AN AGENT BASED SIMULATOR FOR CRITICAL INTERDEPENDENT INFRASTRUCTURES

S. Panzieri[1], R. Setola[2], G. Ulivi[1]
([1]) Dipartimento di Informatica e Automazione, Università degli Studi "Roma Tre", Italy
([2]) Facoltà di Ingegneria, Università CAMPUS Biomedico di Roma, Italy

Introduction

The increasing relevance of the technological infrastructures (energy, communication, water supply, transportation, etc.) on our life imposes great attention about their protection. These infrastructures are complex networks, geographically dispersed and can be defined as non-linear systems that interact both among themselves, and with their human owners, operators and users [1].

Moreover, the presence of an increasing number of interdependencies among them has dramatically augmented the complexity of the whole system and, at the same time, has increased its vulnerability. Indeed, due to the tight coupling showed by these infrastructures, an accidental or malicious failure in one of them may easily spread across the networks amplifying its negative consequences and affecting remote (from geographical and/or logical point of view) users.

All of this requires to understand the behaviour of the system of systems composed by the different interdependent infrastructures, as stressed also in [2]. Unfortunately this is a very challenging task because "*the conventional mathematical methodologies that underpin today's modelling, simulation and control paradigms are unable to handle the complexity and interconnectedness of these critical infrastructures*" as stressed in [1].

However, due to the relevance of the topics, many authors have proposed modelling and simulation techniques devoted to the study of this class of systems [3].

In the literature we find, substantially, two main classes of modelling approaches: **Interdependencies Analysis** and **System Analysis**.

The first one encompasses some qualitative approaches used to help analysts to identify critical infrastructures and is devoted to better emphasize their interdependencies.

On the other side, System Analysis modelling techniques are simulation-intensive approaches able to discover hidden interdependencies and to generate (more or less precise) crisis scenarios.

These latter approaches suffer, beside the problem of defining appropriate models, to the difficulties of acquiring detailed quantitative information about each infrastructure. Indeed, the more detailed is a model, the greater the number of numeric parameters it encompasses. Some of them, moreover, may be considered sensitive information and infrastructure stakeholders appear generally very reluctant to their disclosure.

To overcome these difficulties we have developed **CISIA (Critical Infrastructure Simulation by Interdependent Agents)**. It represents a sort of hybrid approach which, on the bases of the mostly qualitative information elicited from infrastructures stakeholders, is able to set up a (rather sophisticated) fault propagation simulation.

More specifically, the simulator is developed using an Agent Based modelling paradigm where the dynamic of each agent is described via Fuzzy Logic quantities in order to take into account the uncertainties that characterize our knowledge about these infrastructures and to facilitate interaction with infrastructures stakeholders.

Critical Infrastructures Modelling and Simulation

Model procedures and simulation techniques of individual infrastructures represent a rather well developed field. Numerous products are commercially available to analyse each single infrastructures at different abstraction level, on multiple time scale and with a selectable level of details.

However, Modelling and Simulation of multiple, interdependent infrastructures are immature by comparison, even thought a number of modelling and simulation approaches are under development to directly address interdependencies and to offer insight into the operational and behavioural characteristics of critical infrastructures.

As stressed in [2], modelling and simulation analysis techniques must be employed to "*develop creative approaches and enable complex decision support, risk management, and resource investment*".

These studies are primarily devoted to determining the downstream consequences of the loss of elements in an infrastructure, such as which other infrastructure is affected (cascading and higher order effects), the geographical extend of the infrastructure outages, economic loses, etc.

In particular, they are useful to provide insight into infrastructures operations during extreme and rare events, such as major natural disasters, or a catastrophic terrorism attack. Given the rarity of these events, and the great and rapid innovation that characterize the today techno-social scenario, we cannot base our strategies only on a very limited record of historical data. Multiple simulations with

stochastic variations could provide useful information on structural characteristics of these events and about their impact on the welfare of the population [2, 3].

Obviously, no one simulation will be predictive, i.e., be able to *accurately* portray the exact consequences associated with each single event. But simulations will provide useful inputs to recovery plans, reconstruction strategies and mitigation plans.

As noted in the Introduction, the modelling proposed in the literature can be divided into two main classes.

The first one, **Interdependencies Analysis,** are qualitative techniques which help to analyse the many "dimensions" which induce interdependencies [4], considering the different characteristics of the systems [5] to identify the more critical infrastructures or sectors [6, 7].

