Agent-based Modeling of Interdependencies in Critical Infrastructures through UML

Valeria Cardellini, Emiliano Casalicchio, Emanuele Galli
Dip. di Informatica, Sistemi e Produzione - Università di Roma "Tor Vergata"
{cardellini, casalicchio}@ing.uniroma2.it, emanuele.galli@gmail.com

Keywords: Agent-based modeling and simulation, UML, critical infrastructures, discrete event simulation

Abstract

In this paper, we investigate the adoption of agent-based modeling and simulation to study the interdependencies in critical infrastructures. We propose an approach to model critical infrastructures through agents as well as some guidelines to simulate such complex systems. We adopt UML as modeling language, thus proposing a new way to represent the interdependencies that occur in a complex system composed by different critical infrastructures. Through an example that regards an information system for civic emergency management, we describe our modeling methodology, discuss about the measurements and results that can be gained from the simulation of the complex system, and present the most relevant issues regarding the simulator design and implementation.

INTRODUCTION

A critical infrastructure is a physical system that, if disrupted, can seriously affect the national security, the economic and social welfare of a nation. Example of critical infrastructures include telecommunications, electric power systems, natural gas and oil, banking, transportation, water supply systems, government and emergency services [1].

Within any critical infrastructure there are dependencies among its components and some of them involve humans. Moreover, critical infrastructure are intertwined and heavily dependent on each other [12]. Therefore, in case of disruptions, what happens to one infrastructure can directly and indirectly affect other infrastructures. For example, in case of a natural disaster, a key problem is that when a critical infrastructure such as the power grid goes down, others may come down simultaneously. To protect the critical infrastructures, it is necessary to study the complex system behavior and the interaction processes among humans and the system components, when they are stressed or attacked, and to understand the emergent phenomena originated by the individual behavior of the infrastructure components. Therefore, as observed in previous works (e.g., [5, 12, 11]), a challenging issue consists in providing formalisms, methodologies, and tools to model the entire complex system composed of humans and critical infrastructures.

In this paper, we consider the adoption of the agent-based modeling and simulation (ABM&S) approach to study the interdependencies in critical infrastructures. To model the interconnectedness of such systems, we propose an UML-based methodology, which we present by means of an example that concerns an information system designed for the management of civic emergencies. To the best of our knowledge, this paper represents the first proposal for modeling and simulating through agents and UML the interdependencies in critical infrastructures. Agent-based modeling is largely used in the field of bio-complexity research, which studies the multitude of interactions between living systems and their environment. Such complex systems and the interactions and interdependencies among them could exhibit similitude, in complexity and behavior, to the technological critical infrastructures that drive the life in our society and that directly or indirectly interact with the environment and humans.

Agent-based modeling is obtained by interconnecting agents that is, independent systems that autonomously elaborate information and resources to define their outputs, which become inputs for other agents. Agents are autonomous, problem-solving computational entities, which are capable of effective operations in dynamic and open environments. Agents are often deployed in an environment in which they interact, and maybe cooperate, with other agents that have possibly conflicting aims. The environment is also modeled as an agent. Agents can be easily implemented by objects: all the object-oriented and agent-based technologies can be therefore used to realize simulators of agent-based models.

Three approaches can be followed to design an agent-based simulator of a critical infrastructure. In the *from-scratch* approach, the simulator is designed ex-novo using an agent-based framework and/or an object-oriented programming language. In the *integration* approach, the simulator is designed using existing components, that are specialized simulators of system components. These simulators may use different modeling and simulation techniques to represent the dynamic of the target system. Each model is embedded into an agent, which interacts with other agents. Agents exchange results and cooperate in simulating the whole system. Finally, in the *hybrid* approach, some system components are modeled by embedding existing models into the functional part of an agent, while the models of the remaining system components are developed from scratch. All the simulation process is

based on an agent-based simulation framework.

An example of a critical infrastructure simulator based on the integration approach is provided by EPOCHS [6], which has been realized to study the interactions between the electrical power grid and the communication grid. EPOCHS uses a multi-agent approach to integrate a simulator to study electromagnetic transients, a simulator to study electromechanical transients, and a network simulator to analyze communications. Each simulator is an agent that interacts with the other agents by exchanging simulation results through a proxy. A further agent implements the simulation engine that coordinates the whole system simulation. Agent-based simulation has also been used to study the human behavior in case of fire inside a building [4]. A fire propagation simulation model is integrated with a model of humans that escape from a building, using a shortest path algorithm where the weight of the path depends on the presence of fire and smoke. Another agent-based simulator is EpiSims [3], which models the spread of disease and the effects of mitigation efforts at the level of individuals in a large urban area.

