Policy Based Management for Critical Infrastructure Protection

Gwendal Le Grand, Franck Springinsfeld, Michel Riguidel
ENST, École Nationale Supérieure des Télécommunications, Paris, France
{gwendal.legrand, franck.springinsfeld, michel.riguidel}@enst.fr

This work has been realized in the context of the ACIP project [ACIP], a project funded by the European Commission under the "Information Society Technology" Programme.

Abstract: Our current societies are fully dependent on large complex critical infrastructures (LCCIs). These LCCIs are large scale distributed systems that are highly interdependent, both physically and in their greater reliance on the information infrastructure, which logically introduce vulnerabilities. Failures, accidents, physical or cyber attacks can provoke major damages which can proliferate by cascading effects and then can severely affect a part or the whole society. This article aims at providing a solution for enhancing dependability and survivability of such systems by developing new models, methodologies and tools. An assessment of vulnerabilities of existing infrastructures is achieved using canonical architectures. Then, we extract abstract security policies that are implemented using a policy based management approach.

1 Introduction and Background

The economy and security of Europe are increasingly dependent on a spectrum of critical infrastructures, which can be broadly grouped in the following five domains: Information and Communications, Energy (Electrical Power and Oil and Natural Gas Production and Storage), Transportation, Banking and Finance, Vital Human Services (Water and Food Supply Systems, Emergency Services, Government Services).
All the above critical infrastructures are highly interdependent, both physically and in their greater reliance on the information infrastructure. This trend has been accelerating in recent years with the explosive growth of information technology and shows no sign of abating.

"A bounded system is one in which all of the system's parts are controlled by a unified administration and can be completely characterized and controlled". An unbounded system is "characterized by distributed administrative control without central authority, limited vision beyond the boundaries of local administration, and lack of complete information by the network". "An unbounded network can be composed of bounded and unbounded systems connected together in a network".[Ro99]
Because of their interdependencies and their increasing reliance on open systems such as the public switch telecommunications network and the Internet, all these critical infrastructures constitutes an unbounded system where faults may occur and proliferate in a severe way and where security represents a real challenge and requires new methodologies and tools.

Potential threats to the normal functioning of these infrastructures are both natural ("Murphy's Law and Mother Nature") and man-made. Individual outages can be serious enough, but this growing degree of interconnectedness can make possible a whole new scale of synergistic, nonlinear consequences.

1.1 Description of a CI

A CI comprises three components that interact:

- **Administration (Adm)** that includes the human (decision-makers and workforce), economic, regulation, etc. aspects,
- **Physical (Phy)** corresponds to the material aspect of the resource supply system,
- **Information System (IS)** corresponds to the information system of the infrastructures (e.g. Supervisory Control And Data Acquisition (SCADA) …), information technology for business systems, e-commerce …

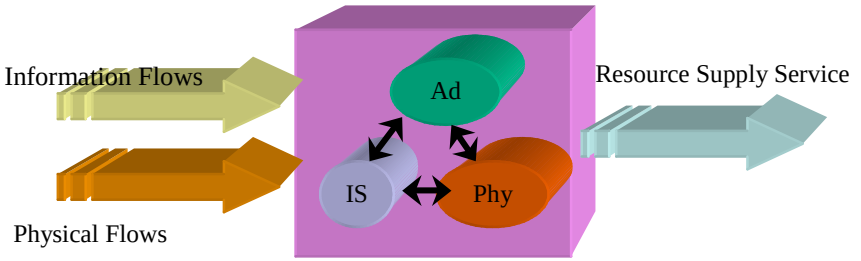

Figure 1. Description of a CI

1.2 Key Properties of CIs

There are five key properties of critical infrastructures that must be understood before one could formulate a methodology for building a CI system to support their protection. These properties are fully exposed in the first deliverable of the first ACIP project work package [1]. They are quickly presented hereafter:

1. CIs exhibit strong, mutual **dependencies**
2. CIs are largely **owned and operated by the private sector**.
3. CIs are becoming increasingly **IT-dependent** in order to accelerate information circulation and reduce the operating cost.
4. As a consequence of the increasing globalization of commerce, CIs are becoming **international**.
5. The advent of "**just-in-time**" business practices has reduced margins for error in infrastructures.

