ought to be supportive for the refinement process of designing/engineering the system, and checked carefully at each of those

refinement steps. The architect develops a number of different views of the architecture reflecting various uses and users (Whitcomb et al., 2015).

Table 1-1: Models and Purposes in Civil Architecture, from (Maier, 2009)

Model	Purpose
Physical Scale Model	Convey look and site placement of building to architect, client, and builder
Floor Plans	Work with client to ensure building can perform basic functions desired
External Renderings,	Convey look of building to architect, client, and builder
Budget, Schedules	Ensure building meets client's financial performance objectives, manage builder relationship
Construction Blueprints	Communicate design requirements to builder, provide construction acceptance criteria

It is apparent that a single modeling approach will not suffice to represent the architecture of complex, large-scale, sociotechnical systems. A sociotechnical modeling strategy ought to be multi-perspective, multi-method and multi-scale (see chapter 4).

1.4 Research Questions and Approach

This thesis ties together two threads: the challenges of architecting sociotechnical systems and model-centric approaches used for systems architecting. Literature features a variety of tailored ad hoc modeling approaches for architecting complex systems in different fields. Some sociotechnical systems such as the healthcare system, receive a great deal of research attention. Other popular topics include network-centric warfare systems, urban mobility systems, air transportation systems, humanitarian logistics, online social networks, maritime surveillance systems etc. In such long-lived large-scale systems past architectural decisions have led to unanticipated effects in present system behavior. System complexity is often blamed for such emergent effects. "It is generally agreed that increasing complexity of systems is at the heart of the most difficult problems facing today's systems architecting and engineering" (Maier, 2009).

Sense-making of system architecture (structure and behavior) and how it creates emergence in complex sociotechnical systems is a challenge for systems architects. Bearing this in mind, this thesis asks the following questions:

- Can a model-centric framework guide sense-making of complexity and emergence in complex sociotechnical systems architecting?
- What insights does the application of such a framework yield on a case study?

Chapter 2 reviews what is meant by complexity in systems in literature, covering various systems fields. Drawing from the literature review, chapter 3 proposes a complexity taxonomy for *sociotechnical* systems that disentangles complexity into social system, technical system and ecosystem complexity factors. However, for the purpose of systems architecting, this thesis proposes using a phenomena-based architecture description of complex sociotechnical systems that reflects the complexity taxonomy. Illustration of the phenomena-based description on three case examples shows that sociotechnical systems of different nature emphasize different types of phenomena and different *flavors* of complexity. It is apparent that one needs models for sense-making of systems architecture, but what modeling approach is best suited to the task is less obvious. This research suggests that appropriateness of the model depends on the *flavor* of complexity displayed by the system, which is mirrored in the types of phenomena identified in systems architecture. Chapter 4 reviews model-centric systems architecting methods. Social complexity factors call for different modeling approaches than technical complexity factors. The proposed framework bridges the gap between the perceived complexity of a sociotechnical systems architecture, described in terms of phenomena, and the model-centric approaches, which may be suited for sense-making of the system's architecture. Chapter 5 applies the framework to an empirical test case on the Collaborative Decision Making System at Paris Charles de Gaulle airport. Chapter 6 summarizes the findings from the thesis, points out some limitations of the test case, suggests how further work could mitigate these limitations and extend the framework to systems of a different nature. Finally, it extends the considerations developed in chapters 4 and 5 on model-centric approaches for systems architecting to the topic of model curation, which has recently received much attention in the systems community.

2 Literature Review of Complexity and Emergence in Systems Architecting

2.1 Introduction to Complexity

Complexity takes on many meanings depending on the field of study. This section reviews understandings of the word "complexity" from a few fields, to give a glimpse into the multiplicity of interpretations and measurements of complexity.

In computational complexity theory, the complexity of a problem is the required amount of resources required for a *computer* to solve it. This one definition already allows for two different metrics: time complexity - the number of steps it takes to resolve the problem - or space complexity - the volume of memory used by the algorithm to resolve the problem.

In algorithmic information theory, the Kolmogorov complexity – also algorithmic complexity – of a mathematical object, such as a string of characters, is the length of the shortest program that outputs that object. Again the metric is non-unique, as it depends on the programming language. For example, the sequence {1, 1, 2, 3, 5, 8, 13, 21, 34, 75…} upper bounded by n_{max}, although seemingly complex, has low algorithmic complexity as it may be output by the following one-liner: "n_0=1; n_1=1; s={n_0, n_1}; while n_1<n_{max} temp=n_0+n_1; n_0=n_1; n_1=temp; s={s, n_1}; end"

In mathematics, the complexity of a graph has been measured in several different ways. Some example metrics developed are the number of its spanning trees, or a certain formula involving the number of vertices and edges, or the number of Boolean operations necessary to construct the graph from a fixed set of generating graphs, or the linear complexity of any one of the graph's adjacency matrices (Neel & Orrison, 2006).

Complex systems are a rather recent discipline of research in systems architecting and undoubtedly, establishing a disciplinary definition of complexity will require further research and discussion among the community. A number of entities initiated research efforts in the field, like the Santa Fe Institute and the New England Complex Systems Institute (NECSI). This thesis does not aim at advancing complex systems theory. Rather, complexity is looked at from a practical standpoint: a tractable definition of complex systems is needed in the field of systems

architecting to be able to devise suitable modeling approaches to tame complex architectures. The next section reviews literature on complexity definitions, measures, and characterizations, and on emergence, which is often related to complexity in systems.

2.2 Complexity and Emergence in Systems Architecting

2.2.1 Systems Architecting

To derive an understanding of complexity in systems architecting, systems and systems architecture are first briefly defined.

A system is a construct or collection of different elements that together produce functionality not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents (NASA SE Handbook, 2007). Complexity has also been studied together with the purpose of understanding the physical architecture of a product with many parts connected through various interfaces. Techniques and tools such as graphs, diagrams, or design structure matrices (Eppinger & Browning, 2012) for example, have been developed to document system decomposition and integration.

Systems architecture refers to an abstract description of the entities in a system and the relationships between those entities, i.e. a model of the system. Architecture is important for all technical systems, e.g. infrastructure, products and services, software and hardware, and networks (Crawley et al., 2004). Systems architecting refers to the practice of architecting a system, i.e. a method to handle or build systems in a way that supports reasoning about these objects' architecture. Architectures arise in the process of deliberately designing a novel system (e.g. satellite) or by evolution from previous designs with strong legacy constraints (e.g. urban transportation systems). They can arise from complying with regulations, standards, and protocols (e.g. internet protocols), form the accretion of smaller systems with their own architectures (e.g. regional electric power grids), or from the exploration of new requirements via a user-architect dialogue (e.g. changes to a new version of the iPhone).

The architecture of a system comprises the structure and relationships, the properties (function, "ilities"), behavior and dynamics of the system. Some behaviors are sought as part of achieving the system's functions or ilities, while some are unanticipated: they *emerge* from the system

being put into motion, or used in unanticipated contexts of operations. Systems architecture is a matter of viewpoint on the system. Finally, the architecture of a system is a determinant of its complexity. For example, as modern systems grow in scale (e.g. electric grids) their complexity becomes overwhelming: it limits ones ability to predict its behavior or change its structure.

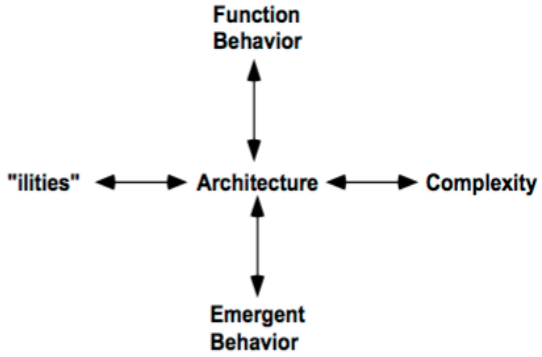


Figure 2-1: Architecture plays a Central Role in Giving a System its Behavior and "Ilities", as well as generating Emergent Behavior and Complexity, from (Crawley et al., 2004) **, p.3**

A system's architecture evolves over time, and this is especially true for long-lived systems (e.g. multi-modal urban mobility systems) and enterprises that must repeatedly be re-organized following the addition or suppression of elements. Systems architects seek to minimize the impact of legacy constraints on achieving future system requirements.

2.2.2 Complexity

In general, "complex" refers to anything that is found to be difficult to understand. The online Oxford dictionary defines complex as "consisting of many different and connected parts - not easy to analyze or understand; complicated or intricate". According to this definition, complexity in systems could either apply to the architecture of the system itself, the arrangement of elements and subsystems independently of the observer, or to the system as it is perceived by people designing it, developing it, managing it (Flood & Carson, 1993; Sheard, 2012). Both the objective and subjective approaches are present in complex systems literature (Figure 2-2). Both are relevant to systems architecting: the complexity of the architecture itself, and how the architect and stakeholders perceive it. Although it can be argued (Senge, 1990; Snowden &

Boone, 2007) that objective complexity is akin to *complicatedness* and that real *complexity* implies human perception of the system, this thesis will use the term complexity to designate both sides of the coin.

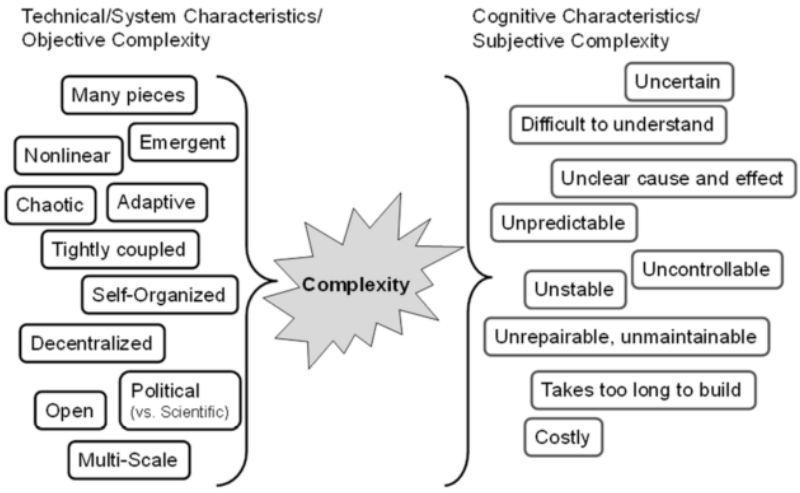


Figure 2-2: Objective and Subjective Complexity, from (Sheard, 2012) p.54

Given the above definition, one is tempted to believe, at first, that complexity is undesirable, because it makes things more difficult to understand or analyze. However, this is forgetting that today's most complex systems also achieve much more functionality than previous simpler systems. For example, the evolution of computers has allowed creating powerful AI programs. "Complexity is driven into systems by asking *more* of them: more function, more performance, more robustness, and more flexibility. It is also driven into systems by asking them to work together, to interconnect" (Crawley, Cameron, & Selva, 2015). This view of complexity supposes that all complexity isn't negative, that there is a required minimum complexity to design into a system for it to achieve a given function, to reach an expected performance level, or to display a required "ility". Adding more value comes to the price of increasing complexity. Imposing more requirements comes with higher minimal complexity (a.k.a. essential complexity). Complexity has been defined as the uncertainty of achieving certain requirements, which increases as the number and interdependence of requirements increase (Suh, 2005).

However, any complexity added over and above essential complexity for any given functionality is superfluous: it is undesirable, gratuitous complexity (Crawley et al., 2015). However, there is no clear boundary as to what is essential and what is gratuitous. Adding redundant parts to a satellite system can be seen as adding gratuitous complexity, while it can also be seen as essential to system survivability.

Finally, apparent complexity is how complicated the system is from the viewpoint of someone interacting with it, how hard it is to understand. It has the potential to scale up with the number of features, functionalities and/or the number of objects and depends on the observer (Crawley et al., 2015). Sociotechnical systems by definition connect technical systems with humans therefore apparent complexity will be important to consider.

2.2.3 Complex Systems

Literature is profuse with definitions of complex systems: "one made up of a large number of parts that interact in a non-simple way" (Simon, 1991), or "a system is complex if it has many interrelated, interconnected, or interwoven entities and relationships" (Crawley et al., 2015), or "composed of interconnected or interwoven parts" (Maier, 2009).

A few characteristics of complexity were outlined but many more could be included into a general definition of a complexity, like notions of time variance, time scales, spatial scales, non-linearity and short vs. long run behaviors (Sussman, 2002). As underlined previously, complexity is related to one's intent with the system. A system displays as many types of complexities as people interacting with it. It would be useful to have a taxonomy to categorize system complexity, yet the profusion of complexity taxonomies in literature is impractical and often *ad hoc*. The following sections will review these taxonomies around recurrent themes (static vs. dynamic, and technical vs. human and social aspects), but for the purpose of this thesis, it is useful to have a tractable definition of a complex system.

A common trait of a complex system is the multiplicity and heterogeneity of elements and connections between these elements. In sociotechnical systems engineering, *elements* would designate components or subsystems (hardware, software, facilities, parts, process steps), humans and social entities (organizations, roles, teams etc.). *Connections* would be interfaces: strictly technical links (logical, functional, spatial, temporal), human-technical interfaces (a

monitor as an interface between the operator and the system, a model as an interface between the architect and the system), and strictly social relationships (in a social network, an organization).

Another key trait of complex systems is the emergence of systemic properties or behaviors. "In such systems, the whole is more than the sum of its parts […] given the properties of the parts and the laws of their interaction, it is not a trivial matter to infer the properties of the whole" (Simon, 1991). "A system is complex when it is composed of a group of related units (subsystems), for which the degree and nature of the relationships is imperfectly known. Its overall emergent behavior is difficult to predict, even when subsystem behavior is readily predictable" (Sussman, 2002). This poses an acute problem for systems architects because emergence makes it difficult to "describe, understand, predict, manage, design, and/or change" complex systems (Magee & de Weck, 2004). In other terms, emergence defies the purpose of a system, hence makes systems architecting a more difficult task.

For the purpose of this thesis, the following definition of complex systems is adopted:

A complex system is a system composed of a many interconnected elements (technical, human, social…) that interact in varying time scales and magnitude, such that the system is difficult to describe, understand, predict, manage, design, and change, and that it might display emergent phenomena.

2.2.4 Emergence in Complex Systems

Emergence in systems generally splits into two perspectives. The first describes emergence as a property or behavior that is not possessed by any of the components, but that is designed-in to emerge at the system level from their interactions. The very definition of systems in a sense already includes this: A "system" is a construct or collection of different elements that together produce functionality not obtainable by the elements alone (NASA SE Handbook, 2007). Similarly, a system of systems performs functions and carries out purposes that do not reside in any component system (Maier, 2015). The second definition of emergence is of greater interest to this thesis.

We define *emergence* as *any observed property or behavior of a system that was not intentionally designed into it, or anticipated prior to the operation of the system*. Cascading

failures and accidents are a typical case of such emergence inside a system. “Normal accidents” (Perrow, 1999) are those that begin with a discrete, benign technical mishap. For example, they can be problems affecting one sub-system, for which there is a built-in safeguard (e.g. local component redundancy). Benign problems can spin out of control and trigger a series of technical cause-effect chains because of misaligned or absent operator intervention. Interactive complexity arises when two - or more - benign failures happen concomitantly and interact in unexpected ways. A complex system can be expected to exhibit many unanticipated failure modes because of dense system connectivity, which favors failure propagation. If emergence is by definition unpredictable, one may however ask if there are indicators of its likeliness, or principles for emergence management (Crawley et al., 2004).

Some system attributes are considered as a *potential* for complexity, in that they set a favorable stage for emergence to arise but don’t necessarily entail emergent behavior/properties. Examples of these attributes are number and variety of elements, strength and number of interactions, number of degrees of freedom, nonlinearity of input/output relationships, asymmetrical architecture, and path dependent constraints (Flood & Carson, 1993).

In sociotechnical systems the added human variability and intrinsic unpredictability of non-deterministic, autonomous decision-making entities (e.g. individuals, organizations) increases the potential for emergence. Human beings are difficult to understand, they have their own values, beliefs, and make decisions that are, to some extent predictable and modeled, but remain uncertain (Flood & Carson, 1993).

Senge (1990) argues that our perception of phenomena as emerging from complex systems is due to our inability to see and understand the whole system structure and behavior (i.e. architecture) from a dynamic perspective. Sophisticated computational forecasting and analysis tools are designed for tackling *detail* complexity: problems of many variables that can be decomposed and solved at a granular level, then integrated to a system level. This logic is effective with the first type of emergence mentioned above, but fails to produce correct outcomes when there is dynamic complexity, when elements influence each other’s behavior, and there is emergence of the second type. Maier (2015) defines four types of emergence, depending on whether the

systems architecture lends itself to a decomposition-integration approach, and whether modeling and simulation have predictive effectiveness about emergence (Figure 2-3).

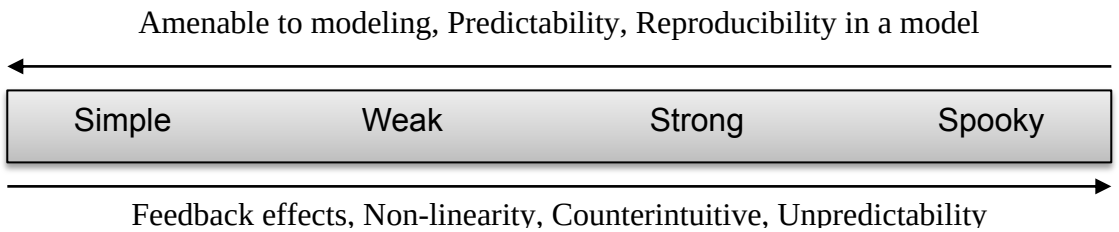


Figure 2-3: Types of Emergent Properties, adapted from (Maier, 2015)

Simple emergence can be readily understood and predicted from system components and reproduced in less complex, abstracted models of the actual system. Weak emergence can be understood and consistently predicted by reduced simulation models; non-simulation models can explain it after observation, but not predict it consistently in advance. Strong emergence is consistent with the properties of system components but can't be reproduced by any simplified model of the system and direct simulations of the system may reproduce the emergent property, but inconsistently and unreliably. Spooky emergence is inconsistent with known properties of the system's components and even a simulation with complexity equal to the real system can't reproduce them.

Traditional systems architecture paradigms, frameworks and other formalisms adopt a decomposition-integration approach. Complex systems with emergent behavior call for another approach: moving up the scale of emergence (Figure 2-3) the decomposition-integration logic is less and less effective in deducing system-level behaviors and properties. Snowden & Boone (2007) derived a set of principles in organizational management of complex enterprises, which could be adapted to systems architecting strategies: for simple systems "sense, categorize, respond", for complicated systems "sense, analyze, respond", and for complex systems "probe, sense, respond". Emergent, complex systems demand an experimental approach to be dealt with, instead of a prescriptive one.

Emergence is often assumed to be negative (e.g. cost overruns, schedule delays, performance shortfalls), and therefore being able to identify and deal with tackle it early in the systems development life cycle would allow reducing the magnitude of its impact, short of preventing it (Sheard, 2012). On the bright side it has been claimed that there are things emergent systems can do, that other systems can't (Bouarfa, 2015), and that they are robust and resilient. There is no single point of failure, so if a single unit is no longer functional, the system still works. Emergent systems are flexible thus well suited to a "messy" real world. Engineered systems may be designed to be "optimal" for a given concept of operations, but will require much design effort and could be fragile or resistant to adaptation and change in the face of unanticipated context shifts or perturbations. In real situations, time matters, and viable solutions need to be taken while they are still relevant, be they sub-optimal. For example, computationally heavy algorithms can take much time to converge to an optimal solution, which can come too late in time-critical systems.

2.3 Views of Complex Systems

2.3.1 Static vs. Dynamic

Detail complexity characterizes systems with many components, problems with many variables, and large hierarchical structures but with little lateral links between elements (Senge, 1990). It can be handled with appropriate tools (e.g. databases, diagrams, queries, statistics, numerical solvers…) and sufficient resources (computational power, time, fast algorithms, expertise…). Combinatorial complexity is a similar concept: it refers to the number of, or links among, the elements of a system, or the dimensionality of a search space (Sterman, 2002). Dealing with these types of complexity is *analogous with* finding the solution to a large set of equations, which to certain extents can be executed computationally. Advances in applied mathematics and computer power enabled solving such problems numerically. Computational fluid and structural dynamics, and the development of software and simulation environments, have enabled mechanical engineers to optimize the design of aerodynamic bodies with CAD. These are complex problems with coupled fluid and structural equations and nonlinear behaviors (e.g. elasticity-plasticity transition), involving computational intensive calculations (that increase with the refinement of the grid used and the number of equations solved for).

Solving for detail or combinatorial complexity requires knowing the variables that matter in the problem. In complex systems with soft value-laden problems, like sociotechnical systems, these variables are often non-obvious. In systems with time-shifting environmental conditions, the problem can be reformulated with each context shift, into a priori unknowable variables and equations. The levee system along the Mississippi River is a good example of a time-variant problem. The first levees along the lower Mississippi were built to protect farming land along the river from floods. They had to be heightened continuously between 1850 and 1927 because the bed of the Mississippi River elevated itself due to increased sediment confinement, caused by levee construction. Of course, this system's evolution was subject to other technological, political, and economic influences. But failing to see the riverbed phenomenon as *dynamic* drove systems architects to pose and solve the problem in static equilibrium terms instead of dynamic equilibrium terms.

Short-term or static solutions to a complex problem may be significantly different from long run solutions as it appears in the Mississippi example. A similar logic applies to spatial horizons. Consequences to an action can be different locally and in another part of the system. When a system displays such behavior, it displays *dynamic complexity* (Senge, 1990; Sheard, 2012). Dynamic complexity is oftentimes non-obvious to systems architects. Simulations are one way of getting around this myopia, because they allow insight into what may happen in a hypothetical future (scenario) based on current beliefs. However, simulations are no guarantee against unpredictability and emergence, because they contain their own biases and limitations by construction: assumptions, input/output formats, time step discretization, boundary conditions and initial conditions… Simulation models will be discussed further in chapter 4.

Detail (aka combinatorial, or structural) complexity is quantifiable from a "snapshot" of the system: it is related to the size, the connectivity and the degree of heterogeneity in the system (Sheard, 2012). Dynamic complexity is less easily identified. It manifests itself through mismatches between expected and actual performance *a posteriori*, i.e. emergence. For example, a manager oblivious to enterprise culture might find that it is complex to achieve target performance levels because knowing its hierarchical organization and work processes extensively – detail complexity – does not necessarily give insight into employee motivation drivers and productivity factors – dynamic complexity.

Dynamically complex systems are perceived as counterintuitive and policy resistant. They are constantly changing, tightly coupled, feedback-governed, nonlinear, history-dependent, self-organizing, adaptive, and characterized by trade-offs (Sterman, 2002). Agents within the system interact and adapt dynamically. Over time, aggregate system-level phenomena emerge. "The behavior of a system arises from its structure […] the feedback loops, stocks and flows, and nonlinearities created by the interaction of the physical and institutional structure of the system with the decision-making processes of the agents acting within it" (Sterman, 2002). In the example above, the enterprise structure (e.g. hierarchy, processes) may be well known, yet enterprise behavior may be elusive. To infer a certain behavior about the system (Kreimeyer, 2011), an architecture representation should include static and dynamic views of:

- Technical systems
- Human/social systems
- The environment
- Relationships between technical systems, human/social systems, and the environment

2.3.2 Technical vs. Social/Human

A recurrent theme in literature is the dichotomy between physical/technical and human/social aspects of complex systems. Flood & Carson identified a few factors contributing to one or the other complexity (Flood & Carson, 1993), represented in Figure 2-4.

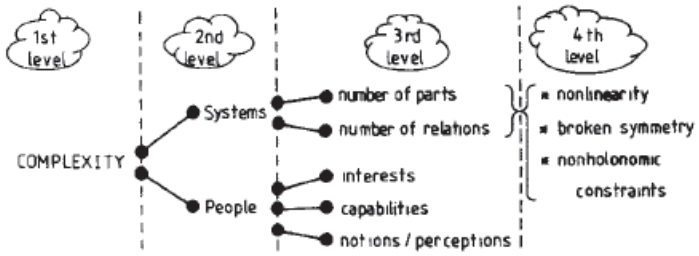


Figure 2-4: Disassembly of complexity, from (Flood & Carson, 1993), p.25

In an analysis of complex, large scale, interconnected, open systems (CLIOS), Sussman (2007) describes *structural* and *behavioral* complexity on one hand, and *nested* complexity on the other.

Structural complexity "exists when the system consists of a large number of interconnected parts" (cf. detail, combinatorial complexity). Behavioral complexity "exists when predictions of system outputs or behavior is difficult" (e.g. with feedback loops, nonlinear transfer functions, delays…). Both the technical *and* institutional systems exhibit structural and behavioral complexity in their own right. The interactions between both systems create *nested* complexity. Lastly, the complexity related to multi-stakeholder systems, is referred to as *evaluative* complexity (Sussman et al, 2007).

Social aspects of complexity – other than stakeholder requirements – are generally not explicitly accounted for in systems engineering processes. Sheard defines structural and dynamic complexity in the technical sphere, and groups under socio-political complexity "anything having to do with humans" (Sheard, 2012). That includes cognitive limitations, sociological phenomena such as fads and marketing, economics, environmental sustainability, politics, and the diversity of concerns within the set of stakeholders. A more practical definition of socio-political complexity would be one that decomposes it into sub-concepts: what is relevant to engineering, what is to social sciences, what is to economics etc.

In sociotechnical systems research, the dichotomy between the technical on one side, and social and human on the other side is salient (Drezner, 2009; Flood & Carson, 1993; Lessard, Sakhrani, & Miller, 2014; Sheard, 2012; Sussman et al., 2007). Authors emphasize one or the other depending on their background, the nature of the system at hand, and the purpose of studying that system, i.e. the problem. Problems are usually formulated either in technical, structural terms, or in cognitive, institutional, organizational, or socio-political terms.

Architecting a sociotechnical system is a different task than architecting a purely technical system. There are several causes for this, including human unpredictability and intellect. Humans can make decisions based on past patterns of success and failure, rather than on logical, rational, pre-defined rules (like automata). They may have different identities and switch unpredictably from one to another (shift in preferences, in behavior, in performance…).

Mostashari (2011) separates socio-technological systems into two interacting and tightly coupled networks: "one layer includes physical/technological components of the system, and the other

layer the social/institutional components of the system, which are usually connected through an information network" (Figure 2-5).

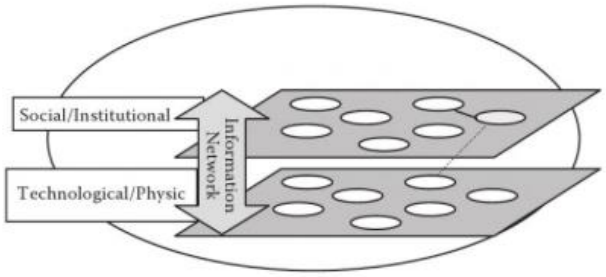


Figure 2-5: Sociotechnical System Layers, from (Mostashari, 2010), p.3

In the field of organization and project management, which can display human/social complexity because people are involved, complex situations defeat traditional command-and-control management styles (Snowden & Boone, 2007). Instead, social/human complexity calls for an experimental mode of management, in other words setting a favorable stage, stimulating the system (e.g. a group of stakeholders, an online social network), allowing patterns to emerge and learning from them. This requires time, a certain tolerance for failure and the capacity to learn and adapt one's strategy which are all characteristic of empirical endeavors. Snowden and Boone suggest some practical management practices, such as:

- Create environments and experiments that allow patterns to emerge
- Increase levels of interaction and communication
- Use methods that can help generate ideas: Open up discussion (as through large group methods); set barriers; stimulate attractors; encourage dissent and diversity; and manage starting conditions and monitor for emergence

If technical and human/social complexities are accepted as different albeit connected concepts, the latter consists of two different dimensions: the social/institutional and the human issues. Although society and institutions are made up of humans, the following two sections explain how these two issues differ, and why the differences aren't just of scale.