These models are generally obtained via experts interview, round-table or workshop, and/or with the help of suitable questionnaires. Models are relatively easy to obtain but they are not able to exploit simulation capabilities neither to discover "hidden" critical elements (i.e., elements not explicitly considered by the experts).

The other approach is the so called **System Analysis.** These techniques are strongly related with computer simulation.

They are definitively quantitative approaches, and need sophisticated computational architectures, as shown also by the foundation of the NISAC (National Infrastructures Simulation and Analysis Centre) by the Los Alamos and the Sandia Laboratories.

However, due to the huge complexity of the problem at hand, one of the most challengeable task is the development of suitable models able to generate useful predictive information.

To overcome these difficulties many authors suggest the use of bottom-up approach: the whole system is described starting from its individual parts [1, 4]. In this kind of approach, generally referred as **Complex Adaptive Systems** (CAS), the whole model is obtained considering a population of interacting agents, where an agent is an entity with a *location*, *capabilities* and *memory*. The agent location defines where it is in a physical space (geographic region or abstract space, such as internet). What the agent can perform is defined by its capabilities. An agent can modify its internal data representation (*perception* capability), it can modify its environment (*behaviours* capability), it can adapt itself to environment's changes (*intelligent reaction* capability), it can share knowledge, information and common strategies with other entities (*cooperation* capability), and it can execute actions without external intervention (*autonomy* capability). Finally, the experience history (for example, overuse or aging) and data defining the agent state represent agent's memory.

Interaction among them produces the "emergence" of behaviours that are not predictable by the knowledge of any single agent.

This technique is largely used in bio-complexity researches. Bio-complexity is a multidisciplinary field that study, models and analyze the multitude of interaction (behavioural, biological, social, chemical, and physical) between living systems and their environment [8].

Actually the most suitable approach to simulate CAS system is the use of Agent-Based Modelling (ABM).

ABM is obtained interconnecting agents: i.e., independent systems that autonomously elaborate information and resources in order to define their outputs; the latter became inputs for other agents, and so on [9].

This approach is particularly useful for situations, as is the case of infrastructure interdependencies, with sparse or non-existent macro-scale information; ABM is able to use the rich sources of micro-level data to develop interaction forecasts.

One disadvantage of these simulation models is that the complexity of the computer programs tends to obscure the underlying assumptions and the inevitable subjective inputs [10].

An other disadvantage is, as mentioned in the introduction, the difficulty to acquire detailed information about each single infrastructure. This task appears, by its own, a difficult challenge [11], because this kind of information is considered very sensible by infrastructure stakeholders due to the relevance for their business. The consequence of a disclosure of these kind of information could have a bad impact on the markets.

In [12], to partially overcome this problem, three commercial simulators (PSCAD/ EMTCD, PSLF, and NS2) are integrated into a message-broker framework. Each simulator is considered as an agent that interacts with the others, exchanging simulation results through a proxy agent which works also as time scheduler for the whole simulator.

Simulator Overview

In this paper we propose a simulator for the study of faults propagation across interdependent and heterogeneous infrastructures. Its main aims are:

- To evaluate the short-term effects of one or more faults;
- To help analysts in what-if analysis;
- To single out the critical elements (i.e., those whose faults produce maximum impact).

Therefore we do not take into account neither fixing activities nor plant wearing and assume that human habits are stable during a simulation run.

Moreover, to simplify information gathering, technical and specific (perhaps sensitive) data are kept to a minimum; the goal is to use coarse grain information obtained by interviewing the managers.

In order to have the maximum level of abstraction in the description of internal mechanism and processes of each element, details and physical insights are reduced to the minimum.

To this end we adopted an ABM approach.

In particular the agents in the simulator interact by three quantities:

- ***Operative Level*** (OL): the capability of the system to perform its required job. It is a measure of the potential production/service, e.g., for an energy production plant OL=100% does not means that it is providing the maximum power, but that it could, if required.
- ***Requirements*** (R): what the system needs to reach OL=100%.
- ***Fault*** (F): it is a structured variable composed by two components: `F.value` is a boolean value that, when *true*, forces OL to zero and cannot be reverted to *false*, while `F.type` is a list of strings representing the different types of failures which affected the agent.