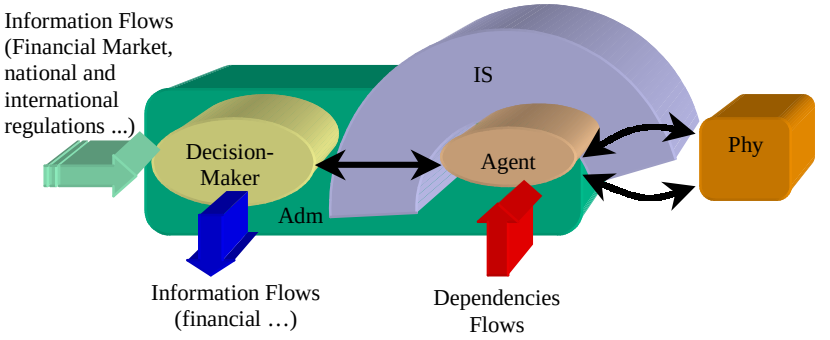

Figure 2. Description of the Administration component

1.3 CIS hierarchies

CIS can be classified hierarchically in four levels: from the top to the bottom, “System of Systems” (government, economy, society), “Systems interdependencies” (compound of critical infrastructures), “Individual Systems” (Telecommunication, Transportation, Energy…), “Technical Components” (computer, power lines…).
Two global levels can be extracted from this classification:

- the high level corresponds to the **business level** and includes the two first levels.
- the low level corresponds to the **technical level** and includes the two last levels.

The remainder of this paper is organized as follows: we first supply basic considerations about the critical infrastructures field. Then LCCI-related definitions are presented in section 2. It results a collection of requirements for models to investigate vulnerability of single components of a CIS and to model impacts, disruptions, interdependencies, and cascading effects within each of the main CIS in section 3. Section 4 presents the characteristics of the security models and section 5 proposes an approach for CI protection, based on Policy Based Management.

2 Definitions

In the end of this document, we give references of documents which well define the following commonly-used definitions: *risks* [Ro99], *dependency* [RPK01], the different *interdependencies* and their consequences in terms of risk (*cascading and escalating effects*) [Ro99] [RPK01], *dependability* and all the notions introduced in the *dependability tree* [PD01] [Ba02], *survivability* [Ro99].

We add the definitions of *criticality and degree of criticality.*

The ***criticality*** of a system refers to several aspects of this system such as its sensitivity (i.e. its capability to move from an equilibrium state to a stressed state because of disturbances), impact of fault (impacts on the system, impacts on population …)….
Close to the notion of *criticality*, the ***degree of criticality*** is an indicator of the state of a system and depends on the degrees of criticality of the components of this system, its morphology, the time (e.g. congestion peak in certain moments in certain locations within a road transport system) and so on.

3 Requirements of the system

The goals of a security management model are to be able to foresee the development flaws, detect anomalous behaviors to proactively manage the system in order to prevent serious problems, install prevention measures, and reactively control the system by making adjustments in response to changes (that may be sudden as when following an attack) within the system or its environment.
Even if it is almost impossible to prevent attacks 100 percent, it is really important to be able to act quickly within the system to stop a potential proliferation of the problem. Consequently, two correlated works of modeling can be distinguished: one concerning CIs and one concerning security management.

Therefore, the basic requirements of the system are motivated by the following security functional requirements: prediction and scenario simulation (development, proactive management, etc.), prevention, monitoring (global view, reactive and proactive management, real-time), distributed intelligence and autonomy.

In [Ro99], requirements for CI modeling and security management models have been identified in the second deliverable of the first ACIP project work package. They are quickly quoted hereafter:

- **CI Model requirements**: Dependencies, Hierarchical levels, Canonical architectures, development of formal cases, Model mapping.
- **Security Management Model requirements:** Definition of required levels for CIs and CI systems, Survivability, System security assessment methodologies, Early warning models, Incident response, System management models, Distributed intelligence, Distributed autonomy, Architecture models, Policies/Strategies.