2.3.3 From Human to Social Complexity

Rouse (2015) proposes reading sociotechnical systems (STS) through a multi-scale lens, where the smaller scale is the human scale, and the larger scale is the social scale. Work processes, information flows, and economics bridge social-level with human-level phenomena. Specifically, social complexity is the complexity of organizations, their structure, internal allocation of roles and resources, and information networks. It is also the complexity of the system of values and norms that shape society into castes, constituencies, coalitions, that motivate negotiations. Figure 2-6 is a visualization of the human-to-social hierarchy on an example STS: a congestion pricing system for traffic management.

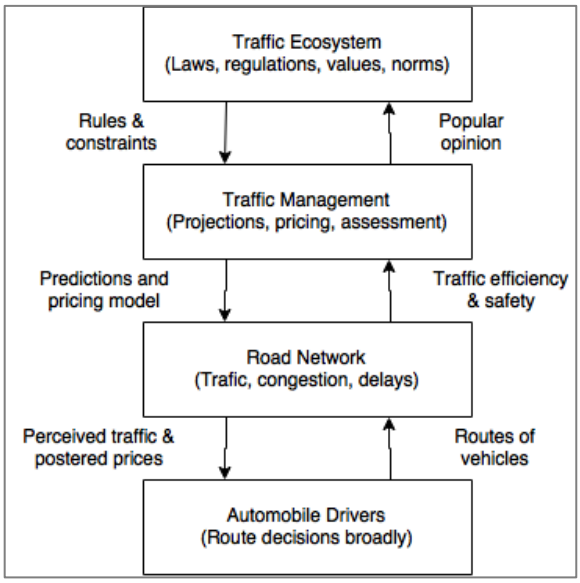


Figure 2-6 Hierarchical Visualization of Phenomena in a Congestion Pricing Traffic System, from (Rouse, 2015), p.38

Implementing effective traffic control in large, densely populated urban areas requires understanding several phenomena. An obvious phenomenon is the architecture of the road network: the physical layout and traffic flows/patterns. The physical layout limits the degrees of freedom each commuter has, and historical and current traffic information is an input for driver

decisions. Route decisions remain in the hands of humans hence they are non-deterministic. Physical traffic on the road network provides performance metrics used at the traffic management level to devise dynamic pricing strategies that are fed back into the road network and drivers (e.g. toll pricing, parking fees).

Given modern communication technologies, effective traffic control needs to account for the information networks that connect drivers. The previous discussion leads to believe that drivers decide autonomously their next action, based on local traffic information. However, communication technologies enable drivers to share routes and shortcuts, to get real-time traffic congestion updates etc. This decentralized network is under no central control. The number of drivers in the network can be significant, and they can be connected through several information channels (radio, social networks, Maps, Waze, in-car GPS devices…). The itinerary each driver takes follows an individual logic that is rational to some extent (e.g. minimize time or distance), but not always (e.g. flocking), and influenced by social behavior on the network scale.

Similarly to Rouse, Murman & Allen (2003) differentiate six levels in the social dimension of aircraft engineering projects (Table 2-1). Human stakeholders at different levels of complexity play different roles: design teams, flight crews, and cabin crews; organizations that develop, manufacture, operate and support the systems; society that interacts with it in many ways.

Table 2-1: Levels of the Social Dimension, from (Murman & Allen, 2003)

Society Nations, communities, etc.
Extended multi-organization enterprises Including partners and suppliers, several business units
Single organizations Business units, divisions, with one or more programs, projects …
Organizational units Independent programs, projects
Working groups Project teams, operations teams
Individuals

2.3.4 Perceived vs. Objective Complexity

Psychology and cognitive sciences address the complexity of interactions between a person and a component, sub-system, or system: the word “complexity” is used in the context of performing tasks with a technical or technological object, of solving problems, of devising a strategy in the face of a perceived situation or object. From a psychology point of view, complexity can be broken down into the complexity potential of the object itself independently from any person interacting with it, the complexity potential of the task to be performed, and the perceived complexity of the person (Figure 2-7).

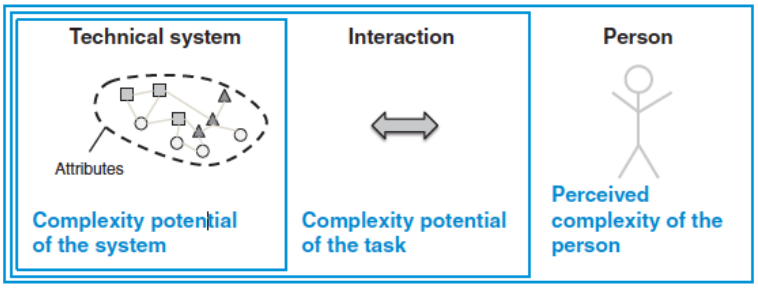


Figure 2-7: Representation of Complexity in a Sociotechnical System, from (Schöttl & Lindemann, 2015), p.4

Operators are limited in their ability to analyze a large number of subsystems, especially when subsystems are tightly coupled and influence each other in great magnitude or faster than the operator can keep up with (Perrow, 1999). This mismatch between human capabilities and technical features contributes to cognitive complexity. System operators make decisions, sometimes with incomplete, ambiguous information, following a mental model of the situation, sometimes responding to rapidly evolving situations, involving a large number of variables to consider.

Industrial processes have become more complex owing to technological developments of hardware, software, and automation. In this context, human reliability and productivity has become a weakness in operator-supervised systems (Li & Wieringa, 2000). Operator-perceived complexity depends on objective factors (task, human machine interface and process designs), operator specific factors (e.g. training, experience, creativity) and the strategy of operation (e.g.

procedure) imposed. The contribution of these factors to perceived complexity is illustrated in Figure 2-8.

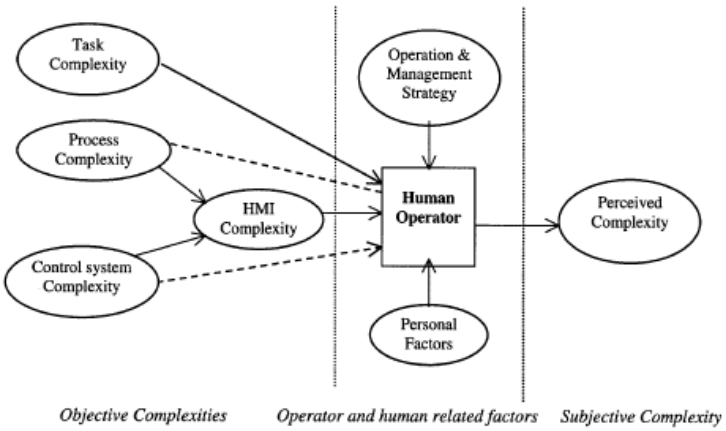


Figure 2-8: Conceptual Framework for Perceived Complexity in Human Supervisory Control, from (Li & Wieringa, 2000), p.77

Air traffic control is an active area of cognitive complexity research (Histon & Hansman, 2002; Histon, Li, & Hansman, 2010; Histon, 2008; Xing & Manning, 2005; Xing, 2007). Cognitive complexity is the complexity of the mental model of the real system that operators construct to represent themselves the situation and task to perform (Histon & Hansman, 2002). The air traffic situation (system state) is processed through this mental model, and results in the operator deciding on a course of action (commands), which affects the system state. This feedback structure is illustrated in Figure 2-9.

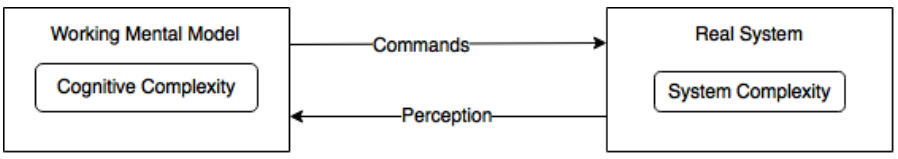


Figure 2-9: Notional Representation of the Feedback Loop between Cognitive and System Complexity, adapted from (Histon & Hansman, 2002)

2.4 Characteristics of Complexity in Systems

Since a decomposition-integration description of complex systems is ineffective, another approach at describing complex systems is through characterization. Sets of complexity characteristics are found across literature, in slightly varying terms. A set of characteristics compiled from (Sterman, 2002), (Righi, Wachs, & Saurin, 2012) and (Sheard, 2012) is summarized below, where similar concepts are grouped from **I.**(Sterman, 2003), **II.**(Righi et al., 2012), and **III.**(Sheard, 2012).

- *No equilibrium when observed on several time scales*

I. Constantly changing system: what appears to be unchanging is, over a longer time horizon, seen to vary. Change in systems occurs at many time scales, and these different scales sometimes interact.
II. Complex systems work far from equilibrium: a steady flow of energy keeps the system running
III. Evolving problems and therefore constantly changing needs

- *Coupled mechanisms, undesired cascading effects ripple across the system*

I. Tightly coupled: The actors in the system interact strongly with one another and with the natural world. Everything is connected to everything else. […] "You can't do just one thing."
II. Complex systems exhibit tight coupling: with constantly changing demands and resources; with time pressure on performing actions; and with effects that quickly propagate themselves throughout the system
III. Difficulties in figuring out where problems come from, because cause and effect are interwoven

- *Prescribing changes/values to variables is difficult because of feedback loops*

I. Governed by feedback: Because of the tight couplings among actors, our actions feed back on themselves. Our decisions alter the state of the world, causing changes in nature and triggering others to act, thus giving rise to a new situation which then influences our next decisions
II. The interaction in complex systems is potentially rich, since the elements influence and are influenced during the process

- *Nonlinearity*

I. Effect is rarely proportional to cause, and what happens locally in a system (near the current operating point) often does not apply in distant regions (other states of the system). Nonlinearity often arises from the basic physics of systems (e.g. bounds, thresholds, tipping points, carrying capacity...)
II. The interaction in complex systems is not linear: the sequences are not usually rigid; the interaction offers, in general, two or more options for decision-making; and small causes can have large consequences
III. Difficulty of predicting future states or behavior because of non linearity

- *History-dependence: irreversibility, learning, legacy systems*

I. History-dependent: Taking one road often precludes taking others and determines where you end up (path dependence). Many actions are irreversible […] Doing and undoing have fundamentally different time constants.
II. Complex systems have a history: which influences attitudes in the present

- *Macro scale behaviors emerge from micro scale interaction rules, spontaneously (i.e. without having been designed in the system)*

I. Self-organizing: The dynamics of systems arise spontaneously from their internal structure. Often, small, random perturbations are amplified and molded by the feedback structure, generating patterns in space and time […] spontaneously.
II. Complex systems exhibit emergent phenomena: situations arising from interactions that cannot be predicted.
III. Self-organizing behavior (entropy decrease), competition, and increase in complexity and specialization over time

- *Adaptive: Element behaviors evolve over time. These evolutions aren't designed for, they're spontaneous adaptations.*

I. The capabilities and decision rules of the agents in complex systems change over time. Selection and proliferation of some agents while others become extinct. People learn from experience.
II. Complex systems exhibit behavior which is between stable and unstable: the behavior of the elements changes as needed, without the systems being aware of these elements
III. Co-evolution resulting from adaptation to environment and vice-versa

- *Counterintuitive for whoever attempts to design, control, manage the system*

I. In complex systems cause and effect are distant in time and space while we tend to look for causes near the events we seek to explain. Our attention is drawn to the symptoms of difficulty rather than the underlying cause. High leverage policies are often not obvious.
II. Complex systems exhibit an indirect and distributed character of information Complex systems cannot be understood from a study of their parts Complex systems exhibit their elements but are unaware of the behavior of the system as a whole
III. Difficulties in figuring out where problems come from, because cause and effect are interwoven

- *Resistant to traditional design approaches, difficult to steer, develop, implement, and manage…*

I. The complexity of the systems in which we are embedded overwhelms our ability to understand them. As a result many seemingly obvious solutions fail or actually worsen the problem. Characterized by trade-offs: Time delays in feedback channels mean the long-run response of a system to an intervention is often different from its short-run response. High leverage policies often cause worse-before-better behavior, while low leverage policies often generate transitory improvement before the problem grows worse.
II. Complex systems require both standardized and new solutions: no situation deemed as simple or complex Complex systems exhibit the uncertainty factor in their interactions: there is not a route to the answer that is fully known
III. Difficulties in developing because of unseen risks

- *Open to their ecosystem, fluxes at the interface, field forces…*

II. Complex systems are open systems: the constant interaction with the environment makes it difficult to set the boundaries of the system

These characteristics apply across many complex systems, including those considered as case examples in this thesis, and are valid indicators of complexity, but they are not a useful framework for disentangling and describing complexity in systems. This thesis proposes a STS complexity taxonomy (section 3.1) that synthesizes major threads of complexity research identified in the literature review. A discussion of three case examples exhibits different flavors of complexity in systems of different nature (section 3.4). A phenomena-based description of systems architecture is proposed to reflect these flavors of complexity (sections 3.3 and 3.5), and prompts one to devise different modeling approaches for sense-making of systems architectures exhibiting multiple phenomena of different types (section 3.6).

3 Phenomena-Based Description of Complex Sociotechnical Systems Architecture

3.1 Complexity Taxonomy

We begin with a synthesis of the findings from an extensive literature review about technical, human and social, and environment complexity types and factors (see appendix).

3.1.1 Technical System Complexity

Technical system size, heterogeneity and structural connectedness are static complexity observables that can be identified from a "snapshot" of the technical system. The number and variety of subsystems, number and variety of functions (transform or process, transport/distribute, store/house, exchange/ trade, control/regulate) (Magee & de Weck, 2004; Martin, 2004), the number and variety of connections between them (matter, energy, information, or value links), and the spatial extent of the system contribute to increasing the amount of information needed to describe and understand system structure (Martin, 2004). Tangible subsystems (infrastructure, equipment, software, parts…) can easily be identified and quantitatively described. Less tangible, logic/control components also contribute to complexity (software, programs, procedures).[3]

Additionally, technical systems display dynamic complexity observables in the processes that are executed over time. The heterogeneity and number of steps and number of actors involved in a process increase its complexity. With a given number of steps, the structure of a process may be more or less complex: the dependencies among steps in a process, and among processes, can be linear (e.g. output-input), coupled (e.g. feedback, control), parallel (e.g. resource sharing, concurrent timing) or any combination of the above. This dynamic process complexity adds to the above static structural complexity, and both drive technical system complexity.

[3] Variants of this concept are named differently depending on the field and background of the authors: combinatorial complexity (Sterman, 2002), structural complexity (Sheard, 2012; Sussman et al., 2007), detail complexity (Senge, 1990), architectural complexity (Dwyer, 2014) or technical system complexity potential (Schöttl & Lindemann, 2015) are concepts that all share this common notion of complexity.

The task of designing or changing a complex technical system is more intricate since changes in one place can branch out to large parts of the system. This is commonly referred to as local cause – global effect behavior in complex systems. The more connected a system, the more difficult it is to architect: dependencies and causal links between requirements, components, subsystems, functions, and processes propagate actions on the system to unintended places in the system. This is especially relevant to long-lived, partially designed, partially evolved systems that are a combination of various technological standards (e.g. software versions) due to incremental modifications. Any modification to a densely connected technical system poses interoperability challenges: interoperability of future capability requirements with constraints from past architectural choices.

Deterministic properties of systems are less complex to apprehend than stochastic properties that face some degree of irreducible uncertainty (e.g. component failure represented in probabilistic terms). The accumulation of uncertainties in many sub-systems may make it difficult to infer system-scale behavior a priori (e.g. safety, and performance) and even when behavior is readily observed, to relate it to its origin(s) (e.g. cascading failures). Coping with uncertainty effects at the system scale is a complex problem. For example, adding redundancy increases the probability of system-level safety, but to the expense of increasing system size i.e. structural complexity.

Software and algorithms have received a great deal of attention for their growing pervasiveness and complexity, fueled by (and motivating) progress in computational power. The fact that they can execute on much shorter time scales than human supervision, judgment, and reaction makes software intensive systems powerful but vulnerable to failure (e.g. infinite loops, errors). Their size and logical structure display structural complexity and several metrics have been proposed to measure this complexity (e.g. cyclomatic complexity, algorithmic complexity).

3.1.2 Human & Social Systems Complexity

This section shifts to individuals, stakeholders, organizations and society in systems. Observables here differ not only in terms of population sample size. Human roles and interactions with systems are diverse, both in the operation of the system (user/ beneficiary, operator, manager, maintainer, trainer) and its development (architect, financial/economic

stakeholder) (Newman, 1999). The study of human and social systems calls for different observation scales but also different observation disciplines, such as cognitive sciences, economics, organizational science, and social sciences.

The study of human-level complexity is much about understanding the determinants of task performance (e.g. air traffic controller overseeing multiple flights at the same time). Performance factors include task factors (e.g. workload, diversity, emergency, predictability, standard or tactical) and operator factors (e.g. expertise, training, experience, stress, accountability, fatigue). Information processing is also an active area of research (e.g. human-system interface design, cognitive processes). How people deal with incomplete information in limited time has many applications to supervisory systems design (e.g. power plant fault detection). How people deal with large, heterogeneous information volumes has applications in human machine interface design (e.g. control tower displays). The structure of human mental models for information processing, decision and translation into action, and the underlying emotional and rational factors (e.g. preferences, social influence) are of interest at the human level (e.g. criteria and decision process for selecting a multi-modal commute from point A to point B). The growing pervasiveness of automation in systems calls for close examination of the interactions between humans, automata and systems.

In organizations, complexity may stem from size, social network structure, spatial distribution (e.g. a humanitarian organization with warehouses, vehicles, headquarters and operations distributed worldwide), and clarity of responsibilities within the organization (e.g. roles definitions in an airport collaborative decision-making system). Organizations are multi-scale social phenomena. Managers will find that an organization is policy resistant when they overlook important individual level factors (definition of roles, accountability; perception of strategy) and focus on institutional level factors (e.g. strategy definition, performance, investment, competition, productivity). Organizational complexity can be somewhat informed by the examination of human complexity, as organizations are aggregations of individuals.

Different stakeholders have different interests and impose different requirements. The larger and the more heterogeneous the stakeholder landscape, the larger the probability of conflicting or unrealistic requirements, hence the more difficult decision-making will be. Heterogeneity may be

in terms of priorities, resources, performance evaluation, political leverage, size, values, culture etc. The more requirements are qualitative (e.g. system ilities), soft, ill-defined and creeping, the less mature techniques developed for technical systems architecting and geared for dealing with known, quantitative problems are effective (e.g. MDO).

Finally, societies can be integral parts of designing a system (e.g. persuasive urban transportation systems, online social networks). The use that society makes of the system can be so important to system design that people (e.g. commuting workforce, Facebook account owners) can be considered as stakeholders, albeit indirectly. The size, heterogeneity, soft-value laden character, and interconnectedness of social networks are difficult to account for.

3.1.3 Ecosystem Complexity

The ecosystem in which the system lives may influence both the technical and social domain. The more open and densely connected to its ecosystem and the more sensitive to external perturbations, the more difficult system behavior is to predict, a fortiori control. Large-scale long-lived sociotechnical systems affect or are affected by many people (e.g. the healthcare system, multi-modal urban mobility systems), often system users. Agents enter and leave the system (e.g. aircraft entering/leaving an airport, donor defection/adhesion to humanitarian organizations). Energy, information and resources flow through system boundaries (e.g. raw materials/products entering/leaving a production line, cars entering/leaving a city). Systems connect to external technical systems and may even rely on these to function (e.g. urban transit rail system depending on the regional electric grid). The systems considered in this thesis all connect to technologies (e.g. satellites, radars) and infrastructure (e.g. rail and road networks, electric grid) beyond systems architecting boundaries.

Furthermore, the ecosystem is a source of soft influence, typically difficult to assess and account for in models, especially formal models. Consequently, one often simplifies or overlooks the social, political, cultural, and other soft factors, which can lead to resistance from the system to architecting attempts.

Additionally, the natural ecosystem is a factor that may impair system functionality under certain conditions, affect people's behavior, provide a more or less safe operational environment etc.

Sensitivity to natural conditions is a factor of complexity especially when natural environments are uncertain.

Finally, there are myriad phenomena at work in the environment. The time-scales of ecosystem evolution are often approximated as exogenous and modeled as such. For example, one airport does not dictate the evolution of traffic on the European level, although it might strongly perturb it locally (e.g. large hubs like Paris, London, Frankfurt). For the sake of designing a long-lived system, approximations and assumptions are made on the future sets of conditions in which the system will operate. These are plagued with uncertainty. Additionally, the ecosystem undergoes unpredictable short-lived perturbations.

3.1.4 Summary

Table 3-1 is a condensed view of complexity factors discussed above. Additionally, the use of such a “checklist” might be helpful in shedding light on important but initially overlooked complexity factors.

Table 3-1: Complexity Factors in Complex Sociotechnical Systems

	Example Factors
Technical System	**Structure: size, heterogeneity, connectedness** - Number and variety of subsystems, of functions - Spatial extent, geographical distribution of the system - Physical/structural connectivity and functional dependencies - Legacy and new systems, interoperability at interfaces **Processes** - Number of steps, number of actors involved - Process structure (e.g. linear, looped) - Dependencies between processes (e.g. parallel, resource sharing, concurrent, timing, input-output relationship) **Uncertainty** - Stochastic sub-system and aggregate properties - Behavior under unanticipated operating conditions
Human/ Social System	**Human** - Task performance: factors, variability - Human system interface: system model - mental model compatibility - Decision making process, emotional vs. rational, unpredictability, error - Human-automata interaction, shared tasks and decision support systems **Organizations** - Individual factors: roles, perceived clarity and accountability - Institutional factors (e.g. strategy, competition) **Stakeholders** - Number and heterogeneity, consensus or not - Requirements: number, heterogeneity, coupling, conflicts, creeping, soft, unrealistic **Social** - Number and heterogeneity of agents - Soft values, norms, codes - Social network structure, connectivity (e.g. decentralized, clustered) - Information network: formal and informal channels
Ecosystem	**System fluxes** - Elements (social, human, technical) entering/leaving the system - Reliance on legacy and support systems, exogenous technology - Information, energy, resources flows **Field forces** - Economic, political influences - Regulations: industry standards, market rules, legislation, safety rules - Natural environment: weather, pollution, hazards, wind, terrain - Public opinion **Dynamics** - Time scales: perturbations vs. context shifts vs. continuous evolution - Futures? E.g. unpredictable, stable, periodic, or random evolution - Sensitivity of internal behavior to external forces/fluxes/evolution

3.2 Definition of Phenomena in Sociotechnical Systems

It is useful to break down large, complex sociotechnical systems (STS) to study them. Chapter 2 explained why traditional decomposition-integration approaches are inadequate for architecting emergent, complex systems, and why a different approach is needed. This thesis describes STS in terms of technical, operational, human, social and environmental phenomena. Literally, a phenomenon is "something (such as an interesting fact or event) that can be *observed* and studied and that typically is *unusual or difficult to understand or explain* fully" (Merriam Webster online dictionary). Complex systems involve "numerous components and interconnections, interactions or interdependencies that are *difficult* to describe, understand, predict, manage, design, and/or change" (Magee & de Weck, 2004). Hence, phenomena appear as a natural approach for disentangling perceived complexity in a STS and describing system architecture.

An STS often displays several phenomena, the understanding of which will tap into different disciplines: systems science, public policy, economics, finance, engineering fields, natural sciences, behavioral and social sciences. Additionally, phenomena are observed at different social and technical scales: individual to society, subsystem to system, and task to process scales. Finally, phenomena can be static or dynamic. Rouse (2015) explores the construct of phenomena extensively: "problem solving should not begin with the selection of mathematical or computational models, but instead should commence with consideration of the *phenomena that must be understood to successfully answer the questions* that motivated modeling in the first place." Rouse (2015) proposes a classification of phenomena to study complex systems and enterprises (Table 3-2) and reviews modeling approaches prevalent in the study of phenomena in each class.

Table 3-2: Eight Classes of Phenomena, from (Rouse, 2015)

Class of Phenomena		Example Phenomena of Interest
Physical	Natural	Temporal and spatial relationships and responses
	Designed	Input-output relationships, responses, stability, optimization
Human	Individuals	Tasking, mental models, knowledge building
	Teams/groups	Team and group behavior and performance
Economic	Micro	Consumer value, pricing, production economics
	Macro	Gross production, employment, inflation, taxation, policies
Social	Organization	Structures, roles, information, resources allocation
	Society	Castes, constituencies, coalitions, negotiations, crowds, media, internet

One could argue that the classes in Table 3-2 are not mutually exclusive or collectively exhaustive for describing STS. Human phenomena are pervasive through economic, process and social ones. Technical phenomena are arguably under-represented for systems architecting purposes. Stakeholders' role in systems architecting is diluted into economic and social phenomena. Figure 3-1 is a conceptual representation of phenomena and phenomena relationships from (Rouse, 2015); it provides a high-level guideline for composing finer grained, formal models.

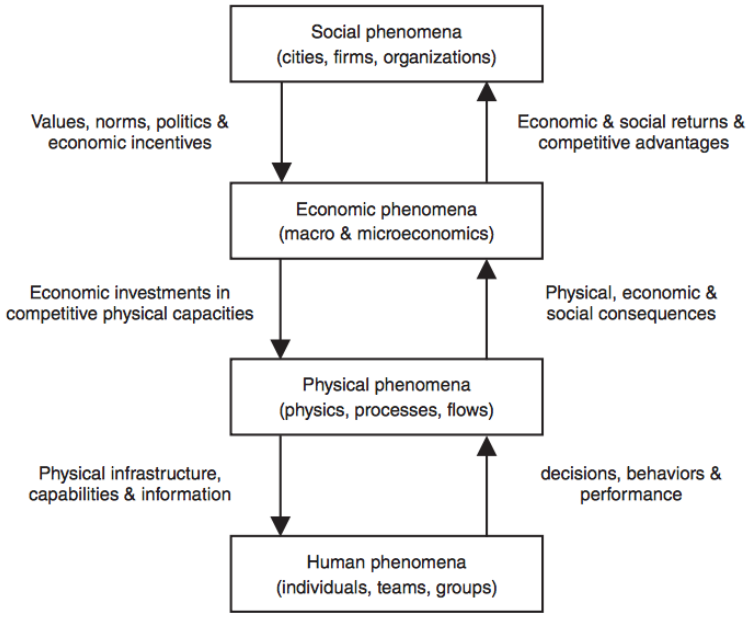


Figure 3-1 Hierarchy of Phenomena from (Rouse, 2015)

3.3 Phenomena-based Description of STS Architecture

This thesis proposes a phenomena-based description of complex STS that reflects the complexity taxonomy extracted from the literature review (social, technical and environment factors) and re-uses the phenomenon construct proposed by Rouse (Figure 3-2). The classes of phenomena that are proposed are human, social, technical and operational phenomena, examples of which are given below.