These quantities, represent important "states" of any agent and, besides, are propagated as inputs (IN) and outputs (OUT) among the connected agents.

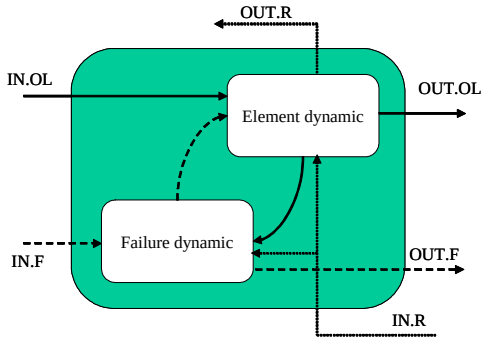

Figure 1: *Agent internal model. All the IN/OUT connections are vectorial quantities.*

Indeed agents are connected together when they exhibit some sort of dependence. In particular, as shown in Fig. 1, each agent has three inputs:

- Induced faults (IN.F): faults propagated to it from its neighbourhoods;
- Requirements (IN.R): amount of resources requested by other agents;
- Operative Level (IN.OL) of those agents whose resources are used by the agent.

The agent's output (OUT.F, OUT.R and OUT.OL) are, *mutatis mutandis*, the same quantities.

The behaviour of any agent can be described considering two different but interconnected dynamics. (see Fig. 1).

One is related to the dynamic of the elements itself. On the bases of the requirements (IN.R) communicated to the agent, and considering the resources produced by the upstream nodes (IN.OL), the agent defines its OUT.OL and identifies the level of resources it needs (OUT.R).

Moreover, OUT.OL and OUT.R depend on the level of failure of the agent.

Failures of different nature may be propagated to the agent from its neighbourhoods (IN.F), or may be induced internally, influencing the performance of the agent itself.

Each agent is connected to a set of neighbourhoods on the bases of the type of dependencies considered.

In particular we describe each type of dependency via an $n \times n$ binary incidence matrix (where n is the number of agents). We define:

- An Operative Level Incidence Matrix (m_{OL}); where the i-th row represents the set of agents that need the output of the i-th agent to perform their activities;
- A Requirement Incidence Matrix (m_R); where the i-th row represents the set of agents providing the needed resources. Note that even thought generally $m_{OL} = (m_R)^T$ we did not exploit this feature in order to guarantee a more general formulation;
- Three Fault Incidence Matrices (FIM); where the presence of a 1 in the ij-th position means that a fault may be propagated from the i-th agent to the j-th one.

Note that the existence of a propagation path does not imply in a straight way that a fault in the i-th agent induces a failure in the j-th agent. Indeed, as better explained later, the target agent will consider also the nature and the type of the fault .

In accordance with [4] we consider three FIMs which correspond to three different types of interdependencies, namely:

- *Physical FIM* (m_{FP}) that describes faults propagation via the physical linkages (i.e., that related to exchange of physical quantities) between the input and the output of two agents. This kind of fault may be generated or may afflict any agents. Note that
$$m_{FP}(i,j)=1 \Rightarrow m_{OL}(i,j)=1$$
but the converse is not true.
- *Geographical FIM* (m_{FG}) emphasizes that faults may propagate among agents that are in close spatial proximity. Events such as an explosion or fire could create correlated disturbances to all the systems localised in the spatial neighbour. The matrix m_{FG} exhibit a pattern of 1s characterized by isolated clusters. Inside each cluster, generally, we have a fully connected structure.
- *Cyber FIM* (m_{FC}), this matrix describes the propagation of faults associated with the cyberspace (e.g., virus, worm, etc.). Obviously, only a subset of the agent may be affected by this class of fault, i.e., computers and apparatus connected to the cyberspace. Note that any phisical failure is propagated, instead, via m_{FP} or m_{FG}. Cyber-dependency

defines, at first approximation, a unicum giant cluster fully connected. In other words, it is possible to re-order the agents so that

$$m_{FC} = \begin{pmatrix} U & 0 \\ 0 & 0 \end{pmatrix}$$

where $U \in \Re^{p \times p}$ is a matrix of 1s.

This characteristic emphasizes that the cyber-dependency is a global properties [4], i.e., a system that uses the cyberspace is, in principle, directly connected with any other system that uses this virtual space.

The use of three different FIM matrices, further to emphasize the different characteristics of each type of dependency, simplifies the interdependencies' discovery.