4 Security models

4.1 Technical level

An infrastructure is constituted by a set of functional entities (source, transformation, relay, transport, user) and a family of links. The links correspond to flows between functional entities and may be oriented links (eg. water flows) or non oriented links (e.g. information) according to the nature of the supplied resource. Therefore, an infrastructure

may be modeled as a non connected graph in which the set of summits (the functional entities) is not convex.

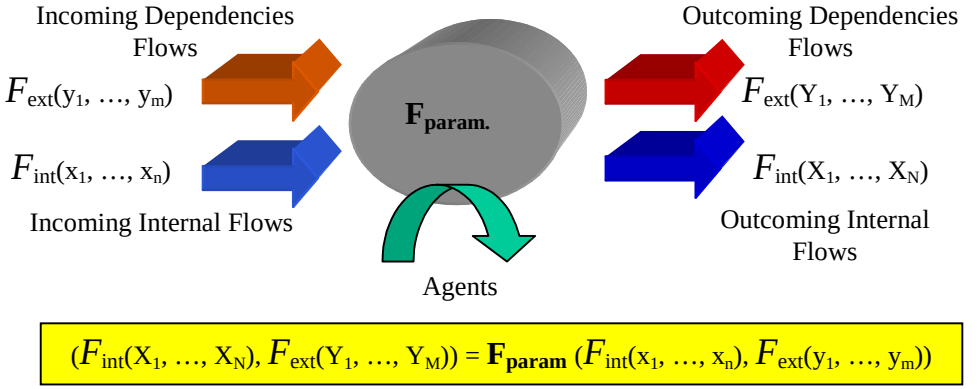

Figure 3. Model of a Functional Entity

Functional Entity

A functional entity may be modeled by a function F_{param}, as represented on Figure . This function is a specific function of flow treatment of the entity. It is parameterized by external agents or by information flows. Moreover, the function can correspond to one or several basic roles: source, transformation, routing, transport, consumption. Each entity can have internal modules. A module is an internal sub-entity that fulfills a basic role within the functional entity only for the benefit of the functional entity. In this model, interdependencies and intradependencies are modeled using two functions Fint (family of flows coming directly from another summit of the CI) and Fext (family of dependencies flows) that interact with F_{param}. Fext may comprise information flows, flows of raw materials which are not handled by the CI, functioning flows (e.g. electricity), risk flows, others.

Each flow may have the following characteristics (oriented/non oriented, material/immaterial, unicast/multicast/broadcast….) and may be of one of the following types: staff (necessary for the correct functioning of F), equipment, hardware, software, … (constituents of F), information, data (supervision, monitoring, operation, …), main resources (directly connected with the resource supplied by the CI), secondary resources (inputs necessary for F), risk flows (geographical interdependency flow, …).

4.1.1 Typology of system morphologies -- Canonical Architectures

Whatever hierarchical level we look at - from a set of functional entities to the compound of CIs – and even in the organizational schemes of CIs, it is possible to extract a limited set of canonical architectures or patterns or morphologies which have intrinsic properties. These intrinsic properties allow to develop specific, standard and abstract security measures for each canonical architectures depending on such parameters such as the context of the fault and the nature of the flow.

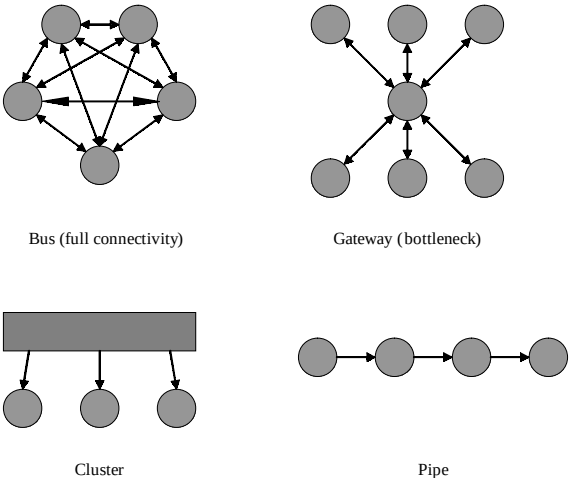

Figure 4. Examples of relevant canonical architectures

Figure displays a non exhaustive set of relevant canonical architectures. Grey circles and rectangles represent physical or management entities, or compound of these entities that are not necessarily from the same CI. Therefore the canonical architecture theory has two main characteristics:

- **Canonical architectures are flow-oriented**. Links in the figure can refer to both intra- and inter-dependencies flows.
- **Canonical architectures are generic.** Entities of a canonical architecture at a certain level can be a compound of canonical architectures at a lower level. This implies a hierarchical modeling with a relevant granularity.