- **Human Phenomena**

Individual mental models: perceptions, beliefs, goals, decisions, intentions, and actions

Cognitive functions such as knowledge building and use, attention allocation, information processing, and emotions processing

- **Technical Phenomena (~technical Systems)**

Physical architecture: elements, and subsystems (hardware, software, infrastructure, equipment, automata, products, and services) and their connections (e.g. energy, matter, physical, information)

Functional structure: sub-system functions, dependencies on other sub-systems, and uncertainty in achieving functional requirements at the sub-system level

- **Operational Phenomena (~processes, technical and human-technical dynamics)**

Human-system interaction in tasks e.g. manual control, supervisory control

Automata behavior, e.g. robots, algorithm execution logic

Information and data flows, data processing by computer algorithms…

Transformation, transportation, and distribution of physical objects, e.g. products

- **Socioeconomic Phenomena (~behaviors of humans in social contexts)**

Stakeholder relations: attributes of interest (economic, technical requirements), performance evaluation, value, consensus, conflict and competition

Organization strategy, structure, and roles

Economics: technical and human resource allocation, performance evaluation, economic strategy making; e.g. offer and demand pricing, competition

Society [4]: social and information networks; diffusion phenomena (e.g. beliefs, products, epidemics); cultural constructs (e.g. castes, values, codes); spatial phenomena (e.g. crowds); social influence on individual behaviors

- **Environmental Phenomena (~beyond systems architecting reach)**

Technical: technological innovation, legacy systems, support systems connected to the behavior/functionality of the system

Natural: impact of weather, resources, hazards on the system

Political/Social: influence of legislation and regulations on system architecture (e.g. safety standards, market rules)

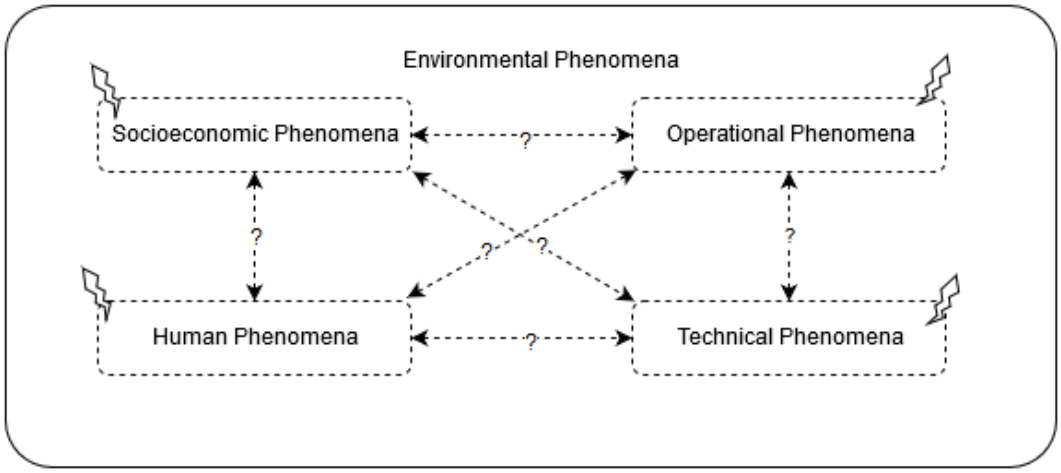


Figure 3-2: Visualization of Phenomena and Relationships

3.4 Example Complex Sociotechnical Systems

The proposed phenomena-based description is briefly illustrated on three examples of complex sociotechnical systems. First, we give an overview of the complexity perceived in each STS (when it was not possible to gather inputs from people or actual data, the examples are treated based on cross-comparison of past research and case studies as a proxy). Then each STS is

[4] Could also be under environmental phenomena depending on the viewpoint

decomposed into prevalent phenomena. It is apparent that the three examples exhibit different types of phenomena and flavors of complexity.

A preliminary list of candidate sociotechnical systems included cases of interest for complexity research was identified. This list included network centric warfare systems, urban military operations, urban mobility and multi-modal goods transportation systems, cyber security systems, infrastructure systems, airport operations, telecommunications and spectrum sharing systems, air and maritime traffic control systems, humanitarian disaster relief operations, and large online organizations and social networks. These systems were all considered as potential examples for illustrating the classification proposed above, and three were selected as a suitable subset. In order to assess how suitable a candidate example is, its nature, feasibility and tractability are considered. The nature covers the technical and social systems. It is desirable to select a set of examples complementary in nature such that different types of complexity/classes of phenomena may be explored. Feasibility expresses the quality of available information (information ease of access, depth, relevance). Tractability depends on factors like the dimensionality of data, the contemporariness of available information, the controversial character of available information and sources. A preliminary review of literature and past case study material excluded systems that were deemed less feasible or tractable (network centric military operations, military urban operations, out of data accessibility concerns), or too similar in nature (en-route air traffic systems and maritime traffic control systems, where the system's technical and social landscapes and stakes resemble those of an airport). Finally, author interest guided the down selection among shortlisted examples to a set of three.

3.4.1 Joint Humanitarian Disaster Relief Logistics

The number and impact of humanitarian disaster relief operations seems to have been increasing over the past decades. As global relief responses grow in scale and magnitude, and are scattered around the world, the associated logistics grow in complexity. Disaster response involves severe time pressure, high uncertainty in the assessment of needs, many stakeholders and several autonomous humanitarian agencies. Hazards are heterogeneous: disasters can be technological and/or natural (example: Fukushima nuclear plant hazard and tsunami flooding). Situations are highly dynamic and degrade in the absence of timely intervention. Transportation and logistics are major components of the operations of relief organizations, and improving efficiency of

transportation and logistics systems has potential to significantly decrease operational costs and to improve humanitarian relief services. However, the large number of organizations involved and the diversity of players make effective logistical coordination a challenging task. The different relief players often include United Nations (UN) agencies, international and local nongovernmental organizations (NGOs), the Red Cross Movement, governments of the affected region and military (Dolinskaya, Shi, Smilowitz, & Ross, 2011). Humanitarian logistics is different from commercial supply chain logistics because demand is highly unpredictable until a disaster strikes. Commercial supply chain logistics don't deal with unusual constraints such as substantially degraded infrastructure, terrain roughness, political instability, volatile funding from donations etc. Regardless of what's being supplied (food, water, sanitation, housing, shelter), the goals of disaster relief are to optimize inventories ahead of time and throughout the intervention, and to deliver rapidly the correct amount of goods to the needed locations when disasters strike. Essentially all players pursue this same objective, and one would expect therefore that relief be improved by their added efforts. There is wide consensus that coordination of the effort is important but its implementation (who does what and how it is done) provokes fierce debate (Reindorp & Wiles, 2001). Joint humanitarian logistics require a high degree of collaboration and information sharing and some degree of delegation of authority, hence it is problematic and the means to achieve it elusive. It can bring together players with various expertise, experience and capabilities within an affected region to share and mutually benefit from collaborative work. From a structural perspective, organizations have their own governance structure, set their own imperatives and answer different political interests. The push for profile necessary to attract funding posits other organizations as competitors. Organizations are reluctant to release personnel towards the coordination effort, rather than retaining their human resources for organization-centric operations. The urgency of relief response and pressure to distribute emergency supplies in timely manner are prioritized over forming collaborative relationships between relief participants in the short run, unless such relationships are established prior to the disaster. Information sharing about the specific needs, available commodities, transport routes, and infrastructure status is key. However, the variety of information standards used and unequal reliability of data sources makes information compiling difficult. From a technical perspective, uncoordinated use of infrastructure, equipment, fuel and other resources creates bottlenecks. If each organization locally decides on supplies to provide, it can clog the

logistical system (ports, airports, customs, warehouses, storage places…), create imbalances in the nature of goods shipped and waste of unused resources. Disaster relief logistics is a very dynamic system: planning is plagued with unpredictability about where, when and what the next disaster will be, and once a disaster strikes, the logistical response must be swift. It must deal with uncertain, incomplete information about the geographical distribution and nature of needs in the field, often reported by operatives who aren't logistics experts or trained to assess logistics needs (Jahre & Jensen, 2010; Moshtari & Gonçalves, 2012).

3.4.2 Airport Collaborative Decision Making Systems

In 2008, EUROCONTROL highlighted a serious airport capacity[5] challenge in Europe. The economic and traffic downturn since then has reduced the ability of airports to deliver the expansion plans that they reported then, and indeed reduced the urgency of the plans. Forecasts for air traffic growth and airport expansions show that by 2035, 12% of total demand will not be able to be accommodated (*Challenges of Growth 2013, Summary Report*, 2013). New infrastructure will inevitably be needed to bridge the airport capacity gap, however less costly measures for improving the efficiency of use of current infrastructure are under way: Airport Collaborative Decision Making (CDM) is an approach to optimizing resource usage and improving timeliness at an airport. In the past, airport partners/stakeholders – airlines, air navigation service providers (ANSPs) including air traffic control (ATC) and air traffic flow management (ATFM), airport operators, and ground handlers – performed their tasks more or less independently. Airport CDM, proposed by EUROCONTROL, brings together all partners to share operational data. Such information sharing is key to achieving common situational awareness, which improves decision making (Hall-May, Surridge, & Nossal-Tüyeni, 2011). The means to this end are to pool flight-related information and to coordinate flight-related activities and throughout aircraft flight phases. The resulting message traffic between partners makes for a dense information network (García, Valero, Prats, & Delgado, 2014). The IT change is one of the implementation challenges, because hitherto unconnected information systems must be adapted, made interoperable to enable data sharing. CDM essentially doesn't affect infrastructure, but it

[5] An airport's capacity is the maximum hourly traffic volume that it can accommodate, given its infrastructure, operation procedures and staffing.

does transform the communication policies and procedures that dominate operations (*Airport Collaborative Decision Making Implementation Manual Version 4*, 2012). Where human operators are involved, this poses major HMI issues, as in any attempt at changing a safety critical system. CDM is complex clockwork new and highly dependent procedures and hence requires significant staff training. The implementation project itself may suffer from reluctance or lack of interest on the part of some stakeholders. The political process between partners is likely to be contentious as different working methods, rooted within people, are challenged by the new procedures. Furthermore, ensuring that airlines will share timely and accurate information is problematic as the airline industry is competitive and data, especially if made available consistently and comprehensively, conveys much sensitive information. (*Airport Collaborative Decision Making Implementation Manual Version 4*, 2012; Corrigan et al., 2014) It has been stressed that CDM is mainly a culture change, not a technical change. Airport surface operations are highly dynamic and processes cohabitate on different time-scales: for example, the departure sequence can change every minute, whereas the airport configuration is revised hourly or daily; turnaround takes a couple hours, whereas taxi and takeoff take a few minutes. Key performance areas of safety, capacity, cost efficiency, and environment are not always aligned: for example low visibility conditions impose greater a/c separation minima, which decreases throughput and effective capacity. Many external uncertainties can significantly affect operations such as boarding delays, flight cancellations, and weather conditions. Airport surface operations were already a complex socio technical system before the implementation of CDM. Does it decrease complexity or does it just change the face of it? (Mehta et al., 2013) (Bouarfa, Blom, Curran, & Everdij, 2013)

3.4.3 Multi-Modal Personal Urban Mobility Systems

The World Bank's 2014 World Urbanization Prospects states that urban population of the world has grown rapidly since 1950s: From 54% in 2014 urban population worldwide is projected to reach 66% in 2050. The fastest-growing agglomerations are projected to be medium-sized cities and cities with less than 1 million inhabitants located in Asia and Africa (United Nations Department of Economic and Social Affairs, 2014). Urban transportation systems in such cities face major challenges due to the continued growth of urban population, increasing private vehicle ownership, traffic congestion, and the fragility of public transportation systems. When

the urban transportation system experiences major difficulties, consequences affect households, businesses, and the urban community at large. Economic activity shapes and is shaped by the transportation system (e.g. businesses concentrate in congested city centers and residential areas in suburbs). It can become a constraint on social development and inclusion by increasing land and housing prices and disparities. Furthermore, increased traffic causes negative impacts on health and on the environment – air quality, noise – that affect the urban community's wellness (Dodder, 2006). Multi-modal urban mobility systems' core concept is to interoperate public transport with other motorized and non-motorized transport systems, as well as with new concepts of vehicle ownership. It involves the use of innovative technologies such as small size electric vehicles, smart phone and mobile applications (*Key to Innovation Integrated Solution: Multimodal Personal Mobility*, 2013). Local governments usually play a regulatory role and fund shared infrastructure maintenance (roads, bike lanes). Oftentimes, private sector companies are involved in the operation and funding of transportation modes (e.g. Zipcar, hubway bikes in Boston) and associated land development projects. External factors influence the optimal system design. These factors are economic (e.g. gasoline price, electricity cost, taxation policy), technical (e.g. vehicle technology, mobile technologies and accessibility), physical (e.g. inherited urban topology, terrain), regulatory (e.g. land use policy, environmental constraints), political (e.g. political will, taxation/incentive policy, public-private partnerships), and social (e.g. cultural habits of inhabitants, popularity of private vehicle use versus bicycle use versus transit use, population average age, wealth of people) (Spickermann, Grienitz, & Heiko, 2014; *Study on Urban Transport Development*, 2000).

Individual travelers are the ultimate beneficiaries of personal urban mobility systems. Designing user-centric transportation architectures is a technical issue (e.g. designing and optimally locating intermodal hubs, pricing services, selecting technologies, business planning). The architectural goodness of a transportation system is not determined by its technical structure alone, but also by the behavior of travelers, which in turn depend on individual decisions and actions. According to social sciences, well-designed environments (i.e. technical system and ecosystem) can strongly influence what people think and do. There is a continuous, dynamic interaction between a person's behavior and its environment. Hence engineering persuasive interventions on the environment allows altering human behavior at scale. Sticks (e.g. congestion pricing, car ownership taxes…) and carrots (e.g. low transit fares, tax rebates for transit/bicycle

commuters…) are traditional, yet short-lived, unsustainable policy levers for fueling motivation. Sustainable motivation is fueled by favorable engineered environments together with social influence (Stibe, 2015). Social influence is information: information that is sensed in one's immediate environment or conveyed as a field force by nowadays pervasive social, media and information networks. Social phenomena create information – e.g. real-time traffic information sharing through Waze and Google Maps – that affects individual behaviors, the aggregation of which create social phenomena.

3.5 Phenomena in the Example STS

All complex STS aren't complex for the same reasons. This section is an attempt at describing what flavors of complexity are prevalent in the examples above. Based on the above descriptions and the gatherings from literature, the following paragraphs dissect perceived complexity in the examples into technical, operational, human, socioeconomic and environmental phenomena. Neither case displays monochromatic phenomena. The order in which types of phenomena are described reflects their prevalence.

3.5.1 Joint Humanitarian Disaster Relief Logistics

A. Socioeconomic: organization structure; human resources allocation (drivers, pilots, ground handlers); conflicting interests in a joint response; leadership definition, clarity of each player's role and transparency of the implemented strategy between NGOs, UN agencies, the military and political authorities; equity in joint operations, performance evaluation
B. Operational: optimization of shipments in terms of stocks and flows of heterogeneous goods (in size, nature, handling requirements, origin, destination); optimal dynamic prioritization and allocation of vehicles and human resources to various shipments
C. Environmental: dynamic, heterogeneous needs i.e. demand; uncertainty in needs assessment; natural environment risks; risk of a shortage of fuel supply; local political barriers
D. Technical: network structure in terms of warehouses/hubs location and capacity; fleet portfolio characteristics (e.g. vehicle type, capacity, speed, terrain penetration capability)

3.5.2 Airport Collaborative Decision Making System

A. Operational: airport processes are dynamic coupled actions of distributed actors using shared, limited resources (runways, taxiways); traffic and turnaround dynamics on the surface network; pre-departure sequencing algorithm/automated system
B. Socioeconomic: roles definition within a collaborative context; decision delegation requires some equity guarantees and much transparency; economic performance results from traffic dynamics (e.g. delay costs, fuel costs)
C. Human: modified air traffic control work practices and displays
D. Technical: surface infrastructure layout, network of gates, taxiways and runways; integrated information system; CDM procedures
E. Environmental: weather conditions; traffic volume

3.5.3 Multi-Modal Personal Urban Mobility Systems

A. Human: itinerary decisions depend on driver preferences and value attributes, such as cost, time, walking distance, comfort, sensibility to environmental issues
B. Socioeconomic: social influence on individual decisions (e.g. social conformity, emulation); diffusion of new transportation modes (e.g. vehicle/bike sharing, electric vehicles); aggregate flows of individual movements (rush hour, stop and go, congestion) can be considered a social effect; real-time information diffusion (mobile apps, social networks)
C. Technical: transportation infrastructure and network; regulations (car taxation, congestion pricing, eco-friendly subsidies); characteristics of different transportation modes (e.g. cost, speed, comfort, congestion, schedules)
D. Environmental: fuel price, exogenous technological innovation, commuting population growth

3.6 Sense-Making of Complexity in STS

Illustration of the phenomena classification on the three example sociotechnical systems has exhibited some nuances behind the generic term “complexity”, and emphasized which phenomena need further understanding, and which are peripheral for sense-making of overall systems architecture.

Sense-making is the process of understanding and explicitly connecting the impact of current decisions and actions on future outcomes in order to select a best course of action. Long-lived, legacy, sociotechnical systems often have developed a complex architecture, which one wishes to change, radically or incrementally, but in a unprecedented direction. Systems architects sometimes have little guidance to glean from experience or data in making architectural choices for the future. Decision-making calls for a comparison of potential options to determine which, if any, best achieves system requirements. This requires an *understanding* of the options, an *assessment* of the range of *outcomes* that might be expected under each option, and an ability to *compare* them and select one when multiple *objectives* must be met. Finally, since the future is unknowable, it is necessary to make assumptions about *future* contexts.

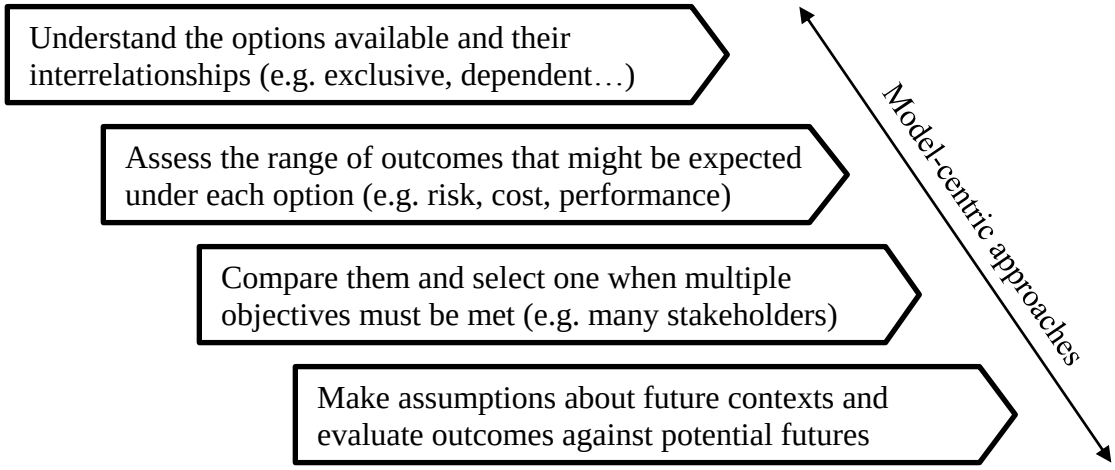


Figure 3-3: Model-centric approaches for Sense-Making

The following chapter examines capabilities of different modeling approaches for sense-making of complex STS (Figure 3-3). Given the nuances exhibited in the previous examples, it is safe to assume that a universal modeling approach could hardly apply effectively to all complex STS. Different depths of understanding - hence of modeling - will be required for different phenomena - technical and social, prevalent and peripheral, static and dynamic… - in order for model users to glean a holistic understanding of system architecture.

4 Model-centric Approaches Used to Study cSTS

"Beware of the analyst who proposes to model an entire social or economic system rather than a problem." (Sterman, 1991)

4.1 Conceptual and Formal Models

Why do we need models? At some point in the lives of most systems, there is a need to study them to try to gain insight into the relationships among various components, or to predict performance of a pre-existing system under some new conditions, or to assess the impact of design choices on performance of a future system (Bouarfa, 2015; Kelton & Law, 2000). Human capability for mental processing is limited. Complex systems architectures need simplified, meaningful representations to serve as communication platforms with stakeholders. In many cases, experimentation with the actual system is impossible because too costly or too disruptive, or because the system has yet to be built. For all these reasons it is necessary to build a model of the system and use it as a surrogate for the actual system, for specification, presentation, communication or analysis purposes.

A *model* is a representation of a system, entity, phenomenon, or process – the referent. An *conceptual* model is an overview of how the referent is "seen", such as a framework, a theory, or a paradigm (Van Hemel, MacMillan, & Zacharias, 2008). It may be implemented in different languages and formats (text, tables, charts and diagrams). It conveys key concepts to stakeholders. It can serve to explain soft phenomena (e.g. cultural drivers, diffusion of beliefs), but does not call for a particular formal modeling technique. Although it could describe analytics at a high level, it would not usually be comprehensive or detailed (e.g. utility theory underlying microeconomics). Conceptual models provide groundings, scope and consistency for developing detailed formal models, for assumptions and approximations that go into formal models, and they inform critique of results and model validation.

A *formal* model is a specification of behavior in formal semantics, such as logical, mathematical or computer language: input/output relationships, differential equations, logic decision tables, domain discretization on a grid, computer objects and their attributes, states and transition rules etc. A formal model's inputs and parameters can be manipulated to see how the model reacts (Davis & Anderson, 2003; Kelton & Law, 2000).

Among formal models, closed form mathematical relationships give exact analytical solutions. For example, distance equals the integral of velocity over time, and a simple pendulum's angular position is represented by a sinusoidal function. Analytical models can be more complicated than these simple examples and require computational support, such as a double pendulum's angular position (chaotic), or a multi-variable optimization problem (multiple local optima). These problems are made even more complicated if the problem statement is not time-invariant, for example if perturbations occur over time. "If an analytical solution is available and computationally efficient, it is usually desirable to study the model in this way rather than via a simulation. However, many systems are highly complex, so that valid models of them are themselves complex, precluding any possible analytical solution. In this case, the model must be studied by means of simulation" (Kelton & Law, 2000). Figure 4-1 depicts the different ways of experimenting with a sociotechnical system for sense-making.

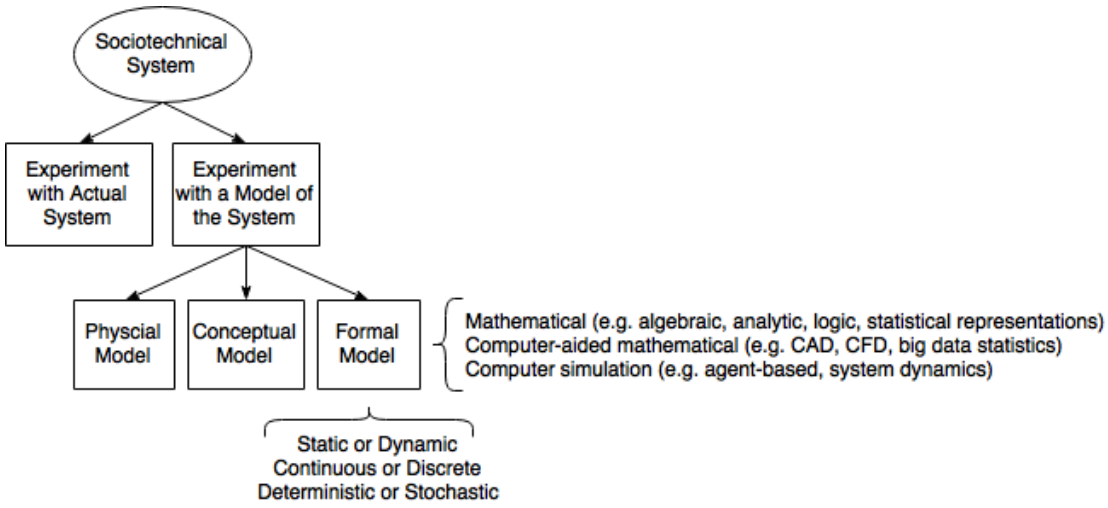


Figure 4-1: Ways to Study/Experiment with a System for Sense-Making, adapted from (Kelton & Law, 2000)**, p.4**

A simulation is an experimentation with a formal model in order to study *its representation of* the referent's behavior over time (Davis & Anderson, 2003). It consists in exercising the model for inputs and monitoring for their impact on outputs (Kelton & Law, 2000). An experimental frame is a set of conditions imposed on a given simulation run: input values, outputs to monitor,

internal parameters, hypotheses and boundary conditions. Simulations are used to *explore* the behavior of a system model over ranges of scenarios (themselves models of possible futures). The problem statement changes after each time step, which in turn affects the next time step.

Ideally, systems architects would have a dynamic model of the system at hand, would be able to reconfigure the model according to alternative systems architectures, simulate and study model behavior over time. With a valid architecture model, architecture choices can be informed with foresight in system behavior, and alternatives and tradeoffs can be explored. However, a comprehensive model of a complex STS would be difficult to build, unwieldy to use or explain without considerable expertise, let alone to maintain and modify if needed. "It is simply not possible to build a single, integrated model of the world, into which mathematical inputs can be inserted and out of which will flow a coherent and useful understanding of world trends" (Sterman, 1991).

4.2 Architecture Descriptions

Architecture frameworks provide standards for the description of architectures, including but not limited to technical systems architectures, e.g. the U.S. Department of Defense Architecture Framework (DoDAF). There is no one "best" architecture framework applicable to all sociotechnical systems. "By characterizing the form, function, and rules governing systems, architecture frameworks serve as a communication tool to stakeholder communities with different views of the system and facilitate comparative evaluation across architectures" (Maier, 2009; Richards, Shah, Hastings, & Rhodes, 2007). They often produce conceptual static descriptions, useful for communication and specification purposes. They provide the necessary level of abstraction and scoping to build a shared mental model of the system amongst stakeholders.

In designing a system's architecture it is critical to understand how architectural choices affect system behavior over time, lifecycle performance and cost, emergent dynamics, and other time-variant properties. This requires analytical or simulation capabilities. For example, a qualitative cause-effect map can conceptually convey the same causal mechanisms as a Bayesian network, but fail to render time-varying system response that latter would (e.g. attractor states, learning, oscillating dynamics). Additionally, systems architects need formal models to support the

analysis and tradeoff of systems architectures, efficiently and with some degree of objectivity. Without executable or computational capabilities, systems architects are limited in their capability to explore large architecture spaces and to compare choices consistently and objectively (Glazner, 2009).

Architecture semantics (e.g. UML, SysML, OPM) are often ill suited for execution: behavior diagrams are too qualitative for simulation, or limited to discrete time stepping, and finite states and flows (e.g. interaction diagrams, state machines, use case diagrams, activity diagrams). There are efforts at making these semantics executable to some extent (e.g. xUML) or at interfacing them with other executable semantics (e.g. Modelica) to leverage respective advantages and mitigate shortcomings.

It is apparent that conceptual architecture descriptions are necessary but need to be complemented with quantitative and simulation capabilities. Different formal modeling and simulation methods have different capabilities and limitations. The selection of appropriate modeling approaches proceeds from the purpose of modeling. Characteristics of modeling methods and characteristics of system complexity are additional criteria for navigating among the plethora of existing modeling methods.

4.3 Model Selection and Composition

4.3.1 Model Purpose

The selection of an appropriate modeling approach is determined by the question or problem being addressed. It is recognized that a model is not a full representation of the system. A model should be a useful representation, and draws its goodness from being simple, yet evocative (Bankes, 1993). Therefore, the purpose for modeling sets the boundary for model depth and breadth.

Various types of models have been enumerated by the systems community, but there appears to be insufficient attention given to model purpose itself. All models are wrong in the sense that they omit some details. Different modeling formalisms differ the goals/questions they prioritize and their capacity and ease of use for addressing specific types of questions. In the field of informatics, McBurney (2012) proposed nine model purposes:

1. To understand, predict or control natural reality.
2. To understand or predict an existing human phenomenon or system.
3. To understand, predict or control future human phenomena or artificial systems in design or development phase.
4. To serve as a locus for discussion between stakeholders, to enable alternative exploration in a structured and shared way.
5. To identify, articulate and potentially resolve trade-offs, action options, and their consequences.
6. To enable rigorous, structured and justified thinking about assumptions and their relationships to one another.
7. To train people in expedited and focused experiences of reality.
8. To enable stakeholders to learn about and assess the assumptions, reasoning process and action plans of the modelers.
9. To play, to enable the exercise of human intelligence, ingenuity and creativity, in developing and exploring the model.

Maier (2009) proposes six model roles in systems architecting that are especially relevant for technical systems:

1. Communication with client, users, and builders.
2. Maintenance of system integrity through coordination of design activities.
3. Assisting design by providing templates, and organizing and recording decisions.
4. Exploration and manipulation of solution parameters and characteristics, guiding and recording aggregation and decomposition of system functions, components, and objects.
5. Performance prediction and identification of critical system elements.
6. Providing acceptance criteria for certification for use.

When tackling the complexity of a *technical* system such as an aircraft, modelers have the intent to classify, to explain, to predict, to control, to detect or to diagnose the system input/output behavior. As for *human* behavioral and *social* phenomena, different modeling approaches are appropriate at the people, process, organization and ecosystem scales (Rouse, 2015). For example, at both ends of the spectrum, models at the people level will seek to detect, to explain and to resolve anomalies in human behaviors (e.g. consumer behavior), while models at the ecosystem level will seek to detect, to explain and to resolve anomalies in societal behaviors (e.g.

intra-firm competitive relations). Lower-level model outputs serve as inputs for higher-level models and higher-level outputs are constraints on lower levels of modeling.

Zacharias et al. (2008) point out that individual, organizational and societal models, do not predict exactly what humans will do, as individuals or in groups, but rather help forecast a range of potential action outcomes, draw attention to potential unintended consequences, and highlight variables that are overlooked in a particular situation. Accordingly, model purposes include: to analyze fragmented information and develop courses of action based on the likelihood of desired outcomes; to train personnel, simulating the environment, dynamics and providing performance feedback; and to design and evaluate a technical system, predict its performance and make decisions based on cost-benefit tradeoffs.