Indeed, physical interdependencies, and then the possibility of failure propagation across the underlined channel, are, generally, well known to infrastructure's experts and could be read from the functional schemas.

On the other side, geographical interdependencies are less understood by experts, but they can be discovered superimposing infrastructures' maps.

Cyber interdependencies are the less understood and the less considered into risk management plans, but, for some aspects, the most important from a security point of view [13] and the most difficult to model too.

Indeed, the hypothesis that cyber-dependency is a global dependency (i.e., nodes are fully connected [4]) is only a rough approximation (even thought this is the better model we have at hand). To have a more precise modelling, we should consider carefully also the topological structure of the cyber-space and its *scale-free* or *small world* characteristics (distinguished by the presence of important hubs inside the network [14, 15]).

Moreover, we would stress that an agent's failure affects the dynamics of neighbourhood also because it nullifies the agent's OL. In this case the simulator propagates the consequence of the failure using the m_{OL} matrix. On the other side, a failure nullifies also the agent's requirement (R), and, in some cases, this might produce even a positive effect on the network reducing the upstream requirements, and then the load of the m_R neighbours.

Finally, this class of systems, and specifically their interdependencies, are characterised by a high degree of uncertainties. While it is relatively easy to obtain, at least via experts interviews, qualitative information on them, it is an hard challenge to discover quantitative and precise information.

These considerations suggested us the use fuzzy numbers [16] to describe (at least) R and OL quantities. In particular we adopted a triangular representation of fuzzy number.

In particular, R and OL are normalised w.r.t. the corresponding nominal values. Notice that in the presence of overload condition, this quantities assume values greater than 1. This assumption facilitates the analysis of simulation results because the deviation of a variable from the unit represent an anomaly which calls for a more careful investigations.

Implementation details

One of the first requirements that we posed on the simulator regards its capability to support an easy-linkage/black box philosophy: the model of a system should be obtained connecting together the agents without any modification of their internal structure.

In particular, no information on the nature, size and type of the target (source) agent has to be explicitly included into the source (target) agent.

However, model consistence and model coherence are automatically checked at run time.

To this end, agents have an interface (shown in Fig. 2) where input, output and internal parameters are defined within the relative measurement's unit and nominal values.

Input Interface

	Unit	*Nominal Value / Type*
IN.OL	[A]	`10`

IN.F	boolean	`'short circuit'`
		`'fire'`
		`'mechanical break'`
		`'flooding'`
		`'virus'`

IN.R	[A]	`10`

Output Interface

	Unit	*Nominal Value / Type*
OUT.OL	[A]	`10`
OUT.FP	boolean	`'short circuit'` `'fire'`
OUT.FG	boolean	`'fire'` `'mechanical break'`
OUT.FC	boolean	`'virus'`

Internal Parameters Interface

Label	*Value*	*Unit*	*Description*
Tau	`2`	[s]	*Filter time constant*
T_{IB}	`15`	[A]	*Threshold for Immediate break*
T_{FB}	`12`	[A]	*Threshold for Delayed break*
T_{RC}	`700`	[bps]	*Threshold for Congestion*
...	...		...

Figure 2: *Agent's interface. "Nominal Values" and "Values" are agent's parameters, i.e. quantities are to be specified at instance-time. Notice that, for sake of brevity, we have introduced into third column also the" type" of the failures which cannot be modified.*

The supported failures are listed in the Input Interface, and they are the only failures that can be induced on the agent from the outside (i.e., transmitted from other agents).

Indeed, as mentioned before, the presence of a propagation path is only a necessary condition, while the simulator takes into account also the nature of the fault: only in the presence of a failure included into

the IN.F list, the fault has a specific effect into the agent.

An induced failure, beside putting out-of-order the agent (i.e., forcing OUT.OL and OUT.R to zero), may spread to other agents, perhaps transformed into a different type of failure. For example a "`flooding`" failure that involves an electrical device can call for the generation of a "`short circuit`" failure, as a cascade effect.

Faults are spread through the different FIM matrices in accordance with agent's output interface (see Fig. 3).