By combining specific properties of canonical architectures with a certain amount of parameters such as the nature of the flow, the type and location of the fault, vulnerabilities of existing architectures can be assessed and so security solutions can be developed to design dependable and survivable architectures.

4.1.2 Temporality and cycles

The characteristic times of an entity can be classified as follows:

- Propagation time of the flows through the entity: they depend on the nature of the flow, the function F and the state of the entity,
- Self-sufficient times for the different flows: they depend on stocking mechanisms, cycles of the entity (e.g. sporadic needs of the resource supplied by a flow), the pattern of the dependencies, etc.
- Development time: this is the time to build, install and operate the entity. This time depends on the geographic location, the nature of the entity, the cycles of construction materials, …
- Others

An entity may also have cycles. A cycle is a typical chain of phases for an entity (e.g. the different phases of the life of an entity (conception, ‘manufacturing’, installation and

deployment, operation and maintenance, obsolescence, modernization or destruction)), or may express the existence of periodicities for incoming flows (some entities have one (or several) characteristic cycles over one (or several) time scale --e.g. the ring road of a city). The analysis of these cycles can reveal vulnerabilities (e.g. the existence of congestion peaks).

4.1.3 Modeling the effects of events

Events in the real physical system can be modeled locally as changes at the level of a functional entity. Inside the functional entity, this concerns the alteration of the treatment function F characteristic of the entity and the modification of the parameters of the function F. At the level of flows, it concerns the modification of the specific values of one or several flows and the activation of a critical/risk flows.

The consequences of these events can be modeled as changes at the level of the outcoming flows of the functional entity. Possible consequences may be:

- Spying: the appearance of an outcoming parasitic information flow
- Performance degradation and corruption: modification of the specific values of one or several outcoming flows
- Drying of a flow: disappearance of an outcoming flow
- Destruction: disappearance of all the flows

4.1.4 Business level vs. technical level

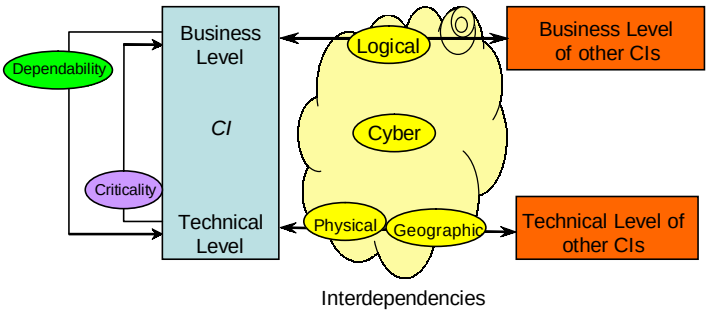

Figure 5. Business/Technical levels relationships

Figure shows the close relationship between the 2 global levels defined in the beginning of this document. For example, when a fault occurs at the technical level, it modifies the degree of criticality of the system and so will affect the business level that can in turn react over the technical level according to its security policy (Dependability). Another fact is that the interdependencies are spread on the 2 levels: logical interdependencies are usually located at the business level, Physical and Geographic interdependencies are at the technical level, and Cyber interdependencies are at both levels. Therefore, modeling aspects cannot be restricted only to the technical level without considering the relationship with the upper level.