There are different motivations that have driven us to choose ABM&S to study the interdependencies in critical infrastructures. First, this approach allows us to simulate a complex system composed by many subsystems, which can be organized hierarchically, and where the complex interactions among the entities make difficult the use of equation-based models. By exploiting the ABM&S features, we can embed the model of the complex system into agents and model such system at different levels of abstraction. Second, ABM&S offers the flexibility needed to customize the target system as required by the case under study. Furthermore, ABM&S not only represents a valid support to conduct a what-if analysis, but it also allows to investigate different aspects of the system dynamic, in particular to study the impact of an unexpected perturbation on the interconnected infrastructures and thus to predict the infrastructure responses to this perturbation. For example, some specific questions that can be answered by an agent-based simulation study of critical infrastructures are: what is the failure propagation path from the affected infrastructure to the interdependent ones and its propagation time? What are the direct and side effects of a failure? Can the system react to a certain type of failure and how much time does the reaction take? What is the impact of a miss behavior in a recovery plan? Last but not least, ABM&S could exploit the benefits of parallel and distributed simulation, which is fundamental to scale up the model and simulate hundreds of thousands of agents.

THE IS4CEM APPLICATION

In this paper, we propose a modeling approach which is general enough to be applied to any critical infrastructure case study. However, for clarity and easy of presentation, we describe how to apply our methodology through an example-driven approach, which considers as a critical infrastructure application the Information System for Civic Emergency Management (IS4CEM), which, in case of some natural disaster or special event, provides information about the health care centers availability, the transportation network availability, and the event evolution.

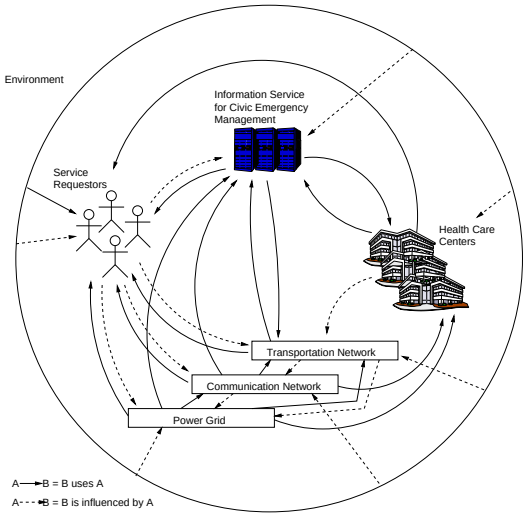

Figure 1. The Civic Emergency Management scenario.

Let us consider the scenario illustrated in Figure 1 and describe its interdependencies. There is a generic service requestor (SR) that accesses the IS4CEM, which is hosted and published by a service provider through a wired network. The SR can be a citizen asking for help (in the following, wounded) or a succorer and she/he uses a set of critical infrastructures to access the IS4CEM. Specifically, the SR uses a wired or a wireless connection to the communication network; she/he can use the power grid to power its connecting device; finally, she/he may use the transportation system to provide first aid to other service requestors or to reach a Health Care Center (HCC) to obtain assistance.

The IS4CEM is an information system that runs on a service provider platform and provides various information to the citizens. It also operates as a broker for first aid requests by selecting the most appropriate HCC to be reached (e.g., by taking into account the transportation system and HCCs status). The IS4CEM is accessed through the communication network and uses the information system of the transportation network to get information about its availability. The IS4CEM does not directly use the transportation network; however, since the transportation network may be used to provide assistance to the IS4CEM, the latter depends on the transportation network. Finally, the IS4CEM relies on the power grid for its functioning.

Each HCC (specifically, its information system) uses the IS4CEM to obtain availability information on the other HCCs

and to get information about emergencies. Each HCC uses the communication network and the power grid.

The transportation network is used by the service requestor and by all citizens. Its information system dialogs with the IS4CEM to update the transportation network status. Finally, the environment influences all the system components.