A shared understanding of model purposes would facilitate the critical transition from the motivation for modeling (e.g. perceived complexity flavors/types in a system), to the choice of an adequate set of models. Empirical observations suggest the manner in which this is performed today is mostly ad hoc, or biased by practice anchoring within a field or a group, for example being blind to alternative description languages, alternative perspectives, and unprecedented types of systems (e.g. social) due to familiarity and/or success with models, languages and systems of a particular type (Maier, 2009).

4.3.2 Model Characteristics

The systems community has instantiated many different models to support designing, evaluating, and testing technical systems (requirements/functional diagram, computer-aided design, design structure matrix, multi-attribute trade space exploration, cross impact analysis, fault tree analysis, hardware-in-the-loop/human-in-the-loop simulation…), analyzing social phenomena (system dynamics, network models, queuing models, agent based models, graph theory models…), and understanding human behavior (mental models, Petri net models, task models, microeconomics models…) among other purposes.

A suggested first step in making the transition from the motivation for, to the selection of, a model would be to examine one's issue and expectations from the model against previous instances. In this endeavor, a modeler would use a classification of previously adopted modeling approaches, as shown in Figure 4-2, with a profile of each model instance along several

characteristics, including model purpose. It is also expected that such a survey would highlight limitations of each modeling approach in its context of use, which are equally important as the purpose.

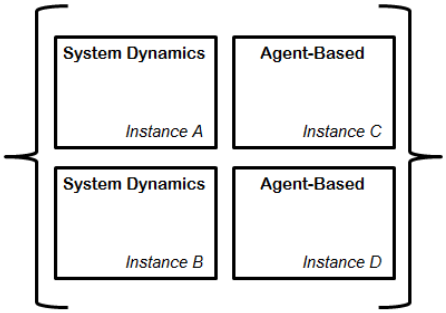


Figure 4-2: Example of classification of model approaches and model instantiations

Model characteristics would provide additional elaboration to the metadata that is needed to enable case-specific selection of models. Table 4-1 provides a – non exhaustive - list of model characteristics examples. Model anatomy characteristics describe model build and use: what inputs humans interacting with the model "dial in", what data must be available to interact with the model, what assumptions are made, how time is modeled, how uncertainty is accounted for etc. Model anatomy characteristics reflect the methodology and logic of the model construction and interaction processes. Once a model is built, assumptions that went into its construction are easily overlooked. Keeping track of anatomy characteristics is a means to inform critique in the interpretation of models. Model referent characteristics proceed from the acknowledgment that a model necessarily leaves out some details of the real system. The second part of Table 4-1 delimits the scope of the model, i.e. what technical, process, human, economic, social and ecosystem aspects of the real system the model represents. It can also be useful to keep track of what aspects are left out of the model.

Table 4-1: Model Characteristics: Model Anatomy and Model Referent

Model Anatomy Characteristics	*Outputs*	Conceptual or formal, Predictive (e.g. grounded in science) or exploratory (e.g. scenarios) Format (visualization, analysis, synthesis)
	Inputs	Data: format and volume required or available (e.g. historical data) Parameterization: initial conditions, boundary conditions
	Tool	Computer environment, language, software
	Assumptions	Variables definitions and approximations (e.g. steady or time-dependent, bounded or not, discrete or continuous, exogenous or endogenous) Logic: linear or non-linear, deterministic or stochastic, Markov or with memory
	Simulation	Y/N Scenario definition elements Hybrid human-in-the-loop or simple computer simulation
	Time	Static or dynamic Dynamic: continuous or discrete Time horizon
	Uncertainty	Stochastic variables Probability distributions: justification, parameters

Model Referent Characteristics	*Human*	Autonomous agents, individuals Human system interfaces and other channels, tools of interaction Internal processes, mental models, learning, gaming, decision-making…
	Social	Spatial movements, flows and diffusion phenomena Social and information networks Codes, systems of values Organizations, roles allocation, hierarchies
	Economic	Performance, cost evaluation Interests, utility, attributes Inter-stakeholder relationships, network, competition, negotiation…
	Process	Human-System interactions, tasks Automated processes Technical system behavior: failure modes, performance, states, transitions, feedback loops, control mechanisms
	Technical	Component, subsystems, or systems level description of technical system Physical, functional or other decomposition logic Hardware, software, infrastructure, procedures Technical requirements
	Ecosystem	Fluxes: agents, interfaced technical systems, information, energy Field forces: economic, political, regulatory, natural, social influences Evolution of the ecosystem: scenarios, possible futures

Social, human and technical elements of the referent system call for different modeling approaches. For example, human and social models range from qualitative to quantitative, from top-down to bottom-up logics, from descriptive to predictive, and from static to dynamic. Van Hemel et al. (2008) assessed similarities between, "kinds of models" and identified clusters of types of models (Figure 4-3). The clustering among modeling techniques highlights the importance of scale in model classification: macro (society, organizations, networks, systems…), meso (teams, groups, neighbors, processes, performance…) and micro scales (individual actors, cognition, technical components, subsystems…).

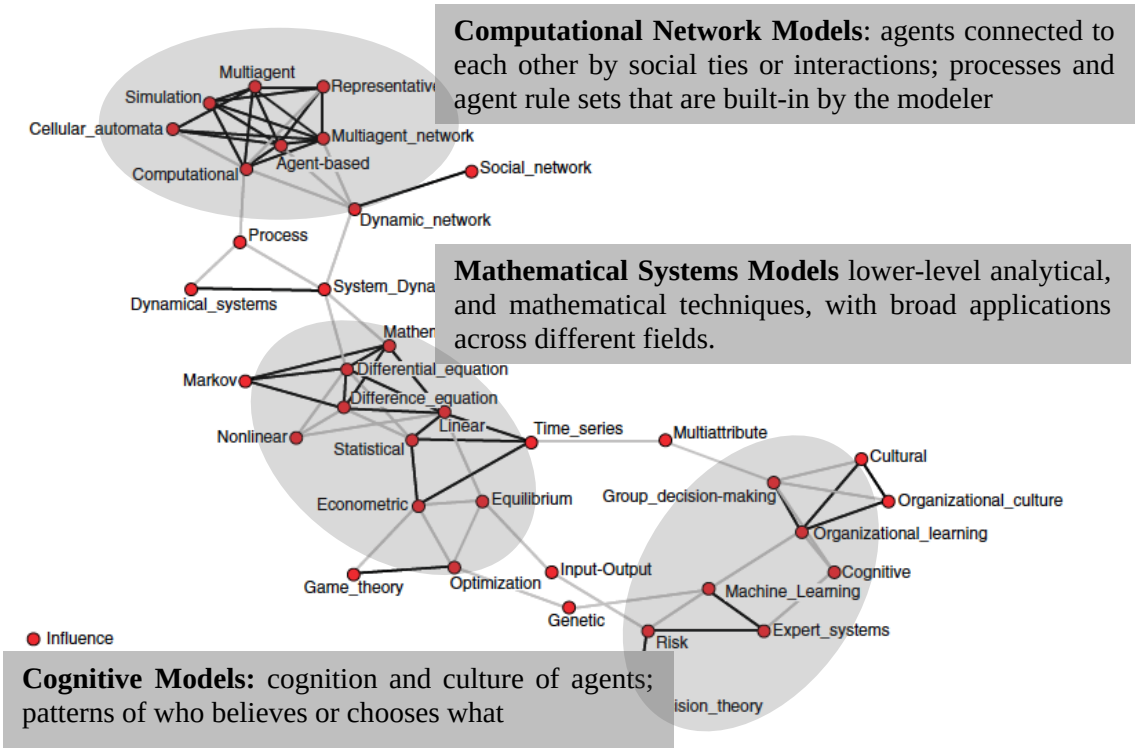


Figure 4-3: Perceived Similarities Among Types of Models, adapted from (Van Hemel et al., 2008); links show similarity

Similarly Rouse (2015) proposes a classification (see chapter 2), that distinguishes between models of human, process, economic and social phenomena, stressing again the importance of scale (Rouse, 2015). Acknowledging referent scale as an important model characteristic, Table

4-2 lists examples of social referents (objects to be modeled) from societal to individual scale, and provides examples of modeling approaches for each scale. The individual scale is easy to represent oneself and identify. Higher scales span a continuum that this thesis does not attempt to classify, but that (generally speaking) go from groups/teams to organizations/enterprises to societies (see Table 2-1 for example).

Table 4-2: Multiple Scales of Human/Social Models

Example Model Referents	Example Models
Diffusion of epidemics	System dynamics model
Macroeconomic policy	Economic theory models, system dynamics model
Crowd dynamics	Agent-based models, network models
Group decision-making,	Analytic hierarchical process, multi-attribute utility models
Work processes and procedures, resource allocation	Queuing models, cognitive and task-performance models
Supervisory control, human-machine interaction	Dynamic Bayesian networks, servo-mechanisms models, estimation theory models
Individual decision-making, under risk, uncertainty, or multiple objectives	Decision theory models, game theory models, utility models
Cognitive/affective processing; e.g. attention allocation, learning, information processing	Cognitive models, task performance models, mental models

4.3.3 Model Selection

Traditional (technical) systems engineering models are insufficient for studying sociotechnical systems, which have interwoven social and technical systems. Computational methods developed to address hard, technical questions aren't geared for addressing cognitive and soft issues, organization/enterprise problems, unknown unknowns, and emergence. In contrast with consolidative computational modeling of deterministic systems (models as surrogates for real systems), Bankes (1993) espouses exploratory modeling of systems, which tend to be plagued with uncertainty and unknown unknowns (models as means of testing hypotheses and exploring ranges of possible outcomes). However, it is argued that exploratory modeling can only produce

useful results through a constellation of alternative models. Instead of expending effort to find the "right" model, leveraging multiple different models and comparing their results can support cross-validation of each model and increase decision maker confidence in their results. Ross et al. demonstrate this model-trading concept on evaluative models (e.g. capability, cost, performance models) underlying trade space generation (Ross, Fitzgerald, & Rhodes, 2016). By using multiple simple models, complexity is exported outside the models to the ensemble of model outcomes, from which modelers and stakeholders must make sense (Bankes, 1993).

Selecting one or several models for a complex case study requires breaking down the problem into scoped questions such that each one translates into a model purpose. For example, designing an urban smart power grid is a complex problem taken as a whole. It could be broken down into the following scoped problems: (1) Trade alternative energy production mixes, predict reliability and vulnerability of each alternative against dynamic demand loads (technical level). (2) Predict consumer preferences regarding energy bills, effort involved, and value attached to contributing to environmental sustainability (human level). (3) Optimize dynamic pricing of utilities – organization level - within the rules set by local government – environment. Rouse (2015) proposes such a multi-level modeling approach for dealing with complex socio technical systems. Each problem would then call for a tailored modeling approach: (1a) Trade space exploration, (1b) Fault tree analysis. (2) Multi-attribute utility functions. (3) Microeconomics model of key stakeholders. This model set (1a, 1b, 2 and 3) is neither unique, nor exhaustive and a modeler might only be interested in one aspect of the problem. Therefore, identifying the problem clearly and formulating a clear model *purpose* is essential in setting the boundaries for the modeling effort.

4.3.4 Model Composition

Large models are usually best designed to be *modular*, i.e. to have component models that can be independently developed, that are seen by the rest of the model as building blocks that can be interacted with only through the inputs and outputs of a well-defined interface such as ports. A component model may be quite complex internally but still have a simple interface and be suitable for reuse. Composing models consists in selecting and assembling models. For modeling complex systems, modularity is desirable:

- It breaks down the modeling task into smaller problems, allowing for model specialization.
- Decomposition of the problem helps comprehend the different facets of a complex system.
- Testing is simplified if done modularly at first.
- Previously validated component models can be reused, allowing for development cost and time savings.
- Maintaining and modifying models is less risky when interfaces are well specified.

Model reuse is a relatively new idea and there is no standard *modus operandi* for composing modules. Some issues in composing models are potentially entangled states of models, and consistency of assumptions in inputs, outputs, parameters, logic and process (Rouse & Bodner, 2013). Furthermore, there lacks a curated repository of previously developed and used models that can be tapped.

Recent work on hybrid modeling (La Tour & Hastings, 2015; Mathieu, James, et al., 2007; Mathieu, Melhuish, et al., 2007; Zulkepli, Eldabi, & Mustafee, 2012) and multi-scale modeling (Rouse, 2015) point towards the usefulness of using not a single but a set of models to study a complex system. La Tour (2015) implements a system dynamics model interactively with a trade space exploration model for investigating the impact of time on the lifecycle and procurement of GPS satellites. Mathieu et al. (2007) implement a system dynamics model, a Petri net process model and an agent-based model for simulating the response timeliness and effectiveness of the Air and Space Operations Center to a series of critical events, at the mission, process and operator levels. Zulkepli (2012) implements a system dynamics model and a discrete-event simulation to simulate the interactions between healthcare personnel stress, and patient non-recovery and readmission rates. Synergies between models are seen to increase modeling capabilities and insights into problem solving while reducing the limitations of individual techniques.

Efforts at integrating architecture framework approaches and simulation capabilities also point in the direction of multi-method modeling. Richards et al. (2007) used a software tool (CORE) to track the availability of Hubble telescope exposed to wear of components. CORE bridges the

requirements, functional, and physical views of the architecture with a discrete event simulation. A discrete event simulation executes functional flow block diagrams over time. Physical component failure is described probabilistically. Two different servicing architectures are designed and their success criteria described probabilistically. A Monte Carlo simulation is conducted to estimate the availability of Hubble across multiple servicing campaigns over time. This hybridization of models allowed the authors to explore functional impacts of servicing architecture choices. (Richards et al., 2007)

In the field of enterprise architecting, Glazner (2009) proposes a hybrid approach for enterprise modeling. It consists in (1) hybridizing simulation techniques to connect inputs and outputs of sub-model simulations, each corresponding to a specific view of the enterprise, and (2) using an enterprise AF to provide a grounded abstraction of the enterprise, to bound sub-model views, to preserve overall consistency, and to define connections between simulations.

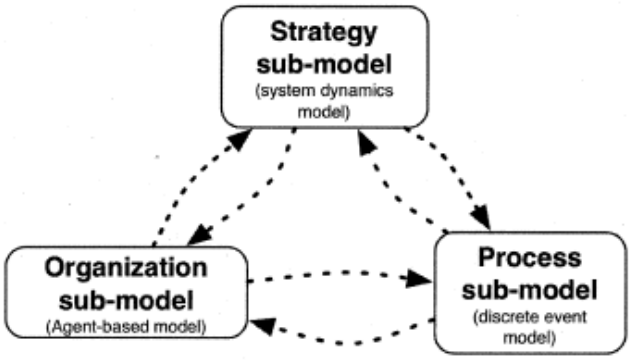


Figure 4-4: Notional Representation of Hybrid Enterprise Architecture-Based Simulation Model, from (Glazner, 2009)

"Enterprise architects and managers alike can understand and relate to […] resulting interaction […] from an organization or enterprise-level strategic systems perspective […] by conducting simulation experiments to help define the future state enterprise architecture options as part of the enterprise architecting process" (Glazner, 2009). Making sense of the overarching systems architecting problem requires integrating sub-model outputs. This can be automated (e.g. computational models) or not. There may be conceptual consistency issues for example with

assumptions, and variable definition (e.g. endogenous vs. exogenous). With hybrid computational models, further issues arise. Hybrid modeling requires more validation effort, since the models must first be tested individually and then together. However, the risk in aggregating many models is to end up with a model of high resolution but low practical utility and transparency (Bankes, 1993).

When the output of one model serves as the input for another (model chaining, Figure 4-5A), the link between the two models may be straightforward (e.g. sequential, passing a variable, triggering an event…), involve a transformation (e.g. translating from one modeling language to another, from hard to soft variables, converting units…) or require the intervention of the modeler. In a different configuration, two models are run in parallel and the modeler confronts and analyzes outputs (Figure 4-5B). The confrontation of results can be straightforward (e.g. variables plotted against each other on a graph), involve some transformation (e.g. of units) or require a more interactive, evocative visualization. If a model provides unexpected outputs, feedback allows reviewing assumptions and modifying inputs, in a goal-seeking manner (Figure 4-5C). Such feedback could be automated (e.g. parameter fitting to data) or require human intervention (e.g. hypothesis testing). Combinations of the above can be constructed, increasing the overall modeling effort and complexity.

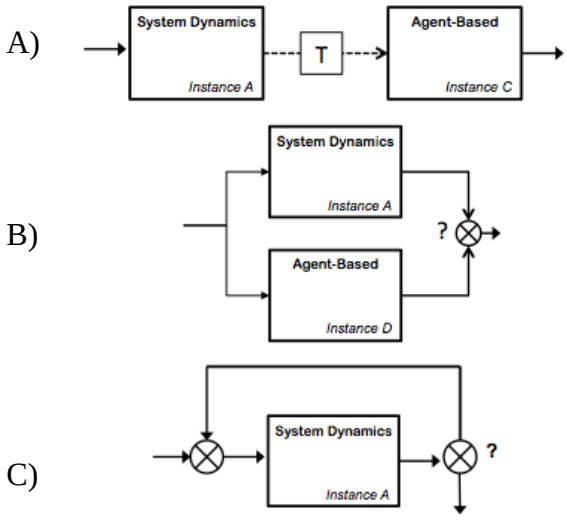


Figure 4-5: Possible Compositions of Models

4.4 Exploratory Systems Architecting

As emphasized earlier, sense-making is the "holy grail" for systems architects, i.e. being able to infer the impact of architectural choices on future system behavior, to inform architectural design with foresight and explore tradeoffs at an early stage when changes can be brought to the architecture at lower cost and effort.

Simulation modeling has become a very popular approach used by disciplines ranging from manufacturing floor planning to social sciences for several decades. Simulation modeling allows users to systematically investigate complex processes and behaviors in systems that do not lend themselves well to experimentation on the real system itself (impossible, impractical, costly…), or to traditional, closed-form mathematical analysis. Simulation models can be relatively quick to develop, cost effective and flexible, or time-consuming and difficult to modify once built. There are three major classes of methodologies used for dynamic simulation models: System Dynamics (SD), Discrete Event (DE), and Agent Based (AB) (Figure 4-6, Figure 4-7).

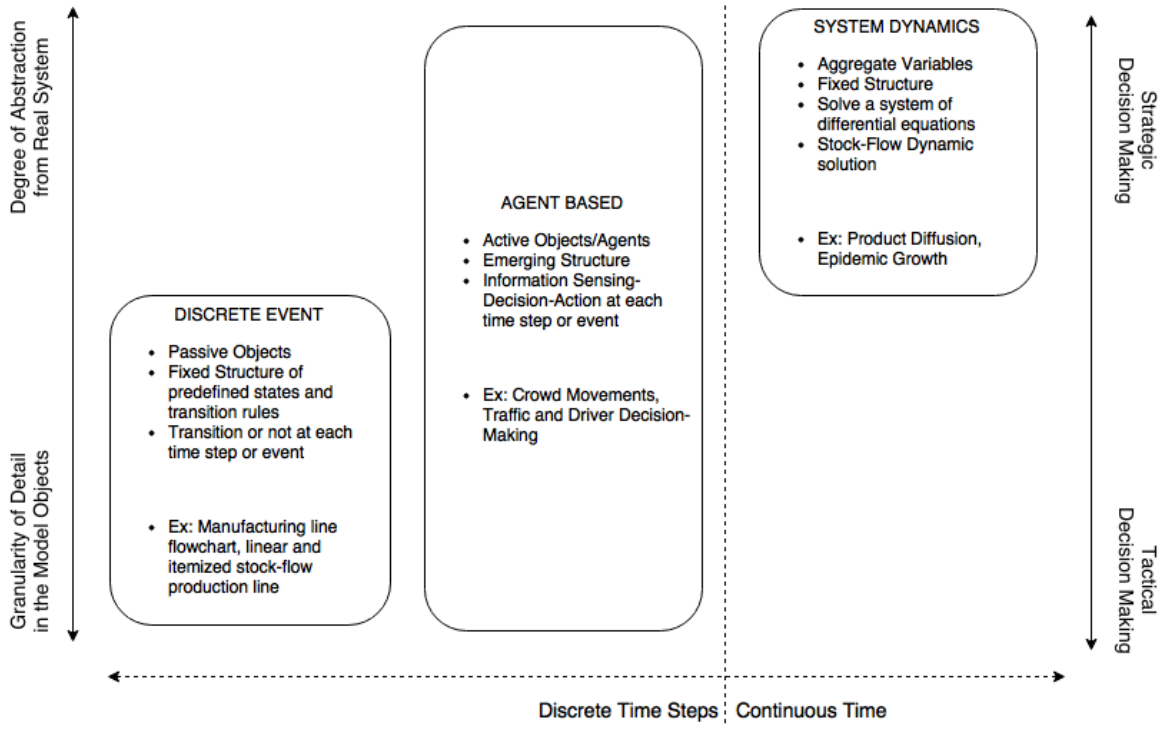


Figure 4-6: Approaches in Simulation Modeling on Abstraction Level Scale, adapted from (Bouarfa, 2015), p.80

The first two employ a system-level (top-down) model definition while the agent-based approach is a bottom-up approach where the model is built at the agent level and system-level structure emerges. SD assumes a high abstraction level and chiefly targeted at strategic level problems. DE modeling is mainly used on operational and tactical levels. AB models are used at all levels: agents can be individuals, competing companies, groups, vehicles, pedestrians, information systems, algorithms or robots, work processes, procedures, or infrastructure (Figure 4-6). Such considerations can help identify a good simulation methodology. Alternatively Marshall et al. (2015) suggest selecting an appropriate simulation method among AB, DE and SD based on model purpose criteria and characteristics of the object being modeled (Figure 4-7).

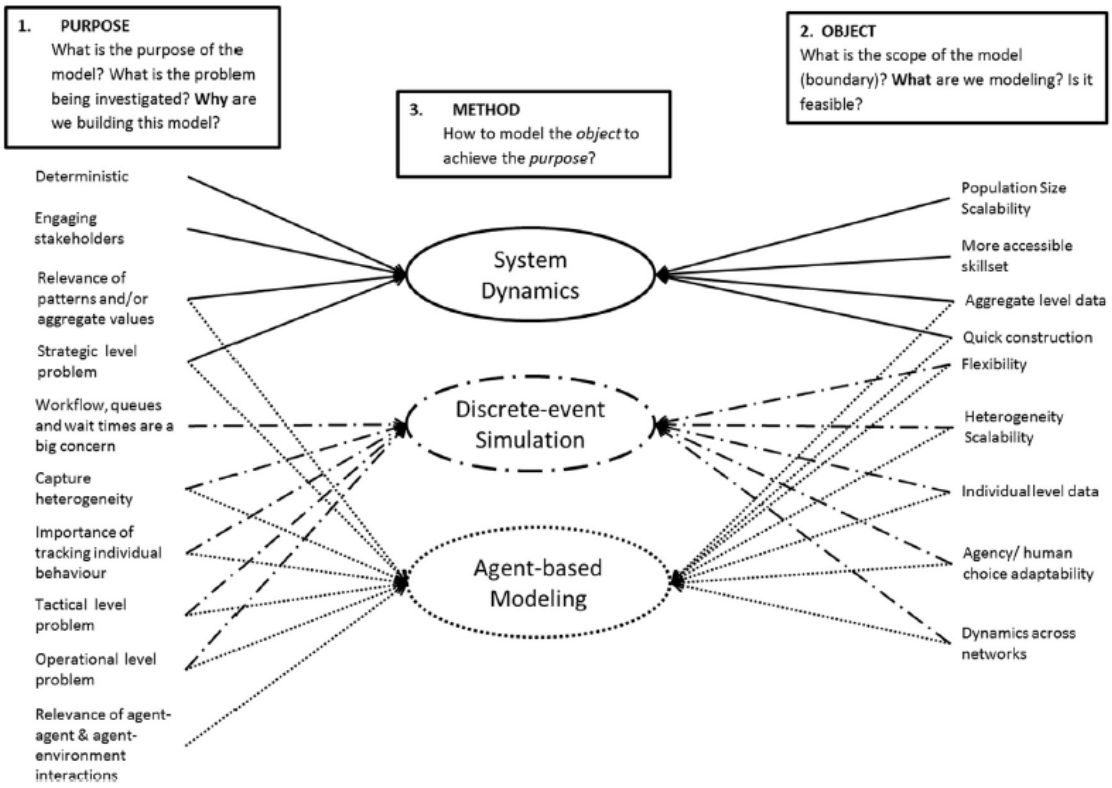


Figure 4-7: High-level summary of criteria for selecting a dynamic simulation modeling method from (Marshall et al, 2015), p.153

4.5 Modeling for Sense-Making of Complex STS Architecture

How can models facilitate sense-making and better inform architectural decisions about complex systems?

Sense-making is the process of understanding and explicitly connecting the impact of current decisions and actions on future outcomes in order to select a best course of action. It is a "motivated, continuous effort to understand connections (which can be among people, places and events) in order to anticipate their trajectories and act effectively" (Klein, Moon, & Hoffman, 2006). Sense-making of complex systems architecture calls for models to help *understand* the architecture alternatives, *assess* the range of *outcomes* that might emerge under each alternative, *compare* architectures and select one when multiple *objectives* must be met. It is necessary to make assumptions about *future contexts*, to assess the validity of these assessments against uncertainty.

Assessing the range of *outcomes* that might emerge under alternative architectures requires formal modeling. When these outcomes are distant in time, futures models representing assumptions about context evolution and *dynamic modeling capabilities* are needed. Initially however, the architecture of a complex system (structure and behavior) is not understood or observed in formal/mathematical terms. "An early focus on quantification may preclude analysts from thinking about important variables that are not easily measured or modeled" (Dodder, 2006). Instead, *understanding* the available options calls for qualitative representations of system architecture. Qualitative/conceptual models establish the boundaries, purpose, inputs/outputs and assumptions for more detailed, formal models to be developed (Rouse, 2015; Sterman, 2002; Sussman et al., 2007). *Comparing* architectures requires a visualization capability for interpreting model results and *comparing* alternative architectures, in terms of evaluative or value models.

Architecture frameworks are widespread frameworks for producing conceptual systems architecture models, but they are not unique nor necessarily best suited to sense-making. Architecture frameworks produce standard, more or less exhaustive decompositions of systems. Sense-making is more than a decomposition/integration, or "connecting the dots", approach. "[Connecting the dots] misses the skill needed to identify what counts as a dot in the first place.

Of course relating dots is critical, but the analyst must also determine which dots are transient signals and which are false signals that should be ignored" (Klein et al., 2006). Rouse (2015) suggests reading complex STS through a phenomena-based framework. Establishing relationships between phenomena (e.g. causal, temporal, functional, logical relationships) indicates which are critical for systems architecting, and how they produce system-level structure and behavior. A single model may capture one or more phenomena, at a given scale, over a given time horizon, but cannot cover all with equal depth. It is not even certain that all phenomena call for the same depth of understanding, depending in the purpose of the modeling endeavor.

This paragraph draws an analogy with aerodynamic modeling to illustrate the idea that different purposes call for different depths of understanding of different phenomena. Aerodynamic modeling is used to shape bodies such as airfoils. Fluid dynamics is modeled by the Navier-Stokes equations, under fluid continuity and Newtonian assumptions. If the purpose is to grossly estimate lift, the flow far away from an airfoil may be modeled as potential flow, but modeling the entire flow field as such leads to the D'Alembert paradox: computed airfoil drag is zero. The potential flow model doesn't render the viscous boundary layer on the body surface. One should change the model near the airfoil if the purpose is to compute drag. Interactive boundary layer methods combine potential and viscous flow models, respectively "far" from the airfoil and "close" to it. Such models can relate airfoil drag-to-lift ratio to airfoil shape. Furthermore, if one wanted to understand the effect of surface roughness, one would have to change the model again and refine the grid. Turbulent equation models require more computational power and modeling effort. The value of additional reaps from more detailed models ought to be weighted against the additional effort and time required, bearing in mind the purpose of modeling. The more assumptions go into the system model, the more caution should be taken in interpreting its results.