For modeling, a CI (and even a set of CIs with its pattern of interdependencies) could be characterized as a very large-scale network system, in which a disturbance somewhere in the system can affect everything else in the system. This network, if exposed to a non-

trivial disturbance, can no longer respond linearly and either a new equilibrium may not exist or it could be reached only by control actions. Thus, there is a need for a global view of the entire network and a possibility of a quick action over the nodes of this network

4.2 Trust Model & Crisis Management Model

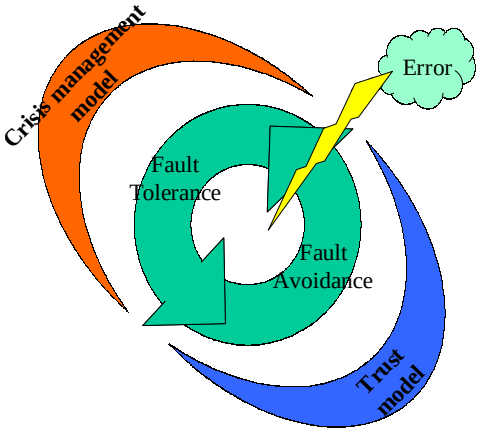

Figure 6. Crisis management and trust model (dependability view)

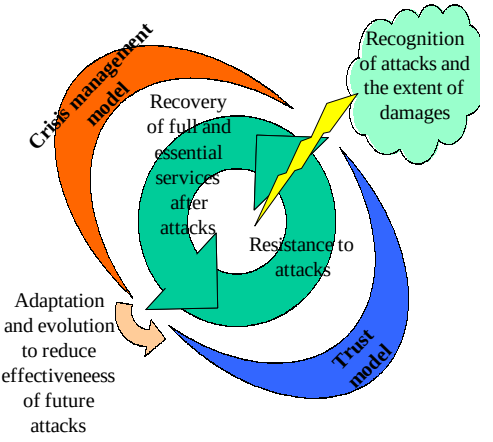

Figure 7. Crisis management and trust model (survivability view)

Two security models can be defined:

- The trust model in a system with a low degree of criticality
- The Incident Response / Crisis Management Model in a system where a fault has occurred.

Figure 6 and Figure 7 define these two models according to the previously defined dependability properties and survivability properties.

The implementation of these models for one system is fully conditioned by the security policy which depends on the dependability of this system and costs.

Some trust model examples are listed hereafter:

- Prevention can be achieved by protection (based on access control), dissuasion (based on reprisals), relation (based on negotiation), or semblance
- Fault Prevention and Fault Tolerance are based on information and on a diminution or increase in the number of dependency flows
- Forecasting and Detection are related to the analysis of faults undertaken by experts, surveillance, and registration.

Crisis management model examples may be:

- Inaction (« wait and see »)
- Fault Tolerance: in that case, two policies can be adopted: fault masking (through filtering or switching to a summit with redundancy of functionality), and fault not masked (through resignation and therefore performance collapse or by stopping critical outcoming flows which aims at stopping an epidemic process)
- Removal and Prevention

5 Policy Based Management (PBM) Approach

5.1 Definitions

This paragraph defines the terminology we use in the PBM approach.
Policy Based Management (PBM) allows a dynamic and global network management. It is g*lobal* since a network is modeled as a state machine in which the union of all local device states gives the global network state. Dynamicity is provided through policies. A network state change provokes a reaction to the event using a bidirectional management

In this chapter we use the following terms *information model*, *data model*, *model mapping*, *policy*, *policy rule*, *policy-based management* (PBM), *configuration management* and *provisioning* defined in [6].

5.2 Policy Based Network

The PBN model (represented on Figure) allows to automatically monitor and reconfigure large numbers of devices to conform to policy parameters. Policies are defined in a high-level language and some mechanism automatically translates them into the various low-level commands that various devices understand. In order to control the devices, policy mechanisms have to be linked to an existing repository of user and resource data. Not only can management tools use directory data to determine policies and locate resources required for the enforcement of those policies, but the tools can also publish information about themselves in the directory. The directory therefore becomes the main point of control on the network, with policy management tools acting as consoles for entering policy definitions and translating them into objects that get published in the directory. The repository's scope integrates all infrastructure services. This helps eliminate a lot of human mistakes, which can come either from error or from not knowing the relevant policy.
PBN provides a client-server model for policy queries and responses. This scheme is designed to be extensible to all types of policies. A policy server (Policy Decision Point, or PDP) communicates with its clients (Policy Enforcement Points, or PEPs). PEPs send requests, updates, and deletes to a PDP. The PDP then returns decisions to the PEPs. These messages must be authenticated and sent over a secure channel between the PEP and the PDP given the fact that policy servers could represent a powerful means for intruders to create massive disruptions.

