Towards an MDA-Oriented UML Profile for Critical Infrastructure Modeling

Ebrahim Bagheri and Ali A. Ghorbani
Faculty of Computer Science, University of New Brunswick
Fredericton, N.B., Canada
{e.bagheri, ghorbani}@unb.ca

Abstract

Infrastructures are networks of highly complex systems that can be classified as socio-technical organisms with hidden consciousness. The hidden consciousness of these types of systems lies beyond their definition. Although these systems are structurally independent of any outside component, but collaborate synergistically to provide their services to the end customer. The interdependencies between these complex systems bring about sophisticated and unpredictable outcomes. In this paper we propose a platform independent metamodel for critical infrastructures. The metamodel precisely defines every aspect of an infrastructure through clear syntactical and semantic definition of existing concepts and relationships. The Platform independent model (PIM) has been defined as a UML profile (UML-CI) and serves as one of the first steps towards building an agent based simulation environment.

Index Terms— Critical Infrastructure, Modeling and Simulation, UML Profiles.

1. Introduction

Critical infrastructures are a network of independent, mostly privately-owned, man-made systems and processes that function collaboratively and synergistically to produce and distribute a continuous flow of essential goods and services [1]. These highly complex systems can be classified as socio-technical organisms with hidden consciousness. They can undergo aging in their lifecycles and hence experience different operational states. Complexity is an inherent feature of these systems that stems from the nature of their responsibilities. These complex systems are mostly in charge of catering specialized services to a wide range of different parties. The type of the collective services that is provided through these systems is so immense that the ordinary lives of all civilians are tightly coupled with their functionality. It is hence supposed that these sorts of services should be ubiquitous, reliable, cheap and conveniently accessible [2]. Ubiquity implies that all civilians will have an equal opportunity of accessing these resources from whatever location that they desire. The penetration rate of many of these services is now to a great extent acceptable in many western countries. The Internet as the most critical infrastructure in the information technology field has had a great penetration rate in many countries. The significant growth of Internet can be taken as a sign to estimate the pervasiveness of other infrastructures that serve as a basis for the Internet operation. Electricity is probably the most significant infrastructure that affects the Internet. It can be taken for granted that the penetration rate of electricity is far beyond these statistics.

Electricity, Telecommunication, Water and Sewage, and Transport are the most prevalent infrastructures that can be considered as omni-present in many developed or even developing countries. Reliability as another feature of infrastructures indicates that the state of infrastructure operation should not be dependent on stochastic real world incidences. It is therefore assumed that the same services will be delivered with the same quality level regardless of the location, time, or the customer. These services should be economically affordable for the average civilian as they play a major role in everyone’s lives. It should also be convenient for new customers to join the network of the infrastructure’s customers and start receiving service.

In this paper we propose an extension to the Unified Modeling Language in order to clearly define the different aspects of an infrastructure. The approach taken in defining the UML extension has been towards establishing a fair basis for multiagent modeling and simulation of critical infrastructures. The rest of the paper is organized as follows. In Section 2, we will explain the structure of the proposed UML-CI profile. The paper will conclude in Section 3 with a brief insight in future work.
2. UML-CI: A Profile to Support Critical Infrastructure Modeling

Due to the inherent complexity and unpredictable collective behavior of critical infrastructures, it is an important issue to pursue research to understand their internal structure and external relationships. Most of the research in this field has been focused on investigation of a specific issue or providing a very domain specific simulation tool or model. The best classification of infrastructure interdependencies has been provided in [4]. Although this type of classification gives much insight into infrastructure operations and interdependencies, but does not provide any guidelines of how infrastructures are internally organized. Other research such as [3, 5] provide a very high level classification of each infrastructure’s internal structure. This type of classification does not provide any details on how infrastructures are interconnected or even how they may be structured. In this section we provide a very detailed description of a typical infrastructure and their most common features.

The proposed model is a platform independent model that extends the Meta Object Facility (MOF). It has been developed based on UML profiles and offers details on how an infrastructure is organized. In this model, different structural elements of an infrastructure’s organization have been differentiated. Suitable constraints have been attached to the elements to control the infrastructure model’s correct composition and well-formedness. The common design theme that had been closely observed was that an infrastructure has four structural layers. The four layers follow the four tiered approach that had been introduced in [5] which are namely the Service Layer, Management Layer, Cyber Layer, and the Physical Layer. The other important fact that has been perfectly incorporated into the model is that different means for interdependency modeling based on the dimensions introduced in [4] have been devised in the model. An example of this is the presence of the Policy_Maker and Regulation metaclasses. These two metaclasses have been placed into the model to allow a detailed infrastructure design through the application of public policies, legal concerns, security issues, and government decisions.