Models should be simple where possible and refined where critical trade-offs are identified. The first step therefore consists in high-level identification of the problem and key phenomena hypothesized to be relevant, without necessarily putting them into algebraic form yet. Delaying formal modeling until later is useful when dealing with soft-hard problems, such as those encountered in complex sociotechnical systems. The framework proposed is dual: On one hand it proposes to use a phenomena-based architecture description for problem identification and

framing, and on the other hand to select and compose models of phenomena modularly, accordingly with the phenomena-based architecture description (e.g. diagram). Systems architects can delve into formal modeling where needed (on a phenomenon, on the relationship between two phenomena), while referring to the phenomena-based architecture description for system-level sense-making and results interpretation against "the big picture". Rouse (2015) summarizes and compares potential modeling approaches suitable for the different classes of phenomena that he proposes, as well as connections between them (Figure 3-1). The framework proposed in this thesis re-uses the notion that model modularity can be achieved at the phenomena level, i.e. that an architecture modeled in terms of phenomena can be explored in a "plug and play" manner, i.e. that leveraging libraries of phenomena models allows to select, compose and trade modeling strategies (Figure 4-8). The recursive experimentation with models and interpretation at the architecture level is conducive to sense-making.

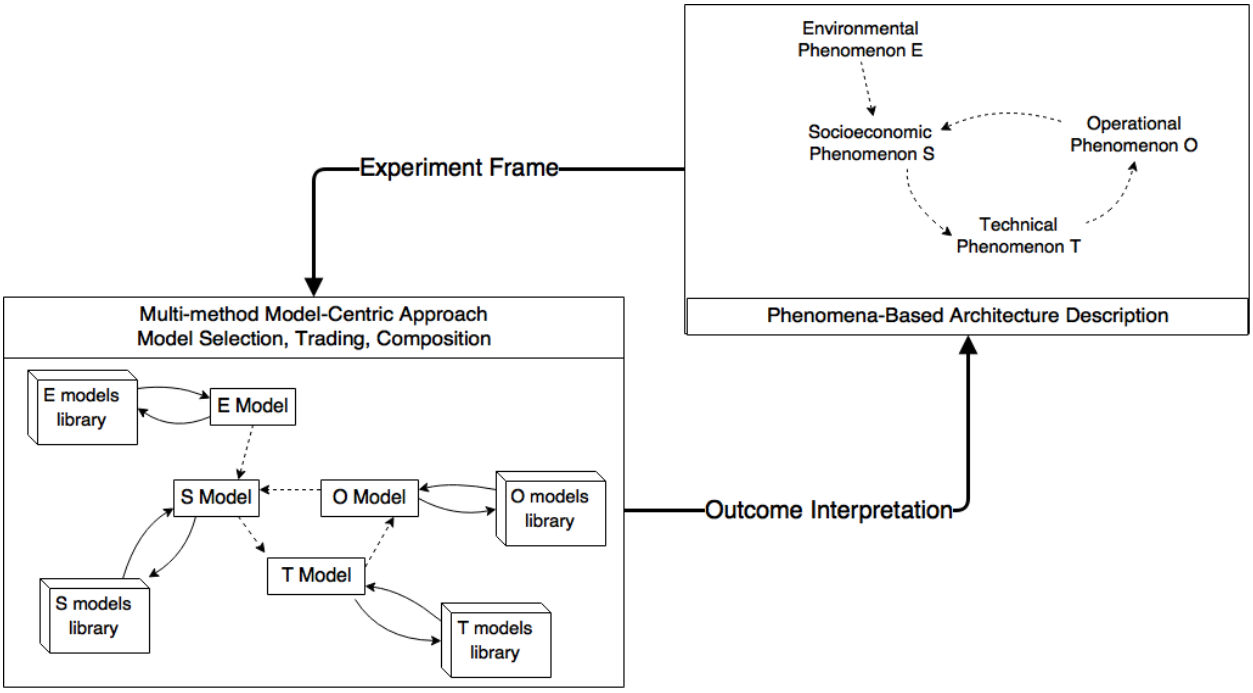


Figure 4-8: Sense-Making Framework - Application and Use Example

5 Application to Collaborative Decision Making System at Paris Charles de Gaulle Airport

5.1 Context

In the 1990s, Paris Charles de Gaulle airport (CDG) experienced a rapid traffic growth: Air France and FedEx hubs opened, and aircraft movements increased by 6.9% between 1990-1995 and by 9.3% between 1995-2000 (see appendix). Infrastructure and work practices adapted incrementally and without coordination between aircraft operators, the airport and the aviation administration. In the winter of 2003, heavy snowfalls disrupted operations at CDG many times throughout the season, resulting in 25% of flight cancellations, over 2 hours of delay per flight, huge traffic jams on the ground, 5000 passengers stranded in terminals and as many in hotels. Poor visibility into the evolution of the situation led to persisting difficulties and a five-day long recovery. Both air traffic control services of *Direction des Services de la Navigation Aérienne* (DSNA) and the airport operator *Aéroports de Paris* (ADP) were overwhelmed by the events. Media and political coverage was important and spurred an initiative to improve handling of adverse conditions.

The year 2003 was also the year when the proportion of bottlenecks in European airports exceeded the proportion of bottlenecks en-route (during cruise). This resulted from increasing traffic demand, significantly increased en route capacity, slowly increasing airport capacity and disjoint airport and en-route traffic optimization. Given a projected threefold increase of European traffic between 2003 and 2025, EUROCONTROL, the European traffic flow manager, set objectives to increase airport capacities by a factor of three, to cut air traffic management costs in half, to increase safety, and to reduce environmental impacts of aviation by 2025. The Airport Collaborative Decision Making (CDM) initiative is one of several initiatives to achieve these goals. CDM aims at bringing pragmatic solutions to optimize the use of airport capacity and better feed the network manager with airport data. A few airports showed immediate interest and trials started at Zürich, Munich and Brussels airports.

In 2004, DSNA, ADP and Air France co-launched the CDM@CDG project to deploy the concept by 2010 to enable the optimization of operations thanks to accurate and timely information sharing, adequate procedures and tools. The cornerstone of CDM is to improve common situational awareness amongst airport partners. Therefrom follow several benefits: improved flight predictability, regularity, and punctuality; improved airport surface and air traffic fluidity; the opportunity to tactically optimize use of airport resources; and closer adherence to EUROCONTROL departure slots. Constraints on achieving these objectives are that the transformations should maintain high levels of safety and limit the environmental impact of airport operations. Furthermore, achieving effective collaboration implies a significant culture change to traditionally competitive *modus operandi*.

A situation room, the CDM Cell, was installed at CDG, where representatives of DSNA, ADP, Meteo France[6], and Air France could manage operations in adverse conditions together (e.g. snow, low visibility). Co-locating stakeholders in a room enables making better tactical decisions faster. However, this preliminary initiative did not introduce new processes or tools, stakeholders did not dedicate personnel, set up a project organization or define objectives and performance indicators for CDM to achieve. Consequently, the winter of 2005 brought about new difficulties.

[6] French weather forecasts provider

In 2007 the CDM@CDG project was reinforced with dedicated staff, organization and resources, to develop tools and procedures for sharing information and re-distributing tactical decision-making, that would benefit all CDM partners, as well as passengers and the environment (Figure 5-1). This example of a failed partial architectural solution (only implementing the CDM Cell) demonstrates the necessity of a joint social *and* technical architecting effort on sociotechnical systems.

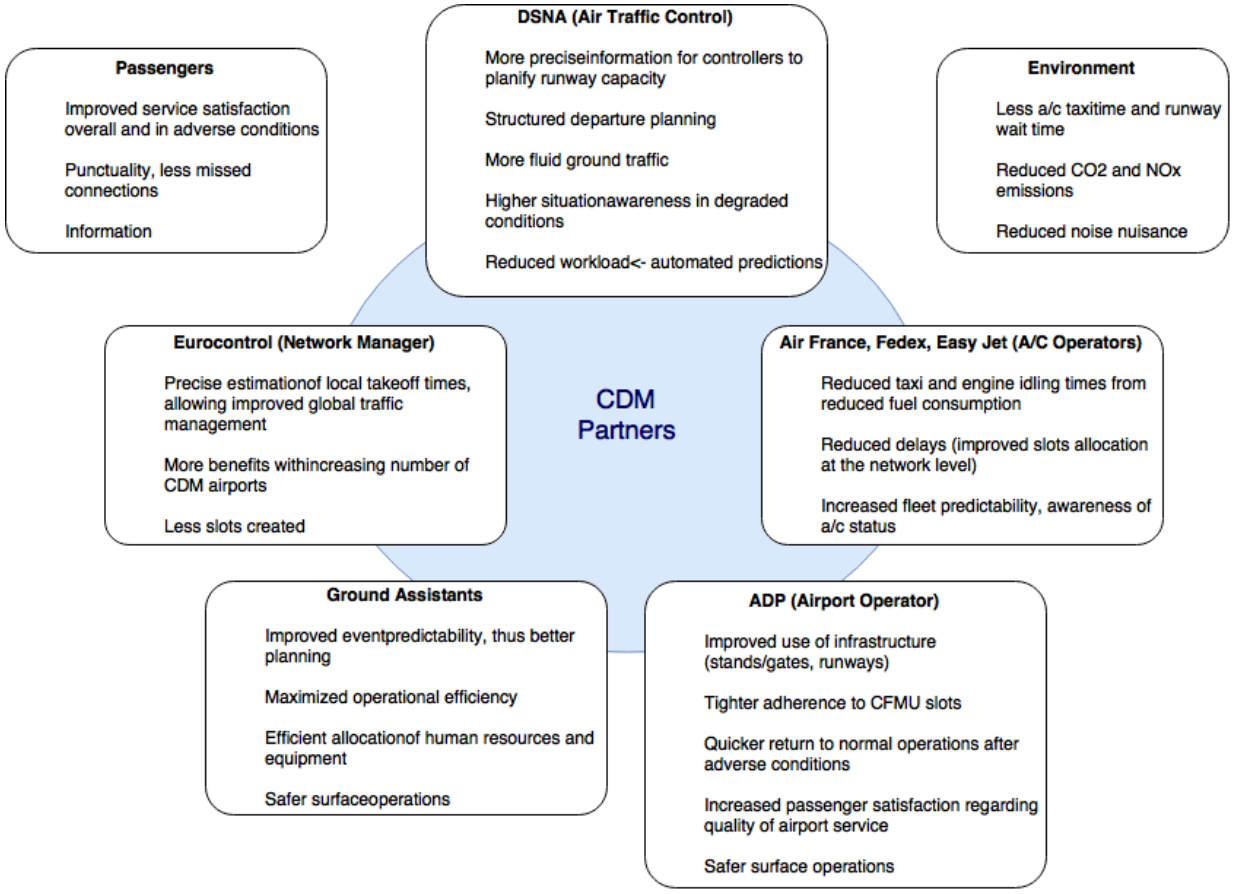


Figure 5-1: Benefits to CDM Partners[7]

[7] Passengers and the environment also benefit from CDM albeit indirectly. Ground assistants (e.g. catering, cleaning) interact with the CDM system, through the aircraft operator sub-contracting them. These stakeholders are not further considered.

CDG was awarded the EUROCONTROL Advanced-CDM airport label in November 2010. Improvements with CDM are significant: 2.5 minutes less taxi time for departures, approximately 4,000 tons of fuel and 12,000 tons of CO_2 emissions saved yearly. The following sections investigate the complexity of CDM at CDG. Phenomena of interest are discussed and modeling approaches are suggested that could help sense-making for CDM architecting.

5.2 CDM at CDG: Phenomena

Turnaround and departure activities manifest emergent inefficiencies, such as takeoff delays, congested taxiways, runway threshold wait times, queues at de-icing bays and de-icing rework, and flight cancellations. Takeoff delays are the result of accumulating delays at the gate, on the taxiways and de-icing bays and on runway thresholds. When taxiways are congested, accidents can happen, such as runway incursions and collisions due to poor controller-pilot shared situation awareness (e.g. May 2000 SH33/MD83 collision - see appendix). De-icing bay queues form in adverse conditions, when many aircraft request de-icing services simultaneously. De-icing rework is necessary when runway wait time exceeds the time of de-icing chemicals' effectiveness.

These inefficiencies undermine effective airport throughput and efficiency (e.g. measured in terms of punctuality). CDG airport is an open STS. Aircraft enter and leave the airport. Aviation safety regulations set an upper bound on airport throughput (separation minima between departing aircraft, regulated departure slots). Environmental conditions (wind, weather) further constrain throughput (increased separation minima) and degrade operational efficiency. As underlined previously, CDG displays emergent behavior, which makes system architecting complex: *deciding where and how to intervene in airport surface operations to have (the most) beneficial impact is problematic because emergence causes unanticipated effects in response to any given intervention*. Decisions affecting one part of the system architecture may have an impact on a completely different part. For instance, turnaround activities form a rigid, procedural, linear sequence: clearances and other communications and milestones time the progress of any flight through the linear sequence of turnaround states. Additionally, each flight depends on preceding and following ones, since they share airport resources (e.g. gates, pushback trucks, ground personnel, de-icing facilities). With such architectural connectivity, transforming the system is difficult as changes can ripple through the operations and cause

negative consequences far away from the locus of intervention. For example, speeding up the turnaround process at the gate may result in shorter in-block times but if too many aircraft pushback[8] simultaneously from different terminals and taxi to the same runway, there could be long wait times at the runway threshold, thereby cancelling the time saved at the gate. These observations align with characteristics of complex systems outlined in section 2.4: policy resistance, counterintuitive behavior, non-linear interactions, local cause-global effects, etc.

The hypothesis is that much of these inefficiencies are due to poor situation awareness, especially on the part of air traffic control and management: reliable tactical information and decision power is needed to enable flexible, real time operational optimization. For example, for EUROCONTROL to have the flexibility of swapping slots between flights of two different airlines, it needs to know airlines' schedules, preferences and current flight status, in addition to knowing current network load and airport capacity and status. Locally optimal solutions and distributed decision-making do not amount to globally optimal airport (and network) resource utilization. If actors transfer the information they hold as well as some degree of authority to a central body, it enables this central body to devise globally optimal solutions. Of course, equity and transparency of the central body's decision-making strategy is *key* for fostering trust from all stakeholders. Referring back to Figure 1-2, stakeholder perceived architectural "goodness" matters as much as factual performance.

A first practical requirement is to develop a common information system (hardware, software, broadcasting channels, standardized data formats…). The volume of information thereby made available and the frequency at which it is updated creates a requirement for developing automated decision support tools to assist human operators (e.g. control tower traffic supervisors, or European traffic managers). The implementation of a new information system and decision support tools is complex for obvious safety, security and training reasons.

[8] *Commercial* flight times correspond to pushback times not takeoff times, which is why airlines are concerned that prolonged in-block times due to the CDM milestone procedure and pre-departure sequencing will affect customer perceived punctuality hence satisfaction.

Phenomena related to airport CDM are identified in Table 5-1 and each class of phenomena is explained in more detail in the following paragraphs. The identification of relevant phenomena proceeded from an extensive documentation search on the CDM system deployed at CDG as well as on the CDM@CDG project, and from diverse expert inputs collected during an immersive visit at CDG. Modeling the underlying mechanisms would help sense-making of high-impact intervention areas in the overall system architecture.

Table 5-1: Phenomena related to CDM at CDG

Human	- CDM modifies flight management procedures and work practices, HMIs, mental models, especially for air traffic controllers and pilots.
Technical	- Heterogeneous IS are integrated into a collaborative IS, posing compatibility, system safety and data security challenges. - CDG surface layout offers many potential taxi patterns and congestion nodes. - Heterogeneous fleet characteristics affects aircraft surface traffic (e.g. wake turbulence, weight, wingspan…)
Operational	- Nominal procedures and actors' roles in the CDM turnaround process. CDM Cell intervention when activated perturbs nominal decision-making strategies and roles. - Taxi patterns: size of QFU-parking pairs space, variable taxi times. Runway dynamics: high frequency, inertia to configuration change, runway pressure, and open to environment. - Departure sequencing requires optimizing for many variables, being able to do so frequently and quickly. It is follows from the milestones and information sharing.
Socioeconomic	- Stakeholders with unequal appreciation of benefits from and trust in CDM: operators (controller, pilot), customer (airlines), and manager/developer (ADP, DSNA, Air France). - Aircraft operator economics: implementation costs (training, IT capital and maintenance), risks (data security, IS safety), and benefits (saved fuel and emissions, avoided delays and cancellations, customer/passenger satisfaction). - Airport economics: benefits (less delays, higher quality of service, increased traffic, avoided emissions, less regulations), costs (training, development, IT capital and maintenance costs), risk (safety, security). - Macroeconomics: fuel price, aircraft operators competition (for airport slots, performance, customers), and negotiation (DFLEX slot swapping system)
Environment	- Weather: wind (configuration); snow, fog/cold (de-icing); visibility (separation minima). - EU Emissions Trading system penalizes polluting emissions from aircraft operators. - Traffic schedule (exogenous variable): ~700 departures/day and as many arrivals, non-uniform because of Air France and FedEx hub activity and international arrivals - EUROCONTROL regulates takeoff slots (theoretically less so for CDM labeled airports) [9]

[9] A flight is regulated when its flight plan cuts through regulated en-route sectors.

5.3 Socioeconomic Phenomena

CDM at CDG is architected by ADP, DSNA and Air France, who coordinate with *Meteo France*, SESAR JU [10] , EUROCONTROL, and the Airline Operators Committee (AOC). EUROCONTROL initiated CDM on the European scale and plays a regulation role in daily operations. The Airlines Operations Committee (AOC) represents other aircraft operators but has less leverage power than Air France (Figure 5-2).

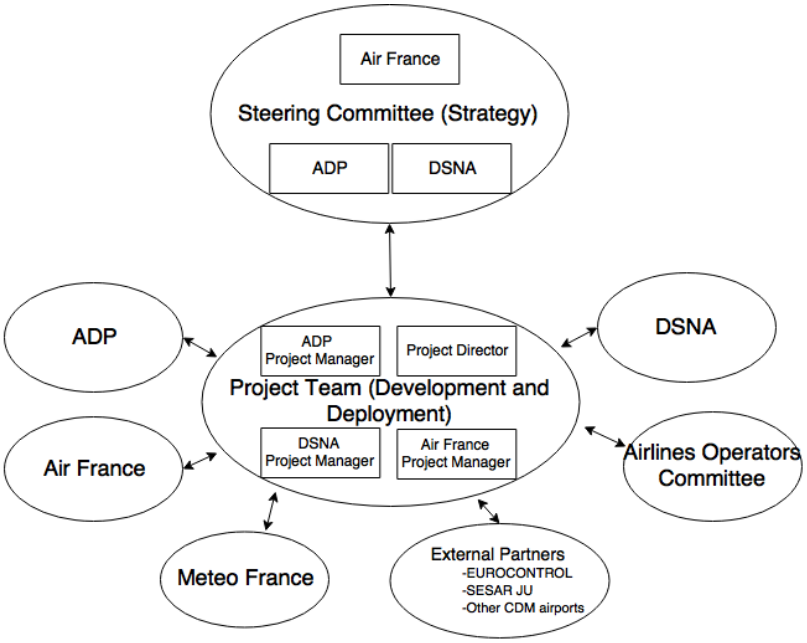


Figure 5-2: CDM@CDG Project Organization

Under nominal operating conditions, turnaround processes are almost unchanged by CDM, except for milestone/flight progress denominations that were made homogeneous across platform users. Under adverse operating conditions, the CDM Cell is staffed in addition to nominal operators. Some functions/roles then overlap between the CDM Cell and the control tower. For example, during a firefighters strike on February 12, 2015, airport capacity was significantly reduced for safety reasons and the CDM Cell was activated. The control tower supervisor and CDM Cell favored different traffic management strategies. The control tower switched to non-sequenced mode (calculated pre-departure sequence is no longer taken into account by

[10] Single European Sky ATM Research Joint Undertaking is the European program (2008-2020) that manages Europe's transition to a high-performance ATM system, CDM being one of its initiatives

controllers), whilst the CDM Cell remained in sequenced mode, leading to temporary misunderstandings. The larger an organization, the more complex it is to architect; currently CDM involves approximately 60 ADP agents, over 300 air traffic controllers, approximately 430 aircraft operators and 8000 crews. February 2015 events also underline the openness and sensitivity of the CDM system to perturbations from its social environment.

The value that stakeholders perceive in CDM is unequal. ADP committed significant financial investments. DSNA and Meteo France were challenged to become close partners and develop a common tool. Air France allocated human and financial resources to the project, although its economic model was increasingly challenged by competition in the airline industry (e.g. low cost models).

EUROCONTROL advocates CDM is a low cost, high return initiative (*Airport Collaborative Decision Making Implementation Manual Version 4*, 2012). However, it is difficult to evaluate the net economic impact, as CDM is interwoven with regular airport and airline economic activities as well as other SESAR JU initiatives. Given there is no authority with the legitimacy to enforce CDM, economic returns on investment essentially determine whether stakeholders buy in. Intuitively, CDM should benefit all stakeholders (Figure 5-1) and produce a virtuous cycle of self-reinforcing collaboration (Figure 5-3). However, Figure 5-3 does not account for competitive or gaming attitudes. It can also turn into a vicious cycle if CDM does not produce the expected benefits. Finally, it aggregates stakeholders on one level, yet it was shown earlier that different stakeholders perceive performance differently (Figure 5-1).

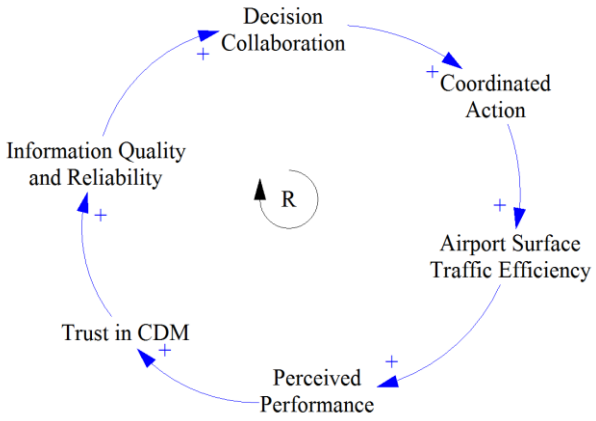


Figure 5-3: CDM Stakeholder Trust-Collaboration-Efficiency Reinforcing Loop

5.4 Human Phenomena

The criticality of air traffic safety compels to predict the impacts of any procedural change on air traffic control before implementing it, i.e. predict how the change will affect controller task performance (Bonaceto, Moertl, Estes, & Burns, 2005). CDM introduces a departure management system (DMAN), which links air traffic controllers with the ACIS, and thereby with the pre-departure sequencing algorithm, EUROCONTROL, crews, ADP and Meteo France.

DMAN provides decision support for giving clearances at the right time and alerts controllers about off-procedure flights. Eventually, controllers have to intervene to actually give clearances to crews (over the radio or data-link) and activate flight plans. Air traffic control and management (ATCM) is a fundamentally human-centered process consisting of the negotiation, execution and monitoring of contracts between human agents (controllers and pilots) for the allocation of limited airspace and airport surface resources. The decision processes tend to be semi-structured/non-deterministic (Hansman & Davison, 2001). Many routine procedures in ATCM can be represented by straightforward rules. However, in disrupted conditions ATCM decision processes are often unstructured. Consequently, the ability to automate ATCM processes is limited in spite of technological advances. The responsibility and authority for negotiation will continue to rest with human controllers and pilots (Hansman & Davison, 2001). Such a "humanized" version of ATCM is arguably slower than an automated one, limited by human capacity for information processing (Hall-May et al., 2011).

The premise of CDM is that information sharing improves situation awareness. Situation awareness is "the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future" (Endsley, 1995). Better SA increases operational efficiency, decreases the risk of incorrect controls (or absence of them) and mitigates their potential damage. The pre-departure sequencing algorithm performs an anticipation function by reverse-computing start-up times at the gate from takeoff times at the runway (see 5.3.6). This gives controllers the capability to project system status in a near future. Architecting the information sharing system to this end requires modeling system state evolution in the "near" future and deciding how near "near" ought to be. Finally, it requires deciding what pieces of information stakeholders should merge, what should be broadcasted and to whom. Systems architects ought to question whether information saturation

might impede on operator efficiency, whether information parity might create ambiguity in control authority, or induce undesirable or unstable gaming behavior between economically competing aircraft operators.

CDG surface traffic controllers are located in two surface control facilities. PREVOL tower controllers are responsible for giving departure clearances, ensuring that pilots are fully ready at TOBT, and for activating flight plans. The tower supervisor (CT) is responsible for devising a tower strategy (e.g. minimize taxi, minimize intersections), and deciding on runway configuration, capacity and pressure. CT can decide to switch to non-sequenced mode if deemed necessary (e.g. computer system malfunction). CT and PREVOL are the actors most affected by new CDM procedures, roles and tools.

Before the introduction of CDM, PREVOL was responsible for assessing the situation of flights, clearing and activating flight plans, based on procedures and often their better judgment (cf. semi-structured decision processes, Hansman & Davison 2001). In Cynefin framework terms (Snowden & Boone, 2007), CDM introduced automation that caused controller tasks to shift from the complicated/complex zone to the simple/complicated zone, from a "sense, analyze, respond" to a "sense, categorize, respond" scheme. DMAN displays all the information to communicate to pilots[11], flight milestones and corresponding actions to take through a color code (Figure 5-4). From a technical standpoint such an "industrialized", standardized process is more reliable than the former "humanized" version. However, from the operator viewpoint the benefits are less evident. Although PREVOL tasks are streamlined and workload is reduced, there is a non-negligible learning curve to adapt to the milestone concept, naming schemes, and timing (see appendix). Additionally the automatic flight sequencing and rigid procedures deprive PREVOL of some degree of autonomy, especially in adverse conditions. For CTs, the intervention of CDM Cell in adverse conditions is sometimes perceived as an infringement on their authority. These factors fostered some frustration on the part of control tower staff and resistance to DMAN, through the voice of (powerful) unions.

[11] Transponder frequency (SSR), assigned runway (QFU: 08L/R, 09L/R, 26L/R, or 27L/R) and standard initial departure (SID: departure route waypoints to fly through)

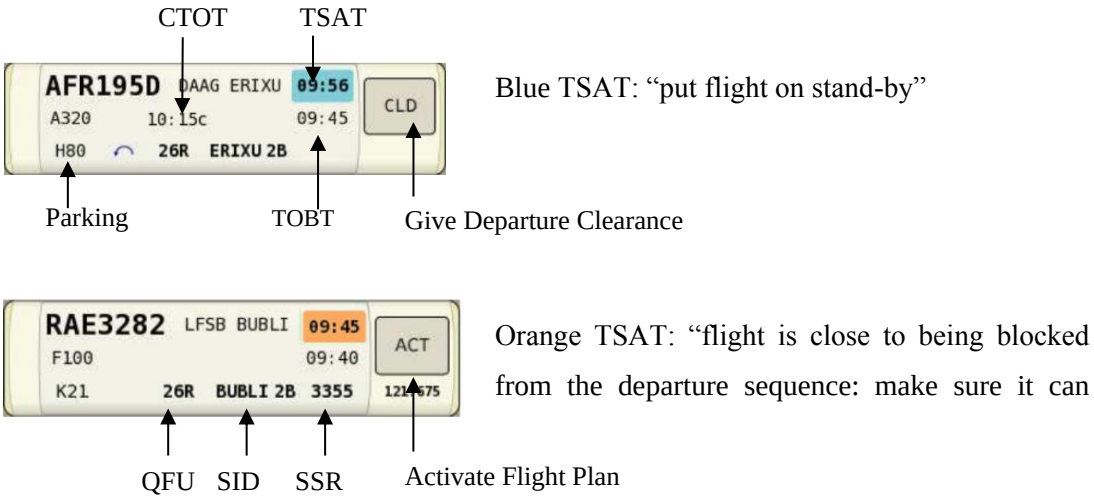


Figure 5-4: Example DMAN Flight Strips and Color Code

Because human factors are soft factors, they are traditionally difficult to account for in assessing systems architecture. CDM functional architecting is a complex problem, as it involves soft and hard factors (automated sequence optimization, but human controller eventually responsible for surface traffic safety), and overlapping functions between tower and CDM Cell.