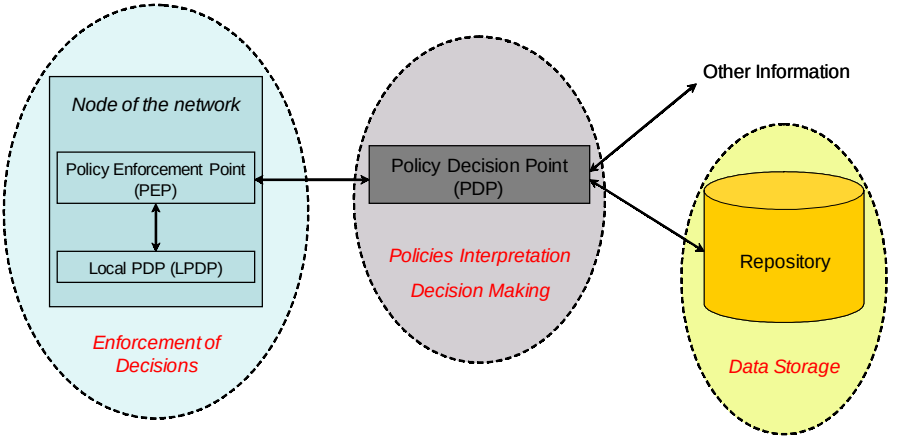

Figure 8. Policy Based Network

The mechanism is stateful. This is critical because service requests from a client PEP must be retained by the PDP until they are explicitly deleted. Without stateful communications, policies could not respond to existing conditions, which would mean that policies could not respond to environmental variables. The PDP may also push configuration information to the PEP and then remove such state information from the client when it is no longer applicable. Two Management Models exist: **Outsourcing** and **Provisioning**. Outsourcing consists in sending data to update the PDP (which in turn may update other PEPs). Provisioning concerns the installation of a policy by the PDP in PEP.
In addition, Local Policy Decision Points (LPDPs) let policy/state data and requests be offloaded to subsidiary policy servers closer to the PEPs they control. LPDPs, however, report all policy decision events to the central PDP, which can override them at any time.

5.3 Architecture

5.3.1 A hierarchical PBN

Figure represents a hierarchical PBN fitted to the CI environment. We basically identify two levels of hierarchy in the network in which the domains we identified in the framework can be mapped. At the high level, the Compound Managing Entity corresponds to a PDP, whereas the CI Managing Entity is a PEP that can be considered as a Compound Managing Agent. At the low level, the Compound Managing Agent is a CI Managing Agent's PDP. **The low level itself can be made up of several levels of hierarchy according to the granularity we wish to apply to the model.**

5.3.2 Information Model

In the context of Critical Infrastructure Protection it is important to use an information model because there is more than just one standard representation of data. The data defines semantics and behaviour of, and interaction between, managed entities. This system also needs to be federated and layered. An information model comprises attributes

that define the basic characteristics, methods that define the access to attributes and the business and system operations, the relationships, and the behaviour of the entities.