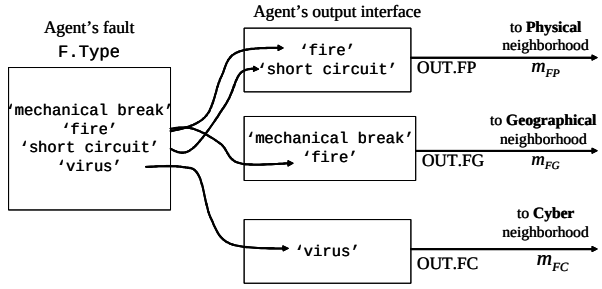

Figure 3: *Faults propagation. An Agent's fault is propagated through the related FIM matrix when its type is present into the correspondent output interface.*

As described in the previous section, agent behaviour is obtained considering its functional and failure dynamics.

At the highest level of abstraction, at each time instant, the Operative Level (OL) can be described as:

$$OL = \Omega\left(F, \overleftrightarrow{IN.R_1}, \cdots, \overleftrightarrow{IN.R_q}, \overleftrightarrow{IN.OL_1}, \cdots, \overleftrightarrow{IN.OL_p}\right) \quad (1)$$

where q is the number of requirements requested at the agent, p is the number of resources used by it. The over-arrow ($\overleftrightarrow{\ }$) indicates that the quantity is de-normalised in accordance with the ratio of the agent's `Nominal Value` (defined in the input interface) and that of the source (specified in the messages, see later).

Function Ω is a suitable law which describes agent dynamics. Notice that dependency on the requirements (IN.R) allows to take into account overload conditions.

Obviously a general formulation of Ω cannot be defined because it strongly depends on the nature of the object modelled. This also because we want to leave the infrastructure's expert free to model the element dynamics in the most appropriate way.

However, in order to illustrate the degree of abstraction, we report in equation (2) the particularisation of g for an agent which depends in affinely on p uncorrelated and inhomogeneous resources (e.g., a pipe, an electrical circuit, etc.)

$$OL = \overline{F} \cdot \left\{ \min\left[\max\left(1, \overline{IN.R_1}\right) \right], \overline{a}_1 \overline{IN.OL_1}, \ \ldots \ , \overline{a}_p \overline{IN.OL_p} \right\} \quad (2)$$

The overbar represents complement operator, while the weighting factor a_i, thanks to the use of Fuzzy quantities, can be interpreter as quantifier attributes on degree of dependence with respect to the specific IN.OL, e.g., agent's OL depends *much* on IN.OL1, *little* on IN.OL2, and so on.

Generally,

OUT.OL = OL

but for some agents we need to model also more sophisticated relation. For example congestion phenomena in a WEB-server can be modelled as

$$OUT.OL = \begin{cases} OL & if \quad IN.R \le T_{RC} \\ OL - \left(\overleftrightarrow{IN.R} - T_{RC}\right) & otherwise \end{cases}$$

The output requirement is defined similarly as

$$OUT.R = \Gamma\left(F, \overleftrightarrow{IN.R_1}, \cdots, \overleftrightarrow{IN.R_q}, R_{\text{int}}\right) \quad (3)$$

We have assumed that OUT.R does not depend directly on the agent's resource (IN.OL), but it may depends on an endogenous requirement (R_{int}).

When the requirements are dimensionally homogenous (e.g., an agent which requires power supply from many sources) the function Γ can assume a form like

$$OUT.R = \overline{F} \cdot \left(\sum_i \overleftrightarrow{IN.R_i} + R_{\text{int}} \right)$$

while in the presence of inhomogeneous quantities, we can consider a formulation like the following

$$OUT.R = \overline{F} \cdot \left(\max\left(\overleftrightarrow{IN.R_1}, \cdots, \overleftrightarrow{IN.R_q}\right) + R_{\text{int}} \right)$$

For what it concerns the failure dynamic,

`F.value` = OR(`F.value`, F_R`.value`, F_I`.value`)

`F.type= [F.type | `F_R`.type | `F_I`.type]`

`F.type` is obtained concatenating the different types of failure that affect the agent (removing the duplicated entries). Notice that the presence of `F.value` in the right side of the first equation forces the agent to permanently remain in fault condition. F_R is the internal failure due to overload condition, while F_I represents induced failures.

An example of how an internal failure is modelled, is given by equation (4)

$$F_R.value = \begin{cases} true & if \quad IN.R \ge T_{IB} \\ true & if \quad R_{filtered} \ge T_{FB} \\ false & otherwise \end{cases} \quad (4)$$

where the first line models an immediate break due to an excessive load, while the second line models failure produced by a protracted overload condition (see Fig. 4).