5.5 Technical Phenomena

The Airport Collaborative Information system (ACIS) was developed to enable data exchanges between actors. ACIS integrates legacy information sub-systems, and homogenizes data formats. Security and safety features are important areas of concern for all actors. Additionally, airlines are concerned with leaking commercially sensitive data in a competitive context. Therefore, the architecture of the information system is subject not only to technical considerations of software compatibility, hardware maintenance and data formatting, it is also an economic investment with risks and opportunities that need to be assessed to engage stakeholders.

Another technical challenge lies in fitting CDM procedures and pre-departure sequencing tool to the complex layout of CDG platform its traffic dynamics, while keeping to the heuristic that simple is better. Section 5.3.6 expands on this issue.

The CDG platform ground layout counts four runways, 100km of taxiways, 2 control towers, and over 400 parking positions, 3 passenger terminals and 1 cargo port. Outbound and inbound traffic weave complex taxiing patterns between runways and gates, and can create congested nodes in the taxiway network (Figure 5-5). CT adopts either a "minimal taxi" strategy, or a "minimal crossing" strategy. Under minimal taxi strategy, flights are directed to the nearest-by runway. This is typically the adopted strategy in low-density traffic situations. Under minimal crossing strategy, flights are directed to the runway that will put them on a departure course such that crossing of aircraft in the airspace is minimal. This might entail significant taxiing distances. The tipping point between both strategies is determined from CT judgment.

A variety of aircraft is simultaneously found on the surface, in terms of operator type (cargo, regular, low cost, charter) and vehicle characteristics (taxi speed, minimum/maximum landing speed, number of passengers, turnaround minimal time, de-icing time, wake turbulence, separation minima, weight, wingspan). Agility in sequencing heterogeneous aircraft is needed. There are documented itineraries for taxiing aircraft to follow upon departing or arriving at CDG and pilots follow instructions from ATC. For example, some large aircrafts (e.g. A380) cannot use some taxiways because of their weight and wingspan.

An observation from control tower immersion is that controllers have derived tactics and heuristics from experience for handling familiar situations. Ground controllers navigate taxiing aircraft such that they don't all arrive at the same time at the same runway threshold before handing them over to runway controllers. Ground controllers accommodate different aircraft runway length requirements on different feeder ramps to the same runway in parallel to pace up departures. PREVOL controllers know that large passenger aircraft usually have last-minute TOBT updates (e.g. missing baggage, boarding delay). Therefore, if a series of large passenger flights is scheduled before a series of smaller passenger flights, controllers will tend to clear some smaller ones earlier if they're ready, anticipating that some large aircraft will vacate their slot at the last minute. Another example observed in PREVOL immersion is that cargo flights can be more flexible with their milestones, as their payload is less delay-sensitive than passengers. Controllers leverage heuristics and expertise to make judgment calls and such a decision-making logic is difficult to model formally let alone mathematically or in a computer.

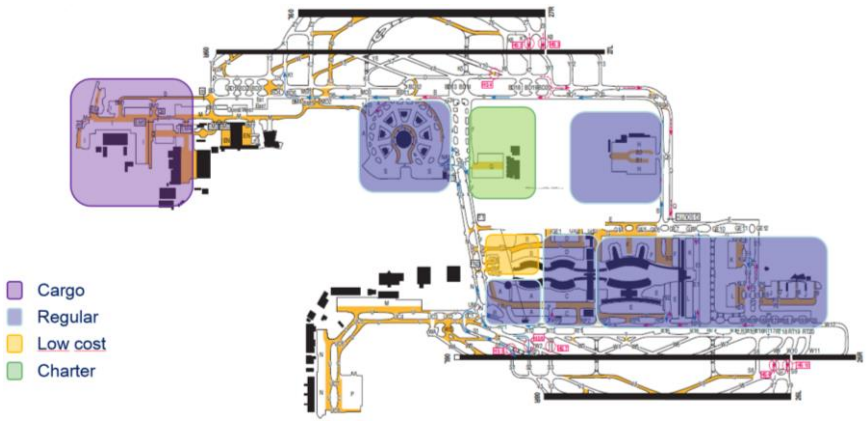


Figure 5-5: CDG surface layout and terminal types

5.6 Operational Phenomena

The turnaround process under CDM involves a complicated procedure that depends on the terminal, not to mention exceptions (e.g. military flights). Given CDM at CDG is a test case – not the goal – for this thesis we consider a single procedure type (CDG 2 milestone procedure), without loss of generality. The milestone procedure is described in the appendix.

Taxi time is unequal between flights, and is affected by several variables:

- Airport layout and infrastructure (Figure 5-5)
- Current configuration (i.e. runways in use) and runway pressure
- Runway crossings
- Flight assigned parking stand location
- Flight assigned QFU (runway)
- Meteorological conditions
- Aircraft type, weight and wingspan
- Remote (off block) de-icing time, including wait time at the de-icing bay
- Current surface traffic situation

The pre-departure sequencing algorithm (PDS or GLD for *Gestion Locale des Départs*) computes an optimal takeoff sequence and reverse calculates the optimal start-up sequence (Figure 5-6) by subtracting taxi time.

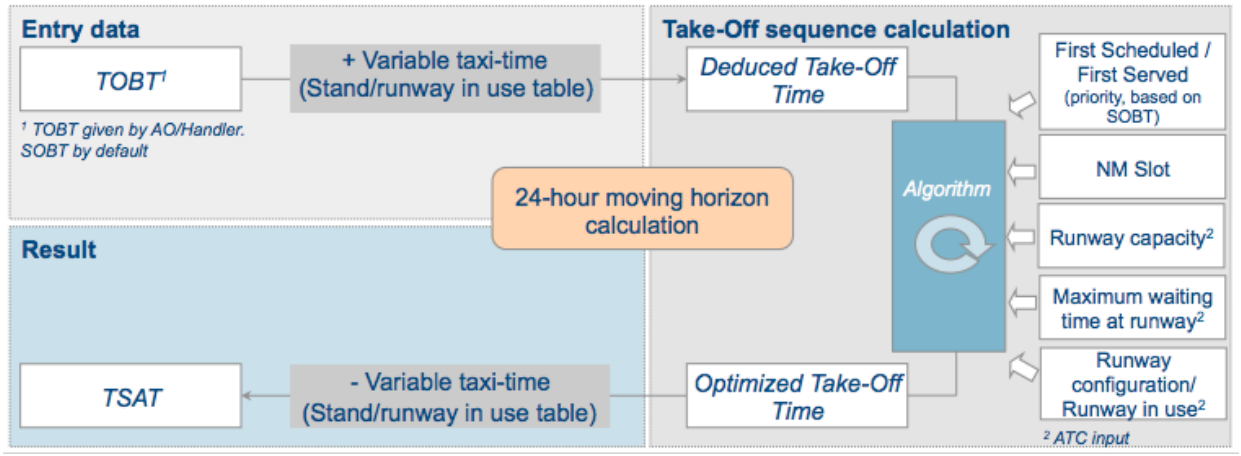


Figure 5-6: GLD Algorithm Logic

Nearly all stakeholders are involved in providing data to GLD. Ground assistants usually only do so indirectly, through the aircraft operator. Figure 5-7 represents the GLD inputs and outputs as an information flow diagram. [12] However, the information exchanges do not occur simultaneously or at the same frequency. For example, COHOR and SARIA-P process information weeks in advance, while TOBT updates can happen several times in the minutes preceding departure clearance. The optimal TSAT sequence never reaches equilibrium as it always covers the next-24-hour window and is refreshed every 30 seconds by the GLD engine within this window. The variety of time scales, the number of variables to solve for, the volume of information processed, the openness to information from EUROCONTROL and weather services, and the non-equilibrium of the departure sequence are characteristics that confirm the complexity of the data and information system, which the pre-departure sequencing algorithm is at the heart of.

[12] For the sake of clarity, communications after start-up approval aren't represented.

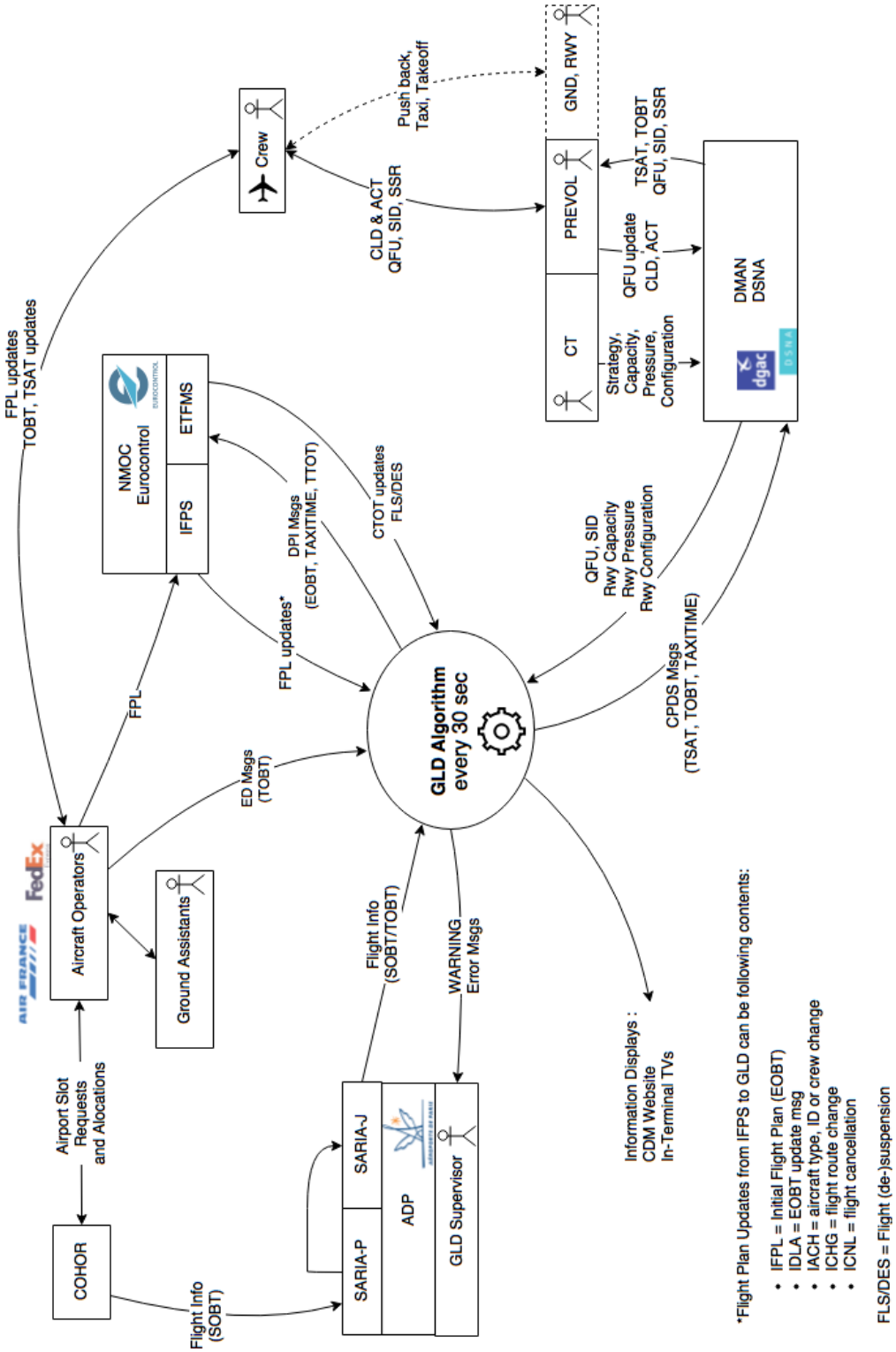


Figure 5-7: GLD Inputs and Outputs
(dotted lines: post sequencing information exchanges for push back, taxi and takeoff)

5.7 Environmental Phenomena

As explained above, architecting a pre-departure sequencing strategy requires reverse calculations from takeoff slots imposed on regulated flights by EUROCONTROL (CTOT slots). These regulations are an important constraint to respect, since flights deviating from their regulated slot at CDG would affect traffic predictions in sectors situated all along their flight route, starting in Paris. Thereby, deviations from regulated slots in CDG ripple across the European air traffic network and can affect other sectors and airports more or less distantly geographically and in time. Theoretically, Advanced CDM airports are granted less restrictive regulations on outbound flights, but CDG's situation in the European network (surrounded by regulated air sectors) is such that nearly all flights emanating from the platform are regulated. This further emphasizes that, although the objective is to optimize traffic locally, external constraints and perturbations to an open complex system can have significant weight in architecting the system.

Traffic is not uniformly distributed over the course of the day and alternating periods of low and high traffic density make for a very dynamic sequencing problem that requires automated anticipatory agility. CDG handles approximately 700 departures daily and as many arrivals. Air France and FedEx hubs activity, together with international flights arriving/departing at different times of day depending on their origin/destination, make for a distinct peaks and valleys in the distribution of traffic density over the course of the day (Figure 5-8). Depending on traffic predictions, CT devises different control strategies or assigns flights to different runways (input to the departure sequencing system).

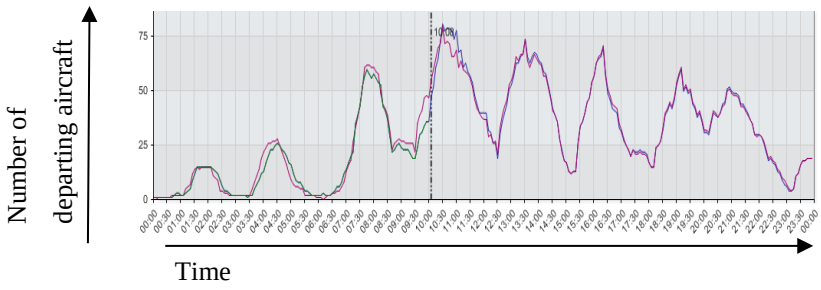


Figure 5-8: Example Outbound Traffic Volume over the Course of a Day at CDG

Varying weather conditions acutely affect airport operations: wind direction determines runway configuration (input to pre-departure sequencing), cross wind intensity affects the ability of

certain aircraft to land/takeoff safely, snow or harsh winter conditions require that aircraft make a pit-stop at de-icing bays thereby affecting taxi-out time (input to pre-departure sequencing), visibility conditions (e.g. fog) affect separation minima and thereby runway capacity (input to pre-departure sequencing). Over the course of one day conditions can be relatively steady (e.g. clear sky, steady or no wind) or quite unstable (e.g. in case of convective weather, wind direction can change drastically over the course of a few hours).

The European Union Emissions Trading system penalizes polluting emissions from aircraft operators. The EU ETS consists in trade-able allowances covering a certain level of yearly CO_2 emissions from their flights. The cost burden on aircraft operators is linked to their aggregate taxi fuel burn. Hence, the economic pressure that EU ETS puts on aircraft operators is an additional incentive for them to engage in CDM, not only to save fuel but also the cost of penalties.

5.8 Modeling CDM at CDG

Having underlined the socioeconomic, human, technical and operational phenomena in CDM at CDG, this section proposes modeling approaches to identify high leverage interventions and more generally to support sense-making in systems architecting.

Socioeconomic and human issues lie in convincing two classes of stakeholders of the net benefit of adopting CDM: users/operators and economic stakeholders. In the case of CDG, users have close ties to economic stakeholders. From the project organization, the main economic stakeholders are Air France, DSNA and ADP. At CDG, Air France pilots and DSNA controllers have considerable leverage because of their critical role and respective (powerful) unions. The incentives for users/operators and economic stakeholders differ and their perception of systems architecture goodness may be opposite. For example, economically speaking more automation might be perceived as more efficient, reliable and predictable, while from a user perspective, more automation might be perceived as work culture disruption and expertise loss. Value models support sense-making of heterogeneous stakeholder preferences (e.g. multi-attribute utility, measures of effectiveness, cost-benefit analysis, analytic hierarchical process). Architectural choices may concern traffic control procedure logic and milestone timing: what data is included in/excluded from information sharing, algorithm logic, how accurate or fuzzy the pre-departure sequencing is, whether it can achieve just-in-time aircraft delivery to the runway or whether

some buffer ought to be maintained (runway pressure or wait time). Choices may concern the degree of control decentralization (e.g. authorizing aircraft operators to swap slots internally), the degree of task automation, the structure and roles allocation of the multi-stakeholder organization running the airport. Value attributes - depending on the stakeholders - may be cost, safety, efficiency, delay, data security, and the equity of departure sequencing amongst aircraft operators. The previous discussion identified couplings between socioeconomic, human phenomena, operational and technical phenomena. Figure 5-9 visualizes main phenomena from Table 5-1 and relations between them, following the framework delineated in chapter 3.

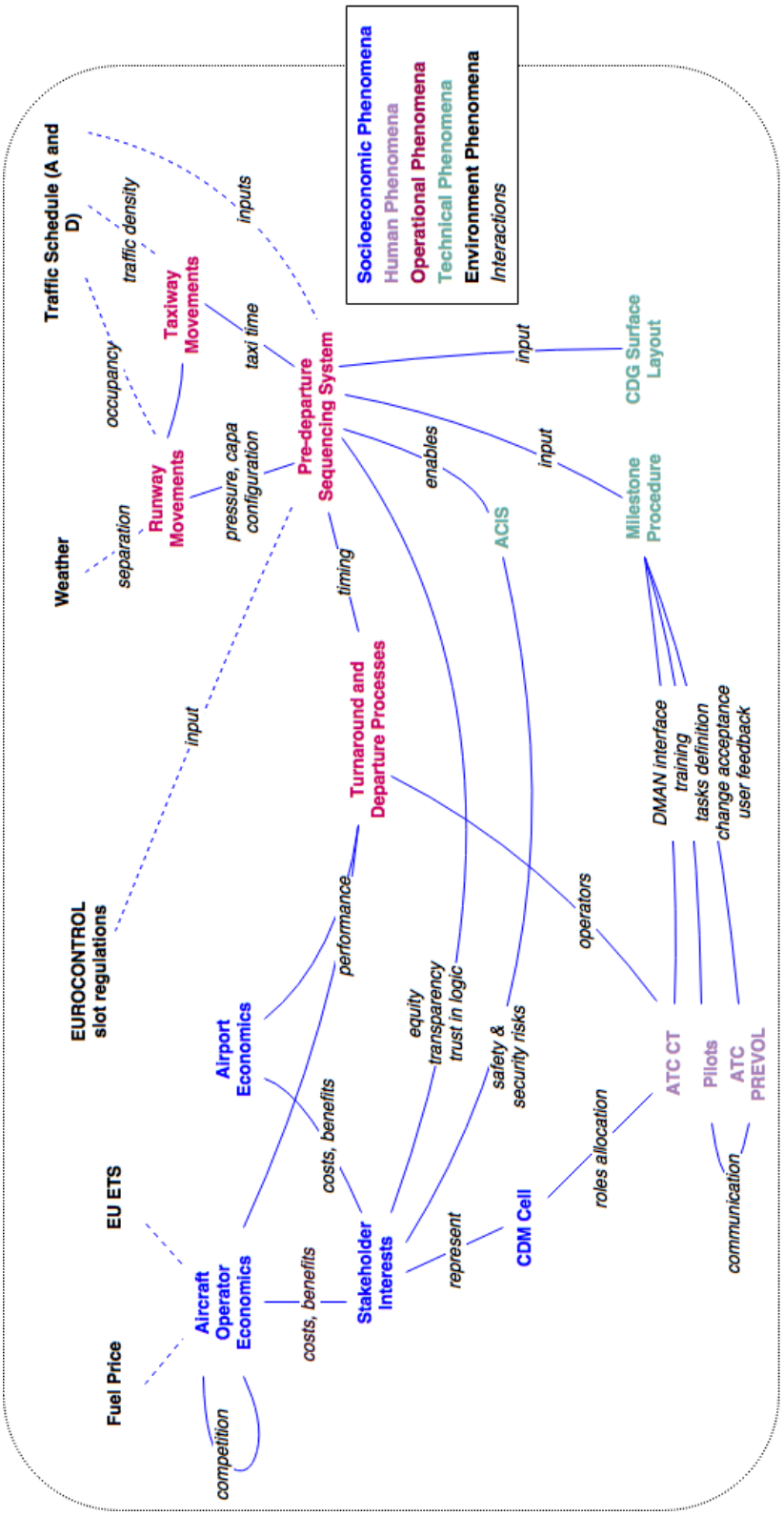


Figure 5-9: CDM Architecture Phenomena-Based Description (cf. Figure 3-2)

The highly dynamic nature of the phenomena compels to devise dynamic models of them. Airport and CDM performance indicators (Table 5-2) cannot be determined a priori: they are function of how system behavior dynamically unfolds. Modeling a posteriori (e.g. statistical models) does inform architects on current architecture performance and sheds light on some potential areas for improvement, but it is constrained by the established architecture. Without a dynamic modeling capability, foresight into the effects of changes to systems architecture is limited. Documentation and communication support material about CDM often comes with static models (e.g. structural diagrams, statistics), which are good at descriptive ends, but come short of prescriptive and exploratory purposes. An effective architectural proof of concept ought to take the form of a dynamic model or simulation with which exploratory and prescriptive modeling can be done – as experimenting with the system itself is prohibited out of obvious safety reasons. Most current modeling practices give stakeholders "read-only" hindsight while systems architecting needs experimental capability and foresight.

The economic benefits of CDM, praised by EUROCONTROL, cannot be determined without accounting for the specifics of each airport. There are as many variants of CDM as there are airports applying it, with specific local constraints and opportunities (e.g. stakeholders, infrastructure, airspace structure, fleet composition). Furthermore, there are non-obvious trade offs that only behavior modeling can resolve. For example, CDM allows avoiding some cancellations and rescheduling; therefore, there is an opportunity to welcome additional traffic. However, with increased traffic comes the risk of saturating capacity and creating new delays. The "sweet spot" (optimal traffic volume with CDM) is a dynamic variable, which will depend among other things on airport layout and procedure design.

Not all CDM performance indicators are quantitative, thus they are difficult to account for (e.g. air traffic controller workload, aircraft operator satisfaction with CDM).

Table 5-2: Example Performance Indicators

Safety	Number of runway Incursions
	Aircraft proximity in terminal airspace
Environment	Departure taxi time
	Runway wait time
	Taxi-time delay (effective-forecasted)
	Total fuel consumption/CO_2 emissions
Capacity and Schedule adherence	Runway throughput, nominal conditions
	Runway throughput, adverse conditions
	Slot adherence
	Off-block adherence (TSAT-AOBT)
	Average de-icing time and wait time
Quality of service	Departure punctuality
	Regularity, Predictability
Economic	Comparison between airlines

CDM-related operations at CDG evolve simultaneously on multiple time scales that range from minutes (TOBT updates for one flight) to hours (runway departure sequence) to days (airport configuration changes). A modeling approach encompassing this continuum of time scales is desirable to analyze alternative futures under different architectural decisions (e.g. simulation-based approach using layered/encapsulated models on different time scales). Turnaround processes are fully characterized by discrete-time statuses and events (for example: landed, taxiing, in-block, servicing, off-block, taxiing, de-icing, queuing, departed). In the space domain, continuous description might be useful in describing an aircraft's position on the surface with fidelity. However, even position can be discretized, for example taxiways between intersections can be represented by links in a network model. The exact position of an aircraft on links (e.g. half-way) is not accounted for. The links may be modeled stochastically from statistical distributions of travel time, stop occurrence and stop length (e.g. collected from historical data). A prescriptive approach would impose a given speed on each link for example. The more accuracy is sought in estimating taxi times, the more detailed the surface positioning model will be. Dynamics of airports can chiefly be modeled as discrete (finite time communications and resource use, milestone-bound flight statuses…).

Under nominal conditions, air traffic controllers and pilots are autonomous decision-making operators. The inter-airline slot swapping system – DFLEX – introduces airline competition and negotiation, i.e. a further decision-making entity, but this referent characteristic is not included in the model proposed here. Ground assistants react to arrivals and departures and allocate their resources accordingly. ADP operators intervene in the background of operations (e.g. an operator monitors the pre-departure sequencing system, a help desk answers questions about CDM, facilities staff inspect infrastructure and do the de-icing). The pre-departure sequencing system (GLD) is a "machine" decision-maker. An action taken by pilots (e.g. TOBT update) will have an impact on the flight schedule, hence on the pre-departure sequencing, and vice versa. Neither party has the power to unilaterally decide on the course of the entire system. A model of system behavior ought to account for these diverse, distributed, decision-making agents, who impact airport operations. It ought to discriminate between the respective degree of autonomy and action options of each type of agent. It ought to account for ground assistants' limited resources, for procedures that constrain both pilot and controller decision degrees of freedom, and for the pre-departure sequencing logic, which determines the algorithm's decision/output. Furthermore, as humans aren't machine-predictable, the nature of the models representing humans ought to be different from the model representing deterministic agents (e.g. algorithm agent).

Information sharing, as defined by EUROCONTROL, sets minimum requirements for information exchanges between agents: the content, timing, sender and recipients of a piece of information. Information exchanges with the environment are usually through a display or monitor, under standardized data formats: weather metadata (e.g. <wind speed>, <wind direction>), DPI message metadata (e.g. <flight ID>, <TOBT>). The degrees of freedom in architecting the information network are limited by EUROCONTROL information compatibility requirements. For instance, to communicate TOBT as defined by EUROCONTROL[13], not AOBT nor EOBT or any other arbitrarily chosen time. The information exchanges can be represented onto a static diagram, or dynamically, using an agent-based network. Such a model ought to be flexible in order to be able to include specifics of the implementation of information

[13] "The time that an aircraft operator or ground handler estimates that an aircraft will be ready, all doors closed, boarding bridge removed, push back vehicle available and ready to start up / push back immediately upon reception of clearance from the tower" (*Airport Collaborative Decision Making Implementation Manual Version 4*, 2012)

sharing at CDG, for example the fact that aircraft operators can negotiate among themselves to swap slots in real-time (DFLEX system).

Among the performance indicators mentioned earlier are traffic variables. A model of CDG surface (gates, taxiways, runways) together with aircraft movements could provide not only an evocative, natural visualization of traffic, but also the data necessary to evaluate traffic-related indicators. A dynamic surface traffic model could generate real-time data (e.g. actual runway wait time, dynamic estimates of taxi-out time) to be inputted back into the decision-models, if desired. Airport surface traffic models have used regression techniques, queuing models or network models. The use of one or another technique depends on the data available to build the model and the purpose of using the model. If the purpose is to investigate the interaction between aircraft on the surface, to track surface movements real-time, a network model is probably best. If it is to evaluate aggregate waiting times, a queuing model is good enough. If data is available from past traffic logs, regression models can be used to calibrate network/queue parameters. If one wanted to dynamically optimize aircraft routing, a network model associated with a search algorithm would be necessary.

5.9 Proposed Composite Modeling Approach

Given the dynamic nature of CDM operations, a simulation capability would be greatly useful for systems architects to evaluate the impact of their choices on systems architecture. Flight progress and CDM procedures are naturally described by discrete states and transitions (landed, in-block, cleared for departure, … off-block, departed). It follows that a simulation platform ought to have a discrete time representation. Given the multiplicity of parallel processes simultaneously carried out at CDG, and the multiplicity of actors intervening in each process, it is believed that AB and DE modeling could both be useful for representing different phenomena.

AB modeling has emerged as a privileged approach for modeling decentralized human and social architectures with distributed decision-making and actions. Agent behavior types range from deterministically reactive (“categorize inputs and respond according to a fixed scheme, a fixed rule set”) to learning and adaptive (“analyze inputs, change internal schema accordingly, and respond”). The validation of agent decision models is a sensitive part of building an AB model, as agent decisions are the foundations of the simulation. Human agent models (e.g. controllers)

should reflect cognitive processes of decision-making, as well as CDM procedures (Figure 5-10). State of the art modeling of human task performance and perception could inform decision-making, human error, and human-automation interaction for building human agent models. Machine agent models (e.g. of the GLD) should reflect the logic of the software embedded in them and CDM procedures. A modular agent definition is desirable to be able to test at will different roles allocations, procedure architectures, machine parameters etc.