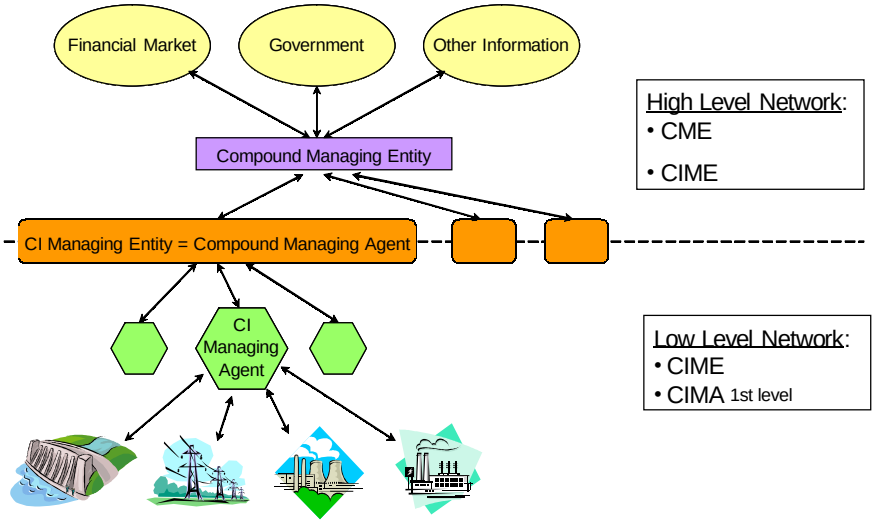

Figure 9. Architecture

5.3.3 Policy

Every configuration change, no matter how simple or how sophisticated, has an underlying set of business rules that govern its deployment. Therefore, policies seem to be adapted to manage such large information systems. A policy rule can be defined as a set of policy conditions and a set of policy actions. This allows the application of **Policy Based Management (PBM)** to these critical infrastructures. PBM is defined as the usage of policy rules to manage one or more entities. Therefore, one or several management entities control the state of the system and objects within the system using policies.

We can identify two different views for the application of a policy on the technical level: device-specific policies and generic policies. Actually, the second approach seems more appropriate since it is important to separate the modelling of policies from the modelling of device mechanisms (e.g. in the case of a fault in a system (Outsourcing Mode), there are standard policies (e.g. Actions = standard security measures) only depending on few parameters like the local morphology of the network (cf. existence of a set of **canonical architectures**), the type of faults, …). Therefore we suggest the utilization of device-independent policy models since this solution is more flexible. For example, in a company, a generic policy could consist in blocking the outgoing web-based traffic, but the actual implementation of this functionality consists in deploying filtering in firewalls and other heterogeneous network equipment of the company that may use different parameters to execute the same function. The policy is thus very high level and has to be instantiated in the devices that are actually part of the infrastructure. The crucial point that

has to be considered in order to achieve resides in the definition of a policy continuum and coherency.

5.4 Modeling approach

The models presented previously typically correspond to an Object Oriented (OO) approach. Thus, we recommend to use an OO Information Modeling focused on describing network elements and services, and how they are related to each other. This model assumes that the network is modeled as a State Machine. A network can be represented as a set of sub-networks which are sets of functional entities (the number of level depends on the wanted granularity). Each functional entity is then modeled as a state machine and so, each hierarchical upper-level entity can be modeled as a state machine. In this OO model, classes and relationships are used to model: the state of an entity, the settings to be applied to an entity that either maintain an entity's state or move the entity to a new state, and the policies that control the application of settings.

6 Conclusion and future works

We defined a modeling architecture that can fit the requirements of any CI. This architecture is based on a hierarchical PBM architecture. An assessment of vulnerabilities of existing infrastructures is achieved using canonical architectures. Then, we extract abstract security policies that are implemented using a policy based management approach. Future investigations are planned by us to implement this approach in a Java and UML based application. Moreover, we will investigate in more detail the confidence relationships between the entities (at the same level of the hierarchy and at different levels of the hierarchy) of the Policy Based Management Model.

References

[ACIP] ACIP project homepage, http://www.iabg.de/acip/

[Ro99] Robert J. Ellison, David A. Fisher, Richard C. Linger, Howard F. Lipson, Thomas A. Longstaff, Nancy R. Mead. *Survivability: Protecting Your Critical Systems.* IEEE Internet Computing, November/December 1999 (Volume 3, No. 6)

[RPK01] Rinaldi, Steven M., James P. Peerenboom, and Terence K. Kelly. *Critical infrastructure interdependencies.* IEEE Control Systems Magazine, December 2001.

[KS00] J. Knight and K. Sullivan. *Towards a Definition of Survivability*. Proceedings of the 3rd Information Survivability Workshop (ISW), Boston, MA, October 2000

[PD01] David Powell, Yves Deswarte, *On Dependability Concepts with respect to Deliberately Malicious Faults*, STCF 2001, Florianópolis/SC, 5-7 March 2001

[St02] John Strassner. *A new paradigm for network management: Business Driven Device Management*. SSGRR 2002s July 29 - August 4, 2002.

[Ba02] David E. Bakken. *Fault Tolerant System Foundations*. CptS/EE 562, Spring 2002.