On the other side, as described before,

F_I`.value` = *true*

when `IN.F.type` is compatible, i.e., the type is present in the agent's input interface.

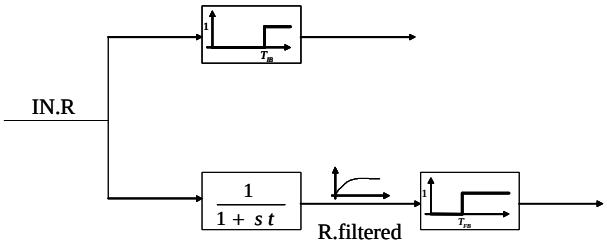

Figure 4: *Internal fault definition.*

In any case the type of the induced fault depends on the nature of the fault and on the characteristic of the agent.

When a failure is induced into an agents, it is propagated to its neighbours in accordance with the different FIM matrices, see Fig. 3.

During simulation, each agent communicates via messages. At every time instant an agent sends messages to its neighbourhoods in order to specify its needs (requirements), communicate its level of service (operative level) and/or propagate faults (physical-faults, geographical-faults and cyber-faults).

To each type of message, as mentioned before, it is associated an incidence matrix which specifies agent's neighbourhoods for that type of message.

In particular "requirements" (operative-levels) messages are structured as a an array of structures, with each component defined as

Value	Unit	Reference

where

`Value`: is a triangular Fuzzy number which expresses, in the range (0,1), the desired requirement (operative-level).

`Unit`: is a string which shows the measurement's unit used to define the variable.

`Reference`: is its nominal value.

The presence of the `Unit` information inside any structure has a double aim: first, during the initialization phase, `Unit`, in conjunction with `Reference`, allows the checking of the syntactic coherence of the model, e.g., verifying that there is a type correspondence between data; second, during simulation, the information `Unit` is used by target agent to extract from the message the proper data subset.

Note that, with this formulation, one has to define only an incidence matrix for requirements (operative level), instead of boring analyst with the need to introduce different matrices for each type of physical quantities.

Also fault's messages are arrays where each component has the structure

Value	Type

where `Value` is Boolean and `Type` is a string describing nature of the fault.

The simulator has been implemented using ABM modelling framework *RePast* [17].

RePast (REcursive Porous Agent Simulation Toolkit) is a software framework for creating agent based simulations using Java. It has been developed at the University of Chicago, Social Science Research Computing centre and provides a library of classes for creating, running, displaying and collecting data from an agent based simulation. In addition, RePast can take snapshots of running simulations, and create quicktime movies of simulations [18]. It is distributed under the GNU General Public Licence

RePast has been widely used to model complex infrastructures [19], in social sciences, in military and intelligence framework , and for the simulation of the Electricity Market [20].

RePast behaves as a discrete event simulator whose quantum unit of time is known as a *tick.* The tick exists only as a hook on which the execution of events can be hung, ordering the execution of the events.

A *RePast* simulation is primarily a collection of agents of any type and a model that sets up and controls the execution of these agents according to a schedule. *RePast* agents are defined using some super-classes like *SimModel*. Using this class, we define a new Java-class for each type of agent that we need in the model, implementing the logic described in the previous section.

Case study

In order to validate our approach we particularized it to simulate the system composed by the interdependent infrastructures existing in the University Campus of one of the authors. For sake of simplicity we have considered only three infrastructures and focused the attention on their most relevant components (see Fig. 4), specifically we have considered the following elements:

- *Power supply infrastructure*: the external power supply and two different circuits: one supplies the air conditioning system and the other the information infrastructure.
- *Information infrastructure:* the IP-network and the related services. Two different set of PC-servers has been considered. *Server A* hosts the automatic control system of the air conditioning infrastructure, while *Server B* hosts critical operation (e.g., DNS, Switch, etc.).
- *Air conditioning infrastructure*: constituted, in addition to the pipes, by one compressor and the thermal exchangers devoted to air conditioning of the Servers' room.

This third infrastructure has been included due to the presence of a sophisticated control mechanism that uses the IP-network, via *Server A*, to regulate the temperature set-point inside each room.