In Figure 1 the dashed lines represent the direct or indirect influences between two system elements, with the arrow indicating the direction of the influence. The plain lines indicate the interactions between two system elements, with the arrow indicating the flow of the interaction. For example, the dashed line from the SR to the transportation system means that the SR influences the transportation network (e.g., generating a traffic jam), while the plain line from the transportation network to the requestors indicates that the requestors use the transportation network. We assume that if a system element A uses another system element B, then A's behavior is also influenced by B's behavior.

Since we deal with critical infrastructures, we need a model that allows to capture the interdependencies in the system, because they are stressed by the happening of some critical event: for example, the impact of a natural disaster on the communication network, the requestor, and the IS4CEM service provider platforms, as well the impact of a service failure on other interdependent services such as the transportation and health care. To analyze the interdependencies effects, we believe that it is more appropriate to use an agent-based model and simulation approach rather than an equation-based approach. Indeed, the dynamic models of each system component help us only in modeling and understanding the single component behavior, without providing us a whole picture of the complex system behavior.

THE AGENT-BASED SYSTEM MODEL

All the subsystems that compose the IS4CEM complex system are modeled as agents. We distinguish among *actors*, *infrastructures*, and the *environment*.

An actor is the entity who uses, directly or indirectly, the infrastructures. In our case study, the service requestor is the generic actor that can become an user of the infrastructures, depending on the phase of her/his life cycle or role. For example, a succorer can use the communication network to interact with the IS4CEM and the transportation network to physically reach the wounded.

An infrastructure is a subsystem that provides some service to the actors and to the other infrastructures: the power grid, the communication network, the HCCs, the transportation network, and the IS4CEM are all infrastructures.

The environment is the place in which actors and infrastructures operate. In this paper, we consider the environment as the source of perturbations that influence (negatively) the life cycle of both actors and infrastructures. Typically, a perturbation (e.g., a natural disaster, an overload condition) will alter the normal operative level of a subsystem.

The UML-based modeling approach

To model our system we have decided to use UML. Although our goal is to model and simulate real complex systems composed not only of software systems but also of humans and infrastructures of different nature, we have chosen UML because all these real entities are modeled as agents and the latter are implemented as components of an agent-based discrete event simulator. Therefore, each real entity is mapped into one or more software agents. The UML choice as modeling tool for the interdependencies among critical infrastructures allows us to easily develop an agent-based simulator using an object-oriented framework and will facilitate its extension, engineering, and maintainance. A consistent number of proposals use UML to model agents and multi-agent software systems: in [10] Odell et al. have introduced new stereotypes and proposed minimal extensions; new UML diagrams have been proposed by Kavi et. al [7]; in [2] da Silva et al. have extended the UML meta-model through the description of new meta-classes and stereotypes and the introduction of new diagrams. Despite the UML-based modeling approaches that have been specifically proposed to model multi-agent software systems, in this paper we use only the standard diagrams defined in UML 2.0 [8] and introduce only some stereotypes for modeling interdependencies.

To model a complex system and its interdependencies we propose the following sequence of steps.

1. We identify the subsystems and associate to each subsystem an agent. We suppose that, at the desired degree of detail, each subsystem can be modeled by a single agent; however, if a deeper degree of detail is needed, it is possible to carry out a next refining step, after which a subsystem will be modeled with two or more agents.
2. For each agent, we identify its behavior and its characteristics that is, location, capability, and memory.
3. We identify and define the use cases.
4. Starting from each use case, we produce a communication diagram to provide a macro description of the main interactions and interdependencies.
5. We refine the macro model of interactions and interdependencies through sequence diagrams. We produce a sequence diagram for each communication diagram.
6. We model the internal behavior of the agents identified in step 1 by using the activity or state chart diagrams.

As example of use case diagrams produced in step 3, let us consider the first aid request and route selection to HCC use cases, which are shown in Figure 2. We use the actor symbol to represent the agent that activates the use case as well as the subsystem that influences the use case, because they

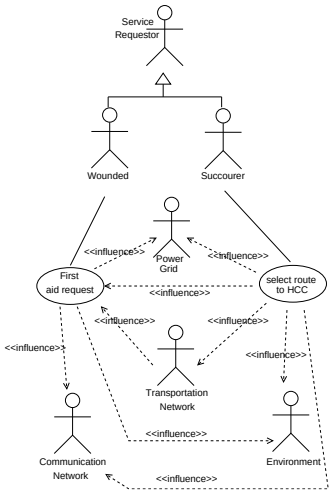

Figure 2. Use cases for first aid request and route selection to HCC.