In the following sub-sections we will give a descriptive sketch of the UML-CI profile. The extended stereotypes will be clearly explained and the applied constraints will be introduced. Some of the most important constraints will be provided in Object Constraint Language (OCL). An abstract formation of a typical infrastructure will be also explained to serve as a guideline for designing infrastructure systems. To transform the PIM model into a general PSM, more details on models of inter-Infrastructure Contract Establishment, Contract Conformance Enforcement models, and Infrastructure Fault Tolerance should be clarified. We will not go into the details of a PSM design since it is well out of the scope of this paper, but we will attempt to clarify the issues that exist in its design process.

A. General Architecture

The base metaclasses that have been used in this approach are the Class and KernelAssociation. The Class metaclass has been extended to provide the basis for modeling structural elements of an infrastructure. On the other hand KernelAssociation has been adopted to handle different sorts of relationships between the extended metaclasses. The profile consists of 17 metaclasses that have been extended from Class and 13 KernelAssociation extensions.

The structural components are designed in a form to enable the designer to manipulate most of the dimensions of an infrastructure’s operation environment. To elaborate on the different aspects of these metaclasses we explain how the six dimensions of infrastructure interdependency can well fit into the proposed UML-CI profile. The first dimension of Rinaldi’s categorization deals with different types of interdependencies. Physical and Cyber relationships can be modeled through the use of the abstract Asset metaclass. This metaclass can be generalized into three other metaclasses that are Non_Physical_Service, Physical_Asset, and Cyber_Asset. All these three different types of assets can be used to model the different types of infrastructure inputs or outputs. Each infrastructures input or output type should be of Asset type. Having used an Asset Trade model for simulating the relationship between two infrastructures, the Asset abstract metaclass can handle physical and cyber interdependencies. For geographical interdependencies the Point_of_Operation metaclass has been devised. Each infrastructure instance can only operate within the borders of one country that has been modeled as the Government metaclass. Each government has a set of boundaries that shows its borders of jurisdiction. The internal government territory area can be shown through the use of several Point_of_Operations. The Government_Territory metaclass is an aggregation of many different Point_of_Operations. Geographical interdependency can hence be modeled through the use of this metaclass. Logical interdependency can only be revealed through the interpretation of real simulations and is therefore irrelevant to the modeling context.

The next dimension of infrastructure operation is its operation environment that includes economic and business opportunities, public policies and legal concerns, security issues, and government decisions. All
of these aspects can be abstracted into two high level metaclasses called Actor and Regulation. The government, public policy makers and many other role-players can be considered as Actors. Actor is an abstract metaclass from which the Policy_Maker metaclass can inherit. Policy makers can enforce Regulations on different Tasks or Assets performed or manipulated by a System which is the main constituent part of an Infrastructure. The coupling and response behavior of two interdependent infrastructures can be modeled through their dependency types.

Different types of failure that can be incorporated into an infrastructure design are an outcome analysis measure and are out of the scope of our modeling profile. Failure models should also be included in the PSM. One of the ways that failure can be represented is through the asset trade model. If the required inputs of an infrastructure are not provided at the right time, the infrastructure will not be able to keep up with its guaranteed products. This causes cascading failures. Other types of failures can be simulated in a similar fashion.

An infrastructure is a complex network of socio-technical systems, each of which pursues a predefined goal. For this reason we can model every infrastructure as a set of System metaclasses. Each System metaclass achieves its goals by performing certain Tasks. In our modeling profile the organizational and operational characteristics of an infrastructure is represented through System and Task metaclasses. Spatial characteristics are modeled by the Government_Territory and Point_of_Operation metaclasses.

The very important infrastructure characteristic is its temporal operational status. In our modeling profile we view time as an “aspect” that can encompass different concepts of the profile. For this reason the definition of this concept has been shifted to the PSM design phase as an inbuilt feature of the simulation environment. Much similar to failure types, infrastructure state of operation should be incorporated into the PSM model and can only be clarified at specific infrastructure specification and hence has been abstracted out in this profile.

To give a better understanding of the UML-CI profile we provide a top to bottom explanation of the profile here. Each infrastructure instance can only operate inside a specific country which is ruled by its government. For this reason every defined infrastructure in this model would be administered by the corresponding government. On the other hand each government has a limited physical space under his control called the government territory. The government territories are hence governed by that government. Each territory encompasses many geographical points known as the points of operation.