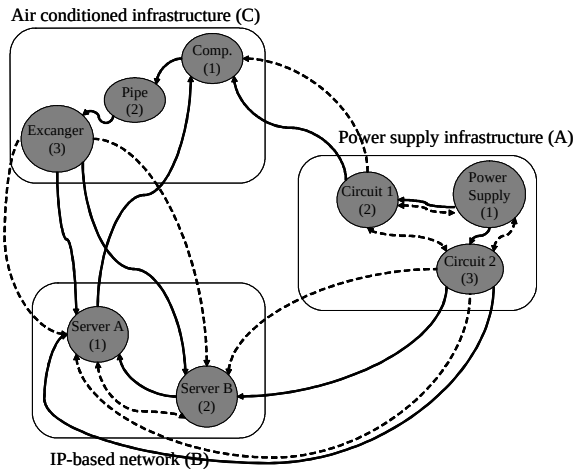

Figure 4 – *Test-bed system. Dotted lines represent fault propagation paths, continuous lines functional dependencies. Each infrastructure is identified by a letter and each node by a number. For sake of clearness the requirement connections are not shown.*

To implement this system we have developed seven classes of agents and instanced 8 agents with their own parameters. The topology of the system has been specified via the incidence's matrices (see Fig. 5).

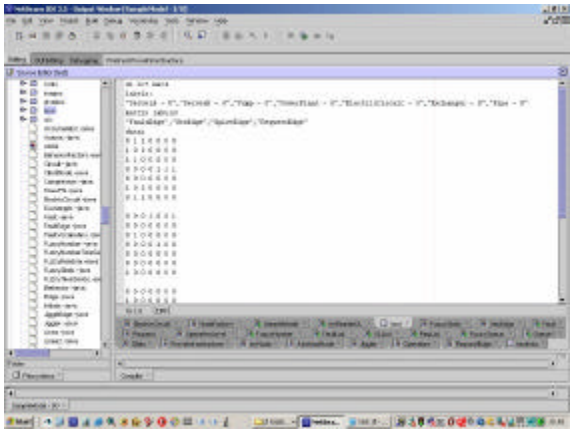

Figure 5 – *Model declaration: this file defines the agents in the model, their parameters and the relative dependencies expressed via five incidence matrices.*

In Fig. 6 the related RePast model is shown, here we exploited the graphical features of RePast to represent the networks. In particular, in Fig. 6, m_{OL} connections are shown, but one can represent any other connection type.

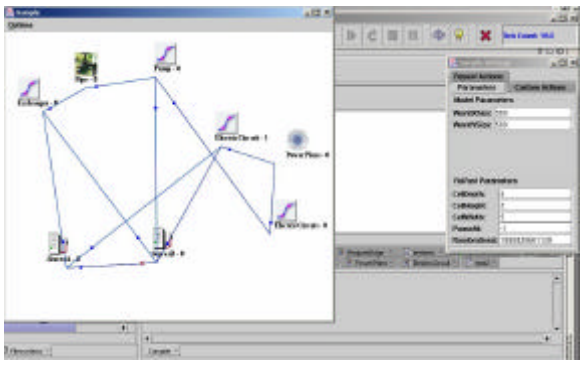

Figure 6 – *Test-bed model.*

At any run we forced a different failure in the agents and analysed both performance degradation of the whole system (i.e., the time histories of the OL variables), and faults propagation across the infrastructures.

Even thought this case study is very simple, and actually the results offered by the simulator did not increase our knowledge of the system, it helped us to validate our modelling approach and to test the correctness of the simulator.

Conclusions

Modelling a network of systems whose nodes are the different and heterogeneous infrastructures at the base of our society is a great challenge for the next years. The intrinsic complexity of each infrastructure, their multi-scale and geographic dispersed nature, the absence of global control mechanisms, and the presence of many physical and logical interdependencies among them, make extremely difficult to predict their behaviour.

However, new and dangerous threats impose to improve the robustness of those networks with respect to accidental and malicious (specifically terrorist) actions.

To this end it is mandatory to develop analytical tools able to emphasize the more critical elements and skilled enough to help us to discover hidden interdependencies. Indeed, the presence of these links certainly constitutes the less perceived element of the whole risk, and then one of the major vulnerabilities for the complex system.

To this end, in this paper we propose **CISIA** a multi agent simulator for critical infrastructures useful to analyse fault propagation across heterogeneous infrastructures. **CISIA** conjugates the powerful of the simulation tools proper of the System Analysis with model simplicity of the Interdependencies Analysis approaches.