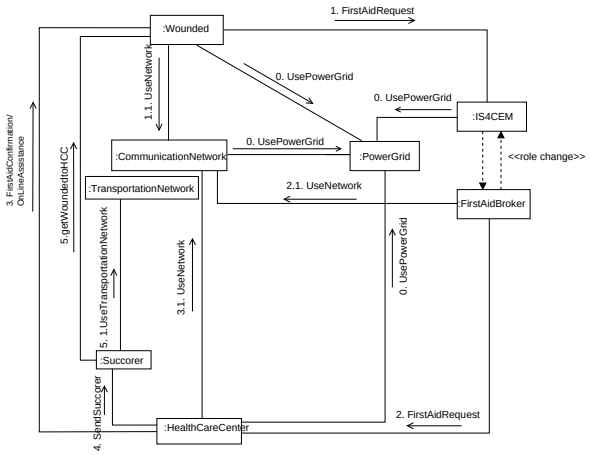

Figure 3. Communication diagram for first aid request.

are directly or indirectly involved in the modeled process. Our choice can appear in contrast with the UML philosophy, where actors are things that cannot be affected by the system design [8]. However, from our point of view, even if we provide a model for the actors, the actors represent subsystems that are not influenced by the system or by the procedure that we design. Let us explain with the following example. The wounded activates the first aid request process. The latter is influenced by the environment, the communication network, and the power grid. In its turn, the first aid request process indirectly influences the transportation network and directly the select route process. In a real case, the design of the management procedure of a first aid request has no effect on the design of the communication network, the transportation network design, or the power grid. Then, we represent these subsystems as actors, even if, in the simulation model, we will provide a model for such agents. The relation *B* influences *A* (or *A* depends on *B*) is represented by the relationship's stereotype «`influence`», graphically depicted with a dashed arrowed line from *A* to *B*.

The interdependency model

In our approach, dependencies and interdependencies are modeled at high level by communication diagrams and are then refined by means of sequence diagrams. Figure 3 shows the communication diagram derived from the first aid request use case in Figure 2.

The participants are the agents. Two participants are linked together if either there is an interaction or an interdependency exists. Interactions are represented by messages with an ordering number greater than 0. The hierarchical numeration of messages indicates the consequent and indirect use of the infrastructure caused by an high-level action. The interdependencies are represented by messages numerated with 0, and are always present (or time independent).

Let us illustrate this concept through our example. The wounded generates a first aid request `1.FirstAidRequest` to the IS4CEM service provider. The service provider changes its role to first aid broker that is, it selects the best HCC (e.g., nearest and least loaded) that may provide the appropriate aid to the wounded, and it sends the first aid request to the selected HCC (`2.FirstAidRequest`). To model the role change we use the «`role change`» stereotype [10]. Both the wounded and the IS4CEM use the communication network to send the first aid request; this action dependency is represented by the `1.1.UseNetwork` and `2.1.UseNetwork` messages. When the HCC receives an aid request, it sends a succorer (`4.SendSuccorer`). The succorer will reach the wounded and get her/him to the HCC (`5.GetWoundedtoHCC`). The interdependency with the transportation network, which is caused by its usage to reach the wounded, is modeled by the `5.1.UseTransportationNetwork` message. At the same way, the dependency of the participant on the power grid is represented with the `0.UsePowerGrid` message.

The detailed description of the interdependencies modeled through the communication diagram is provided by the sequence diagram in Figure 4, which refines the first aid request communication diagram in Figure 3). Although communication and sequence diagrams are equivalent in the standard UML [8], we have decided to use the communication diagram to represent the interdependencies at an higher level of abstraction, because in the communication diagram the focus is on the participants (agents, infrastructures, and environment) and links (interactions and interdependencies). Therefore, at a first glance, it is clear who are the participants and who interacts. On the other hand, the sequence diagram allows to provide a clear description of the message ordering, the synchronous and asynchronous messages. Therefore, it turns out

to be more suitable than the communication diagram to describe the interactions at a lower level of abstraction.