Infrastructures are built from different systems. Each system itself fulfills as set of tasks to reach its goals. Every task requires a set of inputs to perform its process, for this reason it exploits operational requirements. Every task will also generate a set of manufactured products. To allow an infrastructure to spread all over the government’s territory and at the same time enable infrastructures’ geographical interdependency control, each task can specify the locations in which it can perform at. Therefore the assets that are either produced or consumed by a task should be accessible at specific locations. On the other hand there are many different actors that can affect the infrastructure’s decisions or operations such as policy makers. Regulations are the means to control an infrastructure’s operation and are developed by the policy makers. Each task can be controlled by a regulation. Resource access patterns and consumptions are also restricted by regulations.

B. Well-Formedness Constraints

The models in a UML profile can only show the abstract syntax of the overall design. Detailed composition restrictions have to be applied as audits to any abstract syntax to enable high level designers to correctly arrange their designed model. OCL is a language that can express additional and necessary information about the models and other artifacts used in metamodelling, and should be used in conjunction with UML diagrammatic models. We have devised a set of rules to control the overall infrastructure metamodelling procedure and provide the basis for the construction of a well-formed metamodel. In this section we will explain the purpose of each of these constraints. Some of the constraints have been expressed in OCL to allow a deeper understanding, however the rest have been omitted because of space limitations and due to their relative similarity.

R0: Defined Association types should only be applied to their respective source and destination metaclasses. For example the Built_From meta-association can only be applied from the Infrastructure metaclass to System:

Context uml20::kernel::KernelAssociation
inv: stereotypes->includes ('Built_From') implies
  supplier.stereotypes->includes ('System') and client.stereotypes->includes ('Infrastructure')

R1: Each Government only Governs one Government_Territory.

Context uml20::classes::Class
inv: self->oclIsTypeOf(Government)
implies associations-> select(oclIsTypeOf(Governs)).supplier-> select(oclIsTypeOf(Government_Territory))-> size()=1

R2: Every Infrastructure instance can only be administered by one Government.
R3: Every Task should at least have one input as Operational_Requirement and one output as Manufactured_Product:

Context uml20::classes::Class

inv: self->oclIsTypeOf(Task)
implies
(self.associations-> select(oclIsTypeOf(Generates))->size()> 0)
and
(self.associations-> select(oclIsTypeOf(Exploits))->size()> 0)

R4: Each Task can only Exploit Assets that is Accessible at its Point_of_Operation.
R5: Each Task can only Exploit or Generate Assets that are Owned by its parent System:

Context uml20::classes::Class

inv: self.oclIsTypeOf(Task)
implies
self.associations-> select(oclIsTypeOf(Fulfils)).oclAsType(System).associations-> select(oclIsTypeOf(Owns)).supplier-> includesAll(self.associations-> select(oclIsTypeOf(Exploits)).supplier)
and
self.associations-> select(oclIsTypeOf(Fulfils)).oclAsType(System).associations-> select(oclIsTypeOf(Owns)).supplier-> includesAll(self.associations-> select(oclIsTypeOf(Generates)).supplier)

R6: Each Asset should belong to a System’s parent Infrastructure in order to be Exploited by that System’s Tasks.

R7: If an Asset is Restricted by a Regulation, The corresponding Task should also be Controlled.
R8: All Association metaclasses are directed.
R9: Two different Systems cannot be the parent of a Task.

Context uml20::classes::Class

inv: self.oclIsTypeOf(Task)
implies
self.associations-> select(oclIsTypeOf(Fulfils))-> size() = 1

R10: Each System is only used in the construction of one Infrastructure.
R12: Every Regulation is only Developed by one Policy Maker.

To show how these metaclasses can form a unique infrastructure organization a sample model has been created using the metaclasses introduced in UML-CI. This model does not intend to create a sample infrastructure representation and is only used to show how different metaclasses that are present in the PIM form a whole. As it can be seen in Figure 1, abstract metaclasses like Asset or Actor that will not be a part of the real modeling process have also been placed on the diagram. We call the diagram a Symbolic Critical Infrastructure Pattern.

3. Conclusions and Future Work

In this paper we have devised a UML profile that supports the structural design of an infrastructure. The design process follows an MDA approach. The developed profile is completely platform independent and is based on the six dimensions of infrastructure interdependency introduced in [4]. The PIM is composed of 30 metaclasses and 12 constraints. A Symbolic Critical Infrastructure Pattern has also been set up to allow better understanding of the composition of the introduced metaclasses.

We strongly believe that the proposed profiling scheme brings about a better understanding and a clearer picture of the infrastructure internal arrangement and external interdependencies. To form a complete modeling and simulation cycle, we plan to complement this PIM with a platform specific model. The intended PSM will be based on an agent based architecture. There are a few simulation specific issues that need to be investigated and clarified. Issues such as the temporal characteristics of the model, different types of failure, models of inter infrastructure contract establishment, asset transmission, contract agreement and regulation conformance, infrastructure wear out and recovery models are some of the most important issues that should be elucidated.
4. References