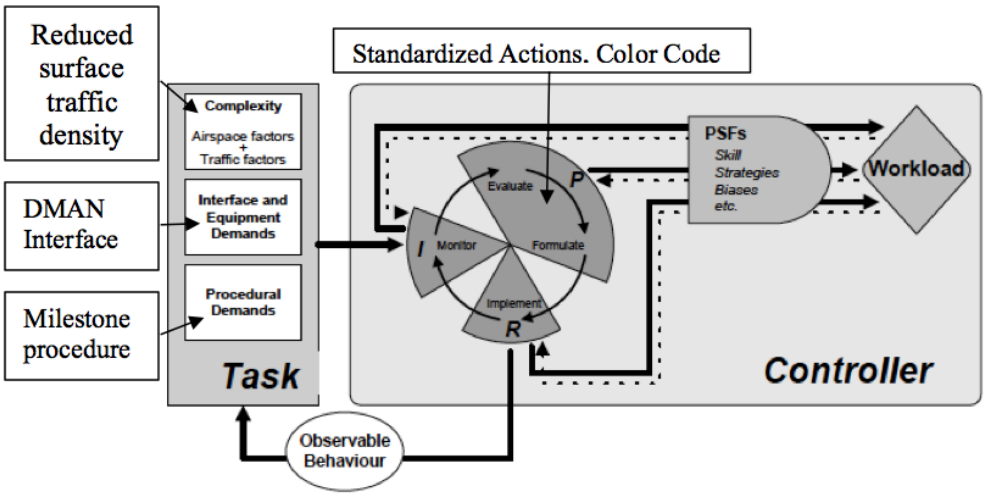


Figure 5-10: Model of Controller-Task interaction, adapted from (Hilburn, 2004) pp. 50-51

It can be argued that, however refined human decision models may be, the simulation cannot render human variability and unpredictability fully. With modular simulation architectures, one could potentially run a human-in-the-loop simulation, where a human, for example, replaces the PREVOL agent model. Of course, this requires some formatting (e.g. inputs formatted to a human-machine interface; controller actions translated to simulation-readable outputs) and running the simulation in real speed. Yet, such an approach would be beneficial, not only by feeding the overall simulation with more realistic behavior, but also by informing modelers about the validity of the substituted agent model in comparison. Air traffic control training benches are human-in-the-loop simulations of readily designed systems in highly realistic environment models, but they are not flexible enough to be used as an exploration tool for alternative systems architectures. For such flexibility in modeling different architectures, a simulation model ought

to be kept simple. It need not be highly realistic in all aspects. Finding the right level of abstraction is a challenge.

Agent models can also embody technical elements, although the internal agent model will be different from a human internal model. A runway for example, can be modeled as an agent. Internally, its behavior could be represented by a discrete event queuing model: the queue being the stack of aircraft waiting for takeoff, the outflow (takeoff throughput) being limited by shared and regulated use of limited resources (runway capacity, separation minima), and the inflow coming from taxiing aircraft arriving at the runway ramp. Minimum allowable separation distances between two aircraft depend on the turbulence in the wake of the first and the size of the second. In reality, the separation between successive aircraft on the runway varies around (usually above) the theoretical minimum; this variability could be stochastically modeled. ATC usually assigns separate runways to departures and arrivals, and currently CDM chiefly concerns outbound flights. However, CDG can be configured such that one runway accommodates a mix of takeoffs and landings, and future CDM developments aim at further integration of inbound and outbound flight legs. A valid runway model ought to be able to account for arrivals as well, for example in the form of stochastic perturbations to takeoff throughput rate. For departures, more detail is needed to eventually be able measure CDM performance (cf. indicators in Table 5-2). Similarly to runways, de-icing bays are a limited, shared resource, de-icing takes a minimum time depending on the type of aircraft, and queues typically form at the de-icing bay thresholds on snowy days. They could be modeled in a similar fashion if needed, but this is a referent characteristic left out of the proposed modeling approach.

Taxiways are more complex than runways (CDG counts 100km cumulated, networked taxiways). Taxi time determination is a hinge in the current pre-departure sequencing architecture (Figure 5-6), yet its present implementation is less accurate than takeoff time determination is: taxi times between pairs of parking stands and QFUs are stored in static look-up tables that were derived from recorded data. The impact of the most accurate takeoff-sequencing tool would be thwarted if intermediate taxi time estimates were inaccurate. It is believed that systems architects ought to evaluate the impact of implementing a more accurate or a fuzzier taxi time determination tool, trading off development effort versus additional performance. Many dynamic and contingent phenomena affect the taxi stage (cf. 5.6).

Oftentimes, more than one itinerary is available between a parking stand and a runway and the difference in taxi time between itineraries can be significant. The combinatorial space offered by the network of taxiways is large and, in fact, pattern variability is observed (see appendix). Currently, air traffic controllers are responsible for assigning an itinerary to pilots. Systems architects have to choose a taxi time calculation strategy for the pre-departure sequencing system. With appropriate, elaborate simulation capabilities, taxi time could be derived from a dynamic network model that accounts for precise, dynamic taxi itineraries. Even a simpler analytical function would improve on current practice by considering dynamic variables such as overall traffic density at time t, and aircraft type concerned. The effect of such choices on the pre-departure sequencing and overall performance could be explored over several simulation runs.

Turnaround processes produce a wealth of data about each flight (boarding information, destination, communication frequencies etc.) and accounting for all data in any model would be overwhelming. The pilot and flight/aircraft can be bundled into one agent model, unless pilot-system interaction is of particular interest. Schedule and aircraft type data are deemed most important for CDM purposes. Therefore, a discrete state-transition model can be adopted, enhanced with variables such as aircraft type, taxi speed, fuel burn rate etc. For calculating performance indicators, both scheduled and actual flight milestones must be recorded in flight/aircraft agent models. EUROCONTROL imposes slot regulations on flights, but European-scale traffic management mechanisms are beyond the scope of the proposed model. Hence, EUROCONTROL regulations could be modeled as external perturbations to aircraft agent schedules.

Interactions between agents are a primordial driver of any AB simulation. Fortunately, in air traffic management, most are rather straightforward to define:

- Procedural (voice or radio) communications between pilots/aircraft and controllers (e.g. clearance requests and approvals).
- Information exchanges between pilots/aircraft and the GLD (schedule, schedule updates).
- Information exchanges between CT controller and the GLD (runway capacity, pressure, configuration entries).

- Flight/aircraft transfer from gate to taxiway, to runway queue, to runway.

A modular, flexible simulation model, with dual technical and social modeling capabilities, would be useful for systems architects to communicate with stakeholders about dynamic CDM concepts (e.g. pre-departure sequencing logic), where static representations are much less effective. It would allow stakeholders to explore options, "playing" with the simulation engine as a serious game, or even in hybrid simulation environments. Provided model interaction doesn't require extensive expertise, that learning to use the model is relatively easy, this could be a means to foster cross-stakeholder awareness and collaborative systems architecting and design. From a systems architecture and design viewpoint, modeling architectural choices or design alternatives into such a simulation and comparing their modeled performance, early in the development lifecycle could save costs of delaying some architecture decisions. From a human-model interaction viewpoint, a bottom-up model would potentially exhibit emergent behavior (positive or negative), and thereby force CDM systems architects to question their assumptions, propose alternative solutions etc. If systems knowledge goes into building models (descriptively or prescriptively), models – especially dynamic models and simulations that surpass human capabilities at dynamic analysis - also generate information that modelers can capitalize on to enrich their knowledge and beliefs.

The proposed phenomena-based description frames the selection and composition of more detailed, formal models and provides a big picture to make sense from multiple modeled phenomena. Sense-making proceeds from recursive experimental modeling activities of model set selection, model trading, hypothesis testing, interactive visualization and stakeholder involvement.

5.10 Extension to other STS

This section considers the extension of the proposed modeling approach to other CDM airports and of the proposed framework to other STS.

A combined AB/DE modeling approach is thought suitable for modeling collaborative airport operations in other locations than CDG, albeit differently. European CDM airports have implemented EUROCONTROL principles with variants specific to local constraints, opportunities, goals, and risks. For example, the DFLEX system that allows aircraft operators to

swap slots amongst themselves is a specificity of CDG. This architectural choice calls for an additional layer of communication and decision-making, with internal negotiation and perhaps even gaming behaviors enabled amongst aircraft operator agents. Depending on the airport, the responsibilities of air traffic controllers may be distributed differently, which impacts agent roles. Additionally, taxiways may not be as complex in other airports as they are at CDG and therefore simpler modeling approaches may be quite suitable for representing variable taxi times elsewhere. At most small airports there is no need for variable taxi times as the layout of terminals and runways normally means that the a default value will be sufficiently accurate. At medium and large airports, the configuration of the runways and layout of terminal buildings can result in significant differences in taxi times for arriving and departing flights. Significant variations can exist depending on the various types of aircraft using the airport, hence different types of aircraft wake vortex categories should be taken into account for airports with highly heterogeneous fleets (e.g. A380 to small business aircraft), or less so for more homogeneous fleets (e.g. medium-sized, regional flights). Some smaller airports may require CDM only at certain periods of the year when traffic conditions require it (seasonal peak, winter conditions, temporary capacity limitation). In such cases a "lite CDM" version ought to be architected that can flexibly be activated and de-activated. Nordic European airports will have a greater need for modeling and architecting the de-icing system. Airports planning on construction to expand their capacity may have a need for modeling CDM under alternative extension plans (e.g. configuration of a new runway/terminal). Each airport's present architecture results from evolutions and extensions in response to its specific operating environment. Each airport is a complex network of interconnected processes, systems and people; therefore, the organizational change process is highly specific (Okwir & Correas, 2014).

Chapter 3 showed that other complex sociotechnical systems, such as joint humanitarian disaster relief logistics and multi-modal personal mobility systems, displayed much different complexity flavors than airport collaborative decision making systems.

The complexity of joint humanitarian disaster relief logistics is chiefly socioeconomic (collaboration issues) and technical (logistics optimization). A hybrid DE/game modeling approach could enable sense-making of this complexity. Serious games are a means of studying social system behaviors such as gaming, competition, collaboration, collusion, negotiation or

strategy making. Given a chosen social/organizational strategy, analytic logistics modeling approaches (e.g. operations research) may be useful in solving technical issues. The two modeling approaches (social game and logistics analytics) may be deployed interactively: at each step, player decisions shape the joint logistics strategy, an analytic logistics model evaluates its performance and players thereafter decide on the next course of action. Such feedback enables stakeholders to explore organizational strategies and assess their performance (e.g. management flight simulators). Although an AB simulation could be used here (coded agents instead of human players), the propensity or adversity of stakeholders for collaboration may be difficult to render/to code without assumptions that would significantly bias simulation results. For prescriptive purposes, such an approach could however, be of interest.

The complexity of multi-modal urban mobility systems stems mainly from human factors and technical infrastructure choices. Decision-making (itinerary selection) is distributed. Individual value and behavior are dynamic functions: they depend on the states of the technical system (transportation options) and social system (load factor, information) at any given time. Therefore, a simulation capability ought to reflect technical architectural choices (e.g. tax policies, transit fares, transit network design). These choices indirectly influence individual/human behaviors, which, aggregated, produce system behavior hence performance. An AB simulation would be useful, and agent decision/value models ought to be traded to mitigate the bias introduced by the modeler at the agent level. The technical system could be modeled as a network with different links (itinerary segments) between nodes (intermodal platforms) representing different available transportation modes. Depending on the knobs (technical architectural choices) that a systems architect has available (e.g. service frequency, fares, introduction/deletion of links…), more or less variables will need to be included.

The selection of suitable modeling approaches depends much on the flavor of system complexity and user intent with the model. All social scales present in a system will likely not equally impact systems architecture (e.g. individual vs. enterprise level decision making). Technical structure may be part of systems architecting but only second to processes in terms of complexity, hence require less modeling depth. Metrics of interest could be static and/or dynamic. Socioeconomic complexity or model training/playing purposes may call for interactive modeling approaches.

Depending on the type of complexity displayed by the system at hand, and on whether sense-making is intended for systems architects or diverse stakeholders, model anatomy and interaction channels will be different. However from literature and the case examples in this thesis, adopting an exploratory phenomena-based architecture description in combination with a modular, multi-method modeling approach and interactive visualization techniques are useful strategies across complex systems.

6 Conclusions and Future Work

6.1 Summary

This thesis proposes a framework for sense-making of complexity in sociotechnical systems for the purpose of systems architecting. The proposed framework builds on research pillars consolidated by a literature review: complexity in sociotechnical systems, phenomena-based system architecture description and model-centric approaches for systems architecting. The framework adapts the phenomena-based logic from (Rouse, 2015) that is especially geared towards complex social systems and enterprises, but puts more emphasis on technical and operational aspects for the purpose of architecting sociotechnical systems. The framework starts with a description of systems architecture in terms of technical, operational, human, socioeconomic and environmental phenomena of interest. The notion of phenomena is key. Phenomena are a natural unit for dissecting complex sociotechnical systems. Indeed, a phenomenon is a fact or situation that is observed to exist or happen, especially one whose cause or explanation is in question (Oxford online dictionary).

Models should be developed with a clear purpose, a defined scope and awareness of the assumptions they rely on. Models provide a tool to answer questions about systems architecture, and hence are well adapted to a phenomena-based approach. These questions may concern the organization of the social system, the structure of the technical systems, their relations to the environment, their behavior, joint performance, or failure modes, for example.

Of course, the initial set of identified phenomena may be imperfect: As one attempts to define relationships between initially identified phenomena, inconsistencies, overlaps and missing or superfluous pieces become more evident. Exchanges with field experts in the case study provided original insights about relevant CDM phenomena at CDG. The down side of using phenomena as a unit for describing complex systems is that their selection is subjective. Hence inputs from diverse stakeholders ought to be taken into account in the identification of relevant phenomena, especially soft ones (e.g. social and human phenomena), that are less well understood and formalized, and often underrepresented by modelers.

Models need scoping in order not to become unwieldy: it was previously explained why a single modeling approach would be inefficient for large systems architecting and why modular modeling strategies should be adopted instead. The question is what goes into the modules. For example, in traditional technical systems architecting (e.g. a space station), one would build models for physically and/or functionally separate subsystems (e.g. life support subsystem, power and thermal subsystem, attitude control subsystem, propulsion subsystem), and connect subsystems through energy (e.g. recycling power demand), physical (e.g. deployed solar panels moment of inertia), and information links (e.g. attitude controls sent to propulsion actuators). Open sociotechnical systems, with soft, value-laden, densely and sometimes informally connected social and technical subsystem are difficult to formally decompose. Humans are much less deterministic than technical (designed) systems. Strongly coupled environmental dynamics, among other things, can introduce significant uncertainty. Flows in and out of the system and distributed decision-making among many actors cause the system to adapt (structurally, behaviorally). The framework proposed in this thesis suggests using phenomena as sub-model referents.

Chapter 4 elicited differences among modeling and simulation approaches. Model purpose and model characteristics guide the selection of appropriate modeling approaches for chosen phenomena. The framework also proposes to visualize phenomena relationships as a frame for composing more detailed, formal models. In this manner, modeling modularity and flexibility is enabled at the phenomena/module level. Agent based models have received considerable attention in literature to model complex sociotechnical systems, with applications in coordination and cooperation phenomena, traffic phenomena, and resource allocation phenomena among others. In the proposed framework, an agent-based model can be used within a module representing a phenomenon (e.g. taxiway traffic), but it is itself modular: agent models are the modules and interaction models the links.

In chapter 5, an agent-based platform was suggested to model the entire system. Sub-models (agent models) would represent runways, the taxiway network, aircraft/flights, controllers, and the pre-departure sequencing algorithm. Interaction links would essentially be information exchanges and aircraft flows. By modifying sub-models internally, one can test different systems architectures (e.g. procedures, algorithm logic, automation and human roles allocation).

Internally, agents of different natures may have very different structures (e.g. queuing model, cognitive model, decision tree, algorithm, equation…). Given the flexibility of a modular modeling approach, sub-models could be traded and the impact of the trade assessed. This was discussed for the "taxi movements" sub-model (network model vs. analytic average taxi time model), but can be done for any phenomenon/sub-model. The building and composition of models poses issues of assumption consistency, timing (in dynamic models), variable consistency (e.g. duplication, exogenous vs. endogenous) and input/output compatibility (e.g. definition, format). These would become salient in the process of model building and validation.

6.2 Limitations and Further Work

This work is limited by a single case study at Paris airport, and would be greatly enriched by a cross-comparison with CDM architectures at other European airports (e.g. Munich, Brussels). It is also believed that the framework can be applied to other collaborative systems, such as maritime traffic control, coordinated disaster relief logistics systems, or network-centric military operations. Chapter 3 illustrated the phenomena-based framework on two other types of sociotechnical systems and a thorough test case on those examples would contribute to further testing of the framework. Similarly, to the CDM at CDG test case, this would require collecting considerable amounts of data and documentation, narrowing down the case study to an amount of information amenable to modeling, making contacts and doing interviews within that case study.

This work made conceptual tradeoffs between modeling strategies for some phenomena in the test case (chapter 5) based on the characteristics and capabilities of known modeling approaches (chapter 4). Making a qualitative argument for the use of a modeling strategy over another, for the justification of assumptions and scope, has been underlined as a key step in model development (Dodder, 2006; Rouse, 2015; Sterman, 2002; Sussman et al., 2007). This thesis illustrated this step in chapter 5 but suffers from the absence of model building and validation steps. It is a limitation of the test case, where counterparts at Paris CDG expressed much interest for a modeling approach to be developed. It is also a limitation for the framework, which would be best tested by building and comparing models across several test cases, which this thesis did not have the time to carry out. If the ambition were to validate the framework, a set of case

studies would have to be carefully chosen to be representative of the variety of sociotechnical systems. These are all areas of possible improvements and future work.

6.3 Model Curation

Finally, chapter 4 put forward the idea of a model classification for the selection and composition of models. Recent multi-method modeling efforts suggest that it would be useful to have such a classification for the construction of modular modeling strategies for complex systems. The systems community has developed and instantiated many modeling approaches, practices, formal languages and toolsets, which are all areas of progress. If models and their instantiations could be managed as assets, documented, archived, protected, retrieved and re-used as such, multi-method modeling tasks would likely gain in quality and timeliness (Reymondet, Rhodes, & Ross, 2016).

Model curation has been raised as a topic of growing importance within the systems community. Rouse (2015) stresses that the wealth of existing models is often not used because of a lack of knowledge of these resources and the difficulty in accessing them. He asserts there needs to be a single point of access to this body of knowledge, enabled with downloading computer codes, documentation on assumptions and use, and dissertations on the development and validation of these models.

The 2015 IMCSE Pathfinder Workshop participants agreed that there is no consistent meaning across the "sea" of models that exist. There is a lack of precise semantics, especially in the behavior and timing/interactions of models (Rhodes & Ross, 2015). Many organizations are beginning to develop repositories of models and archive associated data, which is one of the fundamental activities in model curation. Repositories support re-use of building blocks from previous analyses and tailoring to the specific requirements of one's problem. Potential benefits include improving the quality and timeliness of analysis, by developing a repeatable framework providing guidelines for accessing, designing, and implementing models.

Evolving the science of model curation will involve a partnership of government, industry and academia. There are many challenges and interesting problems related to this topic, and there is great potential benefit for practitioners, researchers and educators. Further consideration is needed to understand the role of model curator and impact this can have in the field.

7 Appendix A: Acronyms

ACIS	Airport Collaborative Information System
ACT	Flight plan Activation
ADP	Aéroports de Paris
AOBT	Actual Off-Block Time
AOC	Airlines Operations Committee
ATC	Air Traffic Control
ATCM	Air Traffic Control and Management
CDG	Charles de Gaulle airport
CDM	Collaborative Decision Making
CDM@CDG	Name of the CDM implementation project at CDG
CFMU	Control Flow Management Unit
CLD	*Clairance Départ*, Departure Clearance
COHOR	*Comité des Horaires*, entity responsible for coordinating and allocating airport slots for major French airports
CPDS	Collaborative Pre-departure Sequencing
cSTS	Complex Sociotechnical System
CT	*Chef de Tour*, Tower supervisor
CTOT	Calculated Takeoff Time = EUROCONTROL slot
DGAC	*Direction Générale de l'Aviation Civile*, is the French aviation administration
DMAN	Departure Manager, or Departure Management System
DPI Msgs	Departure Planning Information Messages
DSNA	*Direction des Services de la Navigation Aérienne*, is the French air traffic services provider, part of DGAC
ERWT	Estimated Runway Wait Time
ETFMS	EUROCONRTOL Enhanced Tactical Flow Management System
EUROCONTROL	European central air traffic flow manager
EXOT	Estimated Taxi Out Time
FLS/DES	Flight Suspension/De-suspension Message
FPL	Flight Plan
GLD	*Gestion Locale des Départs*, PDS function at CDG
HMI	Human Machine Interface
IFPS	EUROCONRTOL Initial Flight Plan System
IS	Information System
NMOC	Network Manager Operations Center
PDS	Pre-departure Sequencing

PREVOL	Air traffic control delivery position
QFU	Runway magnetic orientation
Rwy	Runway
SARIA	*Système d'Affectation des Ressources et d'Information Aéroportuaire*, is the CDG information system that manages and stores information about all flights
SARIA-J	Manages flight information on the day a flight is scheduled
SARIA-P	Manages flight information until the day a flight is scheduled, dumps information to SARIA-J on the day of operations
SESAR-JU	Single European Sky ATM Research Joint Undertaking
SID	Standard Instrument Departure, i.e. route to follow after takeoff
SSR	Secondary Surveillance Radar
STS	Sociotechnical System
TAXITIME	Taxi time = EXOT + ERWT
TOBT	Target Off-Block Time
TSAT	Target Start-up Approval Time
TTOT	Target Takeoff Time

SOBT (scheduled off-block time): airport slot that the aircraft operator negotiated with COHOR. SOBT order determines the priority of flights in the pre-departure sequencing algorithm.

TOBT (target off block-time): real-time estimated time at which the aircraft will leave the gate. Aircraft operators can send TOBT updates at any time prior to TSAT to the pre-departure sequencing system (e.g. in case of a baggage delay). TOBT is by default initialized to SOBT.

TSAT (target start up approval time): time at which an aircraft must leave the gate in order to comply with all the departure constraints: COHOR determined SOBT, aircraft operator determined TOBT, network manager determined CTOT, runway capacity, and other flights scheduled on the runway. TSAT is computed by the pre-departure sequencing algorithm, which takes all constraints into account.

CTOT (calculated take off time): slot calculated by the network manager (EUROCONTROL). It results from considering capacity limitations for all airspace sectors the flight is supposed to travel through to reach its destination.

EOBT (estimated off-block time): a priori estimated time at which the aircraft will leave the gate. The aircraft operator is responsible for estimating EOBT before the flight plan is activated).

AOBT (actual off-block time): actual time at which the aircraft leaves the gate. It is a crucial to input AOBT into the pre-departure sequencing system so that the flight will not see its CTOT further delayed.

8 Appendix B: CDM at CDG - Additional Information

- **The milestone process**

The progress of a flight is tracked by a sequence of events, known as milestones. This unique representation of a flight's profile creates common situational awareness. Operators can update target milestones in real time. TOBT updates up until ASAT trigger automatic downstream updates by the GLD algorithm. Data quality (accuracy, timeliness, reliability, stability and predictability) is key as each milestone is coupled to preceding ones. As a flight progresses, actual milestones become available and can serve to update following estimates.

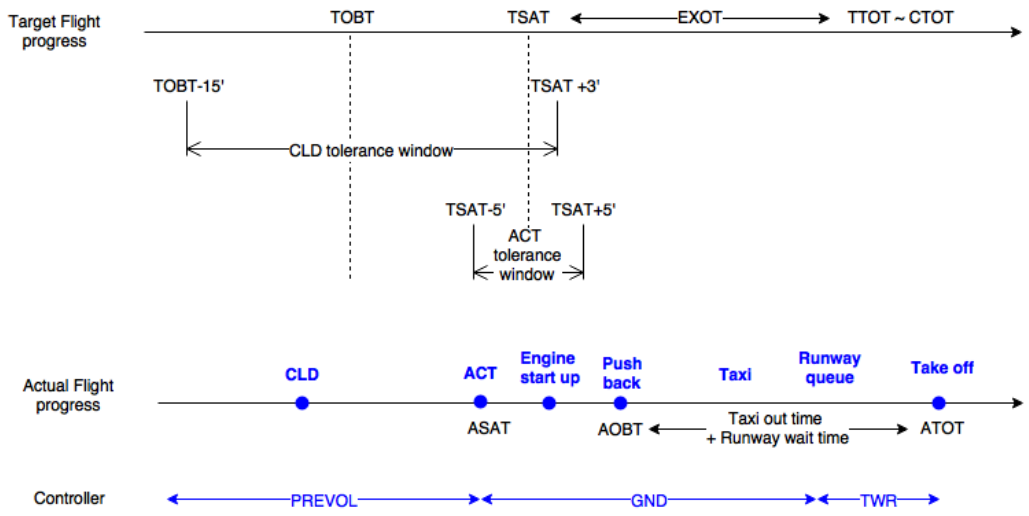


Figure 8-1: Milestone Departure Procedure

- **May 2000 SH33/MD83 collision**

Flight MD83 was cleared to take off from runway 27. Flight SH33 was then cleared to line up and to wait. The controller believed that the two aircraft were at the threshold of the runway, but SH33 had been cleared to use an intermediate taxiway. The tip of the MD83 left wing cut through the SH33 cockpit, fatally injuring both pilots. Poor situation awareness on the part of the controller and the pilots of SH33 contributed to the accident.

- **Taxi time and taxi pattern variability**

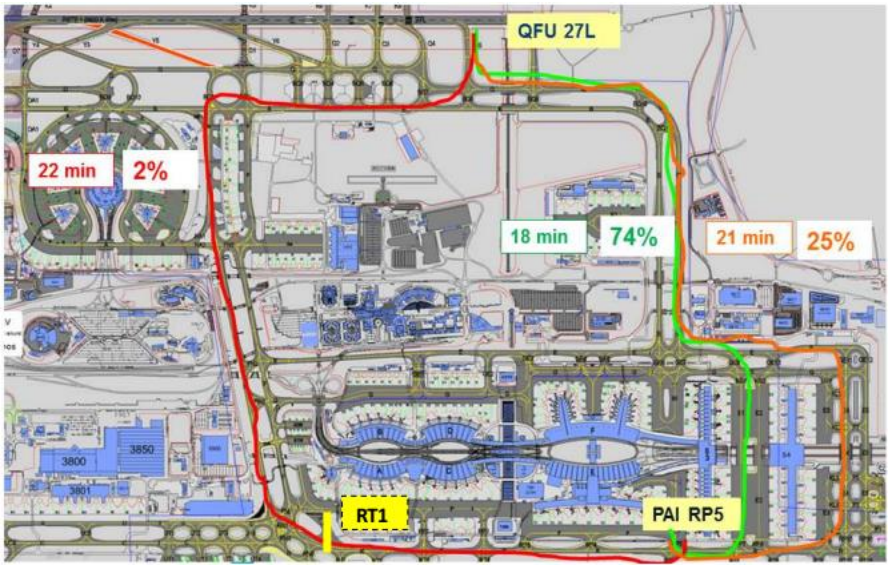
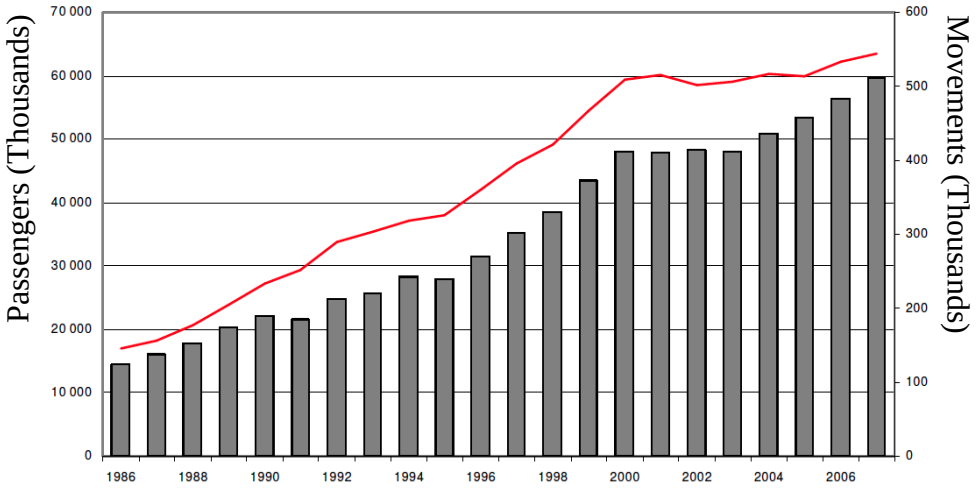


Figure 8-2: Alternative taxi itineraries from Parking RP5 to Runway QFU 27L, respective taxi times and frequency of use (from a 2014 CDG statistical analysis of recorded traffic data)

- **CDG traffic evolution (1986-2007). 2008 initiated an ongoing slowdown, but long-term forecasts predict traffic will grow beyond CDG capacity**



	1986/1990	1990/1995	1995/2000	2000/2005	2005/2007	2006/2007	1986/2007
Passengers	11,2%	4,8%	11,5%	2,1%	5,6%	5,5%	**7,0%**
Movements	12,5%	6,9%	9,3%	0,2%	2,9%	2,0%	**6,5%**
	4,9%	5,9%	5,2%	1,1%	9,9%	8,2%	**4,8%**

9 Appendix C: Complexity Taxonomies from Literature

(Sheard, 2012)	Structural Complexity	Subtypes are size (number of elements, number of types of elements, e.g. people, groups, units, computer nodes, number of systems engineering tasks in a development process), connectivity (number of connections, types and strength of connections, e.g. data, control, physical connections, dependencies between development tasks), and architecture (patterns, chunkiness of connections, inhomogeneity, boundaries, e.g. trees, layers, networks, chunks, fragmentation, teams in a development process).
	Dynamic Complexity	Subtypes are short-term dynamic complexity (nonlinearity, dynamic emergence, sudden rapid changes in system behavior – butterfly effect, e.g. homeostasis, feedback, time patterns), and long-term dynamic complexity (changes in number and types of things and relationships, e.g. system growth, adaptation, self organization, learning, landscape changes).
	Socio-political Complexity	Human cognitive limitations, multiple stakeholders, global context, environmental sustainability, economics, e.g. multiple, soft, value-laden objectives, multiple perspectives from different stakeholders, sociological aspects of teams and organizations, diverse operational boundaries, diffuse boundaries.
(Kreimeyer, 2011)	Structural Complexity	"The term 'structural complexity' addresses the coupling of entities (also called 'elements') in a system via relations (also called 'dependencies' if directed) [...] structural complexity management is interested in modeling, analyzing and synthesizing these entities and their relations with a view to understand or establish a certain behavior of a system".
(Cardoso, 2005)	Process Complexity	"Process complexity [is] the number of tasks in the process (task complexity), the degree of cross-linking in the arrangement of tasks, and the related decision points such as AND, OR, or XOR (control-flow complexity), the degree of dependency of the information objects and their mapping to tasks, resources etc. (data-flow complexity), and the degree of interdependency of resources and their attribution to the tasks in the process (resource complexity)".
(Broniatowski, 2014)	Structural Complexity	"A system is structurally complex when it is difficult to change its internal structure, and achieve a desired goal, e.g., in the system's behavior. [...] Structural complexity is therefore a function of the degree to which a system is ordered, or "not messy." This is a concept not dissimilar from thermodynamic entropy. In particular, we may define the complexity of a system as the entropy of adding a new link".