CISIA adopts an Agent Based approach modelling the different infrastructures' elements. The agents interact each others to exchange resources, but also because faults spread through them.

In **CISIA** agents' dynamics is described using Fuzzy Logic variables to take into account the

uncertainties that characterize the knowledge of this class of systems.

Moreover, the linguistic nature of Fuzzy Logic allows analyst to greatly simplify information elicitation, and helps during the interaction with infrastructure stakeholders.

CISIA has been successfully tested in some low complexity situations. Future work, is devoted to test it on more complex systems.

Thanks

Authors thank Leonardo Carnassale for his help in the developing of the **CISIA**.

References

[1] M. Amin, "Modelling and Control of Complex Interactive Networks", *IEEE Control System Magazine*, pp. 22-27, 2002.

[2] U.S. *The National Strategy for The Physical Protection of Critical Infrastructures and Key Assets*, 2003, http://www.whitehouse.gov/pcipb/physical.html

[3] S. Rinaldi, "Modelling and Simulating Infrastructures and Their Interdependencies", *Proc. 37th Hawaii Int. Conference on System Sciences*, 2004.

[4] S. Rinaldi, J. Peerenboom, T. Kelly, "Indetifying, Understanding and Analyzing Critical Infrastructure Interdependencies", *IEEE Control Systems Magazine*, pp. 11 - 25, 2001.

[5] B. Ezell, J. Farr, I. Wiese, "Infrastructure Risk Analysis Model", *In: Journal of Infrastructure Systems*, vol. 6, 3, pp. 114-117, 2000.

[6] D. Reinermann, J. Weber, "Analysis of Critical Infrastructures: The ACIS Methodology (Analysis of Critical Infrastructural Sectors)", *Critical Infrastructure Protection Workshop*, Frankfurt, 2003.

[7] Predict Defence Infrastructure Core Requirements Tool (PreDICT). http://www.defence.gov.au/predict/general/predict_fs.htm.

[8] A. Mikler, M. Monticino, B. Callicott, S. Khalil, "Agent Based Modelling of Human and Natural Systems and their Interactions", *7th Annual Swarm Researchers Meeting - SwarmFest*, Notre Dame, IN, (2003)

[9] M. Luck, P. McBurney, C. Preist, "A Roadmap for Agent Based Computing", Agent Technology: Enabling Next Generation Computing, AgentLink II, 2003, http://www.agentlink.org/roadmap/roadmap.pdf

[10] A. Wenger, J. Metzger, M. Dunn, I. Wigert *International CIIP Handbook 2004*, ETH, the Swiss Federal Institute of Technology Zurich, 2004 http://www.isn.ethz.ch/crn/publications/publications_crn.cfm?pubid=224 .

[11] J. Moteff, G. Stevens, "Critical Infrastructure Information: Disclosure and Homeland Security", *Report for Congress RL31547*, The Library of Congress, 2003.

[12] K. Hopkinson, K. Birman, R. Giovanini, D. Coury, X. Wang, J. Thorp, "EPOCHS: Integrated Commercial Off-the-Shelf Software for Agent-Based Electric Power and Communication", *Proc. Winter Simulation Conference*, pp. 1158 – 1166, 2003

[13] U.S. *The National Strategy to Secure Cyberspace*, 2003; www.whitehouse.gov/pcipb

[14] M. Newman, "Models of the Small World', Cond-mat/0001118v2, 2000.

[15] A. Réka, A. Barábasi, "Statistical Mechanics of Complex Network", Cond-mat/0106096v1, 2001

[16] A. Kaufmann and M. M. Gupta, "Introduction to Fuzzy Arithmetic Theory and Application", Van Nostrand Reinhold, New York, 1991.

[17] REcursive Porous Agent Simulation Toolkit (RePast), http://repast.sourceforge.net

[18] N. Collier, "RePast: An Extensible Framework for Agent Simulation", http://repast.sourceforge.net/modules.php?op=modload&name=News&file=article&sid=21&mode=thread&order=0&thold=0

[19] http://www.dis.anl.gov/msv/msv_cas.html

[20] Argonne National Laboratory, Electricity Market Complex Adaptive Systems Model (EMCAS), www.dis.anl.gov/msv/cas/EMCAS.html