(Senge, 1990)	Detail Complexity	"Detail complexity comprises of systems with essentially hierarchical relationship structures with no lateral relationship links between the system elements, describable by a large set of variables (similar to complicatedness: many elements, possibly many connections, but the connections do not cause emergent properties, and the elements aren't self-organized)".
	Dynamic Complexity	"Dynamic Complexity describes a hierarchical structure with lateral relationships. It is complicatedness with two additional properties: emergence and self-organization. Cause and effect are subtle; effects over time of interventions are not obvious; the same action has dramatically different effects in the short and the long run; an action has one set of consequences locally and another set of consequences in another part of the system; obvious interventions produce non-obvious consequences".
(Highfield & Coveney, 1996)	Scientific Complexity	Behavior of macroscopic collections of such units that they are endowed with the potential to evolve in time.
	Mathematical Complexity	Number of mathematical operations needed to solve a problem […] sort of complexity of interest in computer science.
(Sterman, 2002)	Dynamic Complexity	"The counterintuitive behavior of complex systems that emerges from the interactions of the agents over time, because systems are constantly changing, tightly coupled, governed by feedback, nonlinear, history-dependent, self-organizing, adaptive, characterized by trade-offs, counterintuitive and policy resistant."
	Combinatorial Complexity	"Number of, or links among, the elements of a system, or the dimensionality of a search space i.e. finding the optimal solution from a very, very large number of possibilities"
(Perrow, 1999)	Interactive Complexity	"Characterizes a system in which two or more discrete failures can interact in unexpected ways. In many cases, these unexpected interactions can affect supposedly redundant sub-systems. A sufficiently complex system can be expected to have many such unanticipated failure mode interactions, making it vulnerable to normal accidents."
(Sussman et al., 2007)	Structural Complexity	Structural Complexity (also known as combinatorial, internal or detail complexity) exists when the system consists of a large number of interconnected parts.
	Behavioral Complexity	Behavioral complexity (also referred to as dynamic complexity) exists when predictions of system outputs or behavior is difficult. This can be found even in systems with low structural complexity when their parts interact over time in closely- coupled feedback loops. Even if we understand the internal behavior of individual subsystems and components, our lack of understanding

		of the relationships between these components and subsystems leads to difficulties in making predictions of overall CLIOS System behavior. Emergence is a specific example of behavioral complexity in which the laws or rules governing the behavior or individual components are simple, but the patterns of overall behavior that result are complex and usually surprising
	Nested Complexity	Nested Complexity is a concept that suggests a complex "physical/technical" system embedded within an institutional system (which we will later refer to as an institutional sphere). Moreover, the institutional system exhibits structural and behavioral complexity in its own right. The two-way interactions between the physical/technical and institutional systems create "nested complexity."
	Evaluative Complexity	Evaluative Complexity reflects the multi-stakeholder environment in which CLIOS Systems exist – different stakeholders value different aspects of system performance in different ways, making decision-making difficult. Simply put, what may be good performance to one stakeholder may not be good performance to another stakeholder. Even if one could make good predictions about the behavior of the CLIOS System when strategic alternatives are implemented, evaluative complexity means it is still difficult to make a decision about what to do.
(Magee & de Weck, 2004)	Complexity is related to the amount of information needed to describe the system (Kolmogorov, 1983) and is also a function of the number of (unique) elements in the system as well as the number and nature of their interconnections.	
(Drezner, 2009)	Technical Complexity	Technical complexity includes weapon system functionality and capability, including that related to the use of embedded informational technology.
	Organizational Complexity	Organizational complexity addresses the structures and interactions of the government and industry organizations responsible for system design, development, production and support.
	Environmental Complexity	Environmental complexity includes the political and economic context of the acquisition process, the threat environment and the operational environment (how the systems are intended to be used).
(Dwyer, 2014)	Architectural Complexity	Architectural complexity refers both to the number of components in a system and to those components' relationships with one another. (1) Technical Architecture represents Mission, Programmatic and Interference relationship between elements (2) Organizational Architecture represents expertise, responsibility, budget and authority interdependency types
	Design Complexity	Design complexity refers to the individual complexity of each of a system's components [...] Function of the technical maturity of each component: as a

		component's design matures and designers gain improved understanding and knowledge of its physical and functional properties, its design complexity is reduced.
	Process Complexity	The third type of complexity is not a function of a technical system itself, but rather, is a function of the external processes by which a system is developed. [...] Function of the constraints or conflicting requirements that are imposed during the component development process.
(Lessard et al., 2014)	Technical Complexity	Features from project location (development, proximity, geography, climate) and elements (number, interdependence, dynamism, diversity of disciplines, novelty-maturity)
	Institutional or Organizational Complexity	Features from framework (legislative, regulatory, normative, cultural-cognitive) and interests (number of stakeholders, interdependence, similarity/alignment, dynamism, novelty-maturity)
(Ham, Park, & Jung, 2011)	"It can be assumed that complexity as a conceptual construct is composed of a set of complexity factors, and that they have complicated interrelationships in determining the degree and nature of complexity. Although complexity factors are supposed to constitute the complexity, complexity can also be considered an emergent property coming out of the interactions between them. [...] Complexity cannot be absolutely evaluated by only a combination of complexity factors. Instead several factors based on several views can collectively indicate it" (e.g. Knowledge, Structure, Design, Context and Roles).	
(Xing, 2007)	Information Complexity	Complexity is the combination of three basic factors: quantity, variety and relation of basic elements; all three are evaluated by the mechanisms of observers' information processing and constrained by task requirements. [...] Information presented via visual display devices is processed in three stages: perception, cognition, and action. Through perception, a user acquires visual features of displayed information. The perceived information then feeds into the cognition stage, where one's perception is integrated with information from long-term memory, and an internal (mental) representation of the display is generated. Based on this representation, users can then make action plans to use the information or interact with the display.
(Schöttl & Lindemann, 2015)	Technical System Complexity Potential	Function of system properties (number of elements, variety, degree of interconnectedness, variety of interfaces, number of interfaces) and system changes (changes in number, in variety of elements, in degree of interconnectedness, in number of interfaces, in variety of interfaces, of elements) [...] The complexity potential of the system, a person is interacting with, is the first and most important building block of the quantification approach. This factor is modeled by established system properties from

		Systems Engineering and System Theory literature. These properties are used to describe the static state of a system, which represents the basic component of quantification. The description of system dynamics is based on the identical system properties and represents systems changes
	Task Complexity Potential	Interaction properties for the type of interaction of the task include a situational and a basic fact component, which are universally valid. The interaction properties are built up from particular influencing parameters (factors) which describe the interaction between the person and the system in the context of perceived complexity [...] tasks are characterized by: Time pressure, demand, content, level of detail, responsibility, number of subordinated employees, work flow, and instructions. Interfaces are characterized by: Localization of communication partners, language of communication partners, and number of communication partners. [...] Interaction classification: easy, ordinary or highly demanding.
	Perceived Complexity	We use the term “mental flexibility”, characterizing the ability to flexibly adapt to volatile problems independent from knowledge of the system and methodologies. In conclusion, experience and mental flexibility are the relevant, influencing parameters to characterize the perception of complexity when interacting with the system. A five-step graded scale, ranging from “very low” to “very high” is used to mathematically model both parameters
(Crawley et al., 2015)	Apparent Complexity	Apparent complexity is akin to how "complicated" the system is; how hard it is to understand. Has the potential to scale up with the number of features of robust functionalities and/or the number of objects.
	Essential Complexity	Required minimum complexity to achieve a given function or performance.
	Gratuitous Complexity	Uncertainty involved in achieving a task accordingly with functional requirements.
(Suh, 2005)	Real Complexity	(Time independent) Uncertainty involved in achieving a task accordingly with functional requirements.
	Imaginary Complexity	(Time independent) Our lack of understanding about the system design, behavior, architecture.
	Combinatorial Complexity	(Time dependent) Performance drift away from requirements with time, makes it increasingly difficult to satisfy functional requirement and to make the best decision, and also decreases our ability to predict system performance.
	Periodic	(Time dependent) When a set of requirements is cyclic, allowing for periodic

	Complexity	"re-initialization", then combinatorial complexity is mitigated with each new cycle. We are able to predict system behavior in the short term (i.e. shorter than cycle length)
(Histon & Hansman, 2002; Histon, 2008)(Histon & Hansman, 2002; Histon, 2008)	Situation/System Complexity	Intrinsic complexity of the air traffic situation or real system. It is an objective and measurable property of the air traffic situation being controlled. Metrics will encompass properties and characteristics of the situation (e.g. distribution of aircraft, underlying airspace structure). The system complexity represents the complexity that is inherent within the real system, independent of an operator controlling it.
	Cognitive Complexity	Cognitive complexity is understood to be the complexity of the working mental model(s) used by a controller to control an air traffic situation and perform a given task. Many different factors will influence the working mental model, and hence the cognitive complexity including the controller's task, their mental models and strategies, as well as factors such as fatigue and stress.
	Perceived Complexity	Perceived complexity is the externalization of the controller's self-reported, or internal perception, of the cognitive complexity.
(Martin, 2004)	Internal Complexity	Complexity of the complex system itself (radar). Metric takes into account the number of links, the number of elements, the function and hierarchy of elements. Decomposed in a scale and a link complexity metrics.
	Interface Complexity	Tension that reigns on the "membrane" that separates the system from its environment. This tension mainly depends upon two variables, which are the resistance of the system and the pressure of the environment. When both the pressure and the resistance are high, interface complexity is high. Metric based upon the information content of the probability of failure of the system under its normal conditions of use and measures how hard it is to achieve what the system achieves.
	External Complexity	Complexity of the environment of the system, i.e. the large-scale complex system, in which the complex system is embedded. Deals with the risk configuration of large-scale systems emphasizing the reliability and the tendency to catastrophe of the system. Metric1 = ratio of (risk associated with failures of higher than average magnitude) to (risk associated with failures of lower than average magnitude). Matric2 = ratio of (risk associated with failures in the top 10% of magnitude) to ((risk associated with failures in the lower 10% of magnitude)
(Hilburn, 2004)	Cognitive Complexity	A subjective phenomenon that is not directly observable. The relationship between complexity and workload is an indirect one that is highly mediated

		by the influence of many individual characteristics.
(Read, 2008)	Intrinsic Complexity	"Arises from the need for increased product capability [...] This increased capability requirement is the result of new technologies that are able to support greater functionality, a greater speed of operation, increased automation and an improved accuracy, while also reducing support costs, increasing sustainability and system reliability" [...] "As product functionality, intricacy and interoperability increases, often the intrinsic complexity of these products also increases." [...] "Intrinsic complexity may, potentially, lead to an increase in emergent properties within the systems, something which may or may not be desirable".
	Induced Complexity	"Induced complexities occur as a result of the development process, and have a potential to increase the effort, and perhaps the cost, required to develop a system". "Not all induced complexity is detrimental to product development, and in some cases enhancing the complexity of a system reduces the workload and development times" (e.g. introduction of products taken off the shelf)

10 References

Airport Collaborative Decision Making Implementation Manual Version 4. (2012). EUROCONTROL. Retrieved from http://www.eurocontrol.int/sites/default/files/publication/files/2012-airport-cdm-manual-v4.pdf

Bankes, S. (1993). Exploratory Modeling for Policy Analysis. *Operations Research*, *41*(3), 435–449.

Bonaceto, C., Moertl, P., Estes, S., & Burns, K. (2005). Naturalistic Decision Making in the Air Traffic Control Tower : Combining Approaches to Support Changes in Procedures. In *The International Conference on Naturalistic Decision Making*. Amsterdam, The Netherlands.

Bouarfa, S. (2015). *Agent-Based Modelling and Simulation of Safety and Resilience in Air Transportation (Doctoral dissertation)*. TU Deft, Delft University of Technology.

Bouarfa, S., Blom, H. A., Curran, R., & Everdij, M. H. (2013). Agent-based modeling and simulation of emergent behavior in air transportation. *Complex Adaptive Systems Modeling*, *1*(1), 1–26. http://doi.org/10.1186/2194-3206-1-15

Broniatowski, D. A. (2014). *Flexibility, Complexity, and Controllability in Large Scale Systems (Doctoral dissertation)*. Stevens Institute of Technology.

Brooks, F. P. (1962). Architectural Philosophy. *Planning a Computer System-Project Stretch*, 5–16.

Cardoso, J. (2005). How to Measure the Control-flow Complexity of Web Processes and Workflows. *Workflow Handbook*, 199–212.

Challenges of Growth 2013, Summary Report. (2013). EUROCONTROL. Retrieved from http://www.eurocontrol.int/sites/default/files/content/documents/official-documents/reports/201307-challenges-of-growth-summary-report.pdf

Corrigan, S., Märtensson, L., Kay, A., Okwir, S., Ulfvengren, P., & McDonald, N. (2014).

Preparing for Airport Collaborative Decision Making (A-CDM) implementation: an evaluation and recommendations. *Cognition, Technology and Work*, *17*(2), 207–218.

Crawley, E., Cameron, B., & Selva, D. (2015). *Systems Architecture: Strategy and Product Development for Complex Systems*. Prentice Hall Press.

Crawley, E., de Weck, O., Eppinger, S., Magee, C., Moses, J., Seering, W., … Whitney, D. (2004). *The Influence of Architecture in Engineering Systems (monograph)*.

Davis, P. K., & Anderson, R. A. (2003). Improving the composability of Department of Defense models and simulations, Rand Corporation Report. *Santa Monica CA: RAND Corporation.*

De Weck, O. L., Roos, D., & Magee, C. L. (2011). (Re)Thinking about Systems. In *Engineering systems: Meeting Human Needs in a Complex Technological World* (pp. 45–63). MIT Press.

Dodder, R. S. (2006). *Air quality and intelligent transportation systems: Understanding Integrated Innovation, Deployment and Adaptation of Public Technologies (Doctoral dissertation)*. Massachusetts Institute of Technology.

Dolinskaya, I. S., Shi, Z. E., Smilowitz, K. R., & Ross, M. (2011). Decentralized approaches to logistics coordination in humanitarian relief. In *IIE Annual Conference. Proceedings* (p. 1). Institute of Engineers-Publisher.

Drezner, J. A. (2009). *Competition and innovation under complexity*. (No. RAND-RP-1386). RAND CORP ARLINGTON VA NATIONAL SECURITY RESEARCH DIV.

Dwyer, M. M. (2014). *The Cost of Jointness : Insights from Environmental Monitoring Systems in Low Earth Orbit (Doctoral dissertation)*. Massachusetts Institute of Technology.

Eppinger, S. D., & Browning, T. R. (2012). *Design structure matrix methods and applications*. MIT Press.

Flood, R. L., & Carson, E. R. (1993). *Dealing with Complexity: An Introduction to the Theory and Application of Systems Science*. Springer Science & Business Media.

García, I., Valero, M., Prats, X., & Delgado, L. (2014). Agent-based simulation framework for

airport collaborative decision making. In *Proceedings of the 6th International Congress on Research in Air Transportation (ICRAT)*.

Glazner, C. G. (2009). *Understanding enterprise behavior using hybrid simulation of enterprise architecture (Doctoral dissertation)*. Massachusetts Institute of Technology.

Hall-May, M., Surridge, M., & Nossal-Tüyeni, R. (2011). Resilient critical infrastructure management with a service oriented architecture: A test case using airport collaborative decision making. *International Journal of Applied Mathematics and Computer Science*, *21*(2), 259–274.

Ham, D., Park, J., & Jung, W. (2011). A framework-based approach to identifying and organizing the complexity factors of human-system interaction. *Systems Journal, IEEE*, *5*(2), 213–222.

Hansman, R. J., & Davison, H. J. (2001). The effect of shared information on pilot/controller and controller/controller interactions. In *New Concepts and Methods in Air Traffic Management* (pp. 143–159). Springer-Verlag Berlin Heidelberg.

Highfield, R., & Coveney, P. (1996). *Frontiers of complexity: The search for order in a chaotic world*. Ballantine Books.

Hilburn, B. (2004). Cognitive complexity in air traffic control: A literature review. *EEC Note*, *4*(04).

Histon, J. (2008). *Mitigating Complexity in Air Traffic Control: The Role of Structure-Based Abstractions (Doctoral dissertation)*. Massachusetts Institute of Technology.

Histon, J., & Hansman, R. J. (2002). *The Impact of Structure on Cognitive Complexity in Air Traffic Control*. (Report No. ICAT-2002-4). MIT International Center for Air Transportation.

Histon, J., Li, L., & Hansman, R. J. (2010). Airspace structure, future ATC systems, and controller complexity reduction. In *Digital Avionics Systems Conference (DASC), 2010 IEEE/AIAA 29th* (p. 4–A).

Jahre, M., & Jensen, L.-M. (2010). Coordination in humanitarian logistics through clusters. *International Journal of Physical Distribution & Logistics Management*, *40*(8/9), 657–674.

Kelton, W. D., & Law, A. M. (2000). *Simulation Modeling and Analysis*. Boston: McGraw-Hill.

Key to Innovation Integrated Solution: Multimodal Personal Mobility. (2013). European Commission. Retrieved from https://eu-smartcities.eu/sites/all/files/Multimodal personal mobility january.pdf

Klein, G., Moon, B., & Hoffman, R. R. (2006). Making Sense of Sensemaking 1: Alternative Perspectives. *IEEE Intelligent Systems*, *4*, 70–73.

Kreimeyer, M. (2011). Aggregate views to manage complex dependency models. *International Journal Product Development*, *14*(1-4), 144–164.

La Tour, P. A., & Hastings, D. E. (2015). Combining System Dynamics with Tradespace Exploration to Explore Future Space Architectures. In *66th International Astronautical Congress*. Jerusalem, Israel.

Lessard, D., Sakhrani, V., & Miller, R. (2014). House of Project Complexity – Understanding Complexity in Large Infrastructure Projects. *Engineering Project Organization Journal*, *4*(4), 170–192.

Li, K., & Wieringa, P. A. (2000). Understanding perceived complexity in human supervisory control. *Cognition, Technology & Work*, *2*(2), 75–88.

Magee, C., & de Weck, O. (2004). Complex system classification. In *Fourteenth Annual International Symposium of the International Council on Systems Engineering (INCOSE)* (pp. 1–18).

Maier, M. (2009). *The Art of Systems Architecting*. CRC press.

Maier, M. (2015). The Role of Modeling and Simulation in System of Systems Development. In *Modeling and Simulation for System of Systems Engineering Applications* (pp. 11–41). John Wiley & Sons.

Martin, P.-A. J. Y. (2004). *A framework for quantifying complexity and understanding its sources: application to tow large-scale systems (Doctoral dissertation)*. Massachusetts Institute of Technology.

Mathieu, J., James, J., Mahoney, P., Boiney, L., Hubbard, R., & White, B. (2007). Hybrid systems dynamic, petri net, and agent-based modeling of the air and space operations center. In *INCOSE Symposium. Systems Engineering: Key to Intelligent Enterprises*.

Mathieu, J., Melhuish, J., James, J., Mahoney, P., Boiney, L., & White, B. (2007). Multi-Scale Modeling of the Air and Space Operations Center. In *Symposium on Complex Systems Engineering, The Rand Corporation. http://cs.calstatela. edu/wiki/images/d/db/White.pdf*.

Mehta, V., Reynolds, T. G., Ishutkina, M. A., Joachim, D., Glina, Y., Troxel, S. W., … Evans, J. E. (2013). *Airport Surface Traffic Management Decision Support : Perspectives Based on Tower Flight Data Manager Prototype (No. ATC-398)*.

Moshtari, M., & Gonçalves, P. (2012). Understanding the drivers and barriers of coordination among humanitarian organizations. In *23rd Annual Conference of the Production and Operations Management Society*.

Mostashari, A. (2010). Sociotechnical Systems: A Conceptual Introduction. In *Socio-Technical Networks: Science and Engineering Design*. CRC press.

Murman, E. M., & Allen, T. J. (2003). Engineering Systems: An Aircraft Perspective. Retrieved from https://esd.mit.edu/symposium/pdfs/papers/murman.pdf

Newman, R. A. (1999). Issues in defining human roles and interactions in systems. *Systems Engineering*, *2*(3), 143–155.

Okwir, S., & Correas, A. (2014). Collaborative Decision Making (CDM) in Airport Surface: Europe vs USA implementations, challenges and best practices. *Integrated Communications, Navigation and Surveillance Conference (ICNS)*, G2–1–G2–15.

Perrow, C. (1999). *Normal accidents : living with high-risk technologies*. Princeton, N.J. : Princeton University Press.

Read, C. (2008). *Complexity characteristics and measurement within engineering systems (Doctoral Dissertation)*. Loughborough University.

Reindorp, N., & Wiles, P. (2001). *Humanitarian coordination: Lessons from recent field experience*. London: Overseas Development Institute.

Reymondet, L., Rhodes, D. H., & Ross, A. M. (2016). Considerations for Model Curation in Model-Centric Systems Engineering. In *Systems Conference (SysCon), 2016 10th Annual IEEE*. IEEE.

Rhodes, D. H., & Ross, A. M. (2015). Interactive Model-Centric Systems Engineering (IMCSE) Technical Report SERC-2014-TR-048-2.

Richards, M. G., Shah, N. B., Hastings, D. E., & Rhodes, D. H. (2007). *Managing complexity with the department of defense architecture framework: Development of a dynamic system architecture model (Doctoral dissertation)*. Massachusetts Institute of Technology.

Righi, A. W., Wachs, P., & Saurin, T. A. (2012). Characterizing complexity in socio-technical systems: A case study of a SAMU Medical Regulation Center. *Work*, *41*(Supplement 1), 1811–1817.

Ross, A., Fitzgerald, M., & Rhodes, D. (2016). Interactive Evaluative Model Trading for Resilient Systems Decisions. In *14th Conference on Systems Engineering Research*. Huntsville, AL.

Rouse, W. B. (2015). *Modeling and Visualization of Complex Systems and Enterprises: Explorations of Physical, Human, Economic, and Social Phenomena*. John Wiley & Sons.

Rouse, W. B., & Bodner, D. A. (2013). Multi-Level Modeling of Complex Socio-Technical Systems (Report CCSE-2013--01). Hoboken, NJ: Stevens Institute of Technology. *Center for Complex Systems and Enterprises*.

Schöttl, F., & Lindemann, U. (2015). Quantifying the Complexity of Socio-Technical Systems–A Generic, Interdisciplinary Approach. *Procedia Procedia Computer Science*, *44*, 1–10.

Senge, P. (1990). *The Fifth Discipline: The Art & Practice of The Learning Organization*. London: Century Business.

Sheard, S. A. (2012). *Assessing the impact of complexity attributes on system development project outcomes (Doctoral Dissertation)*. Stevens Institute of Technology.

Simon, H. (1991). *The Architecture of Complexity*. Springer US.

Snowden, D. J., & Boone, M. E. (2007). A leader's framework for decision making. *Harvard Business Review*, *85*(11), 68.

Spickermann, A., Grienitz, V., & Heiko, A. (2014). Heading towards a multimodal city of the future?: Multi-stakeholder scenarios for urban mobility. *Technological Forecasting and Social Change*, *89*, 201–221.

Sterman, J. D. (1991). A skeptic's guide to computer models. *Managing a Nation: The Microcomputer Software Catalog*, *2*, 209–229.

Sterman, J. D. (2002). System Dynamics: systems thinking and modeling for a complex world. In *Proceedings of the ESD Internal Symposium*.

Stibe, A. (2015). *Persuasive Cities for Sustainable Wellbeing [Video file]*. Retrieved from https://www.youtube.com/watch?v=Hy23R1GIOsQ

Study on Urban Transport Development. (2000). The World Bank.

Suh, N. P. (2005). *Complexity: theory and applications*. Oxford University Press on Demand.

Sussman, J. M. (2002). Collected Views on Complexity in Systems. In *Proceedings of the Engineering Systems Division (ESD) Internal Symposium* (pp. 453–478).

Sussman, J. M., Dodder, R. S., McConnell, J. B., Mostashari, A., & Sgouridis, S. (2007). The CLIOS Process: a user's guide. *Course Materials for ESD. 04J Frameworks and Models in Engineering Systems, Spring 2007*.

United Nations Department of Economic and Social Affairs, P. D. (2014). *World Urbanization*

Prospects: The 2014 Revision, Highlights (ST/ESA/SER.A/352).

Van Hemel, S. B., MacMillan, J., & Zacharias, G. L. (2008). *Behavioral Modeling and Simulation: From Individuals to Societies*. National Academies Press.

Whitcomb, C. A., Auguston, M., & Giammarco, K. (2015). Composition of Behavior Models for Systems Architecture. In *Modeling and Simulation Support for System of Systems Engineering Applications* (p. 361). John Wiley & Sons.

Xing, J. (2007). *Information Complexity in Air Traffic Control Displays*. Springer Berlin Heidelberg.

Xing, J., & Manning, C. A. (2005). *Complexity and Automation Displays of Air Traffic Control: Literature Review and Analysis*. (No. DOT/FAA/AM-05/4). FEDERAL AVIATION ADMINISTRATION OKLAHOMA CITY OK CIVIL AEROMEDICAL INST.

Zulkepli, J., Eldabi, T., & Mustafee, N. (2012). Hybrid Simulation for Modelling Large Systems: An Example of Integrated Care Model. In *Simulation Conference (WSC), Proceedings of the 2012 Winter. IEEE* (pp. 1–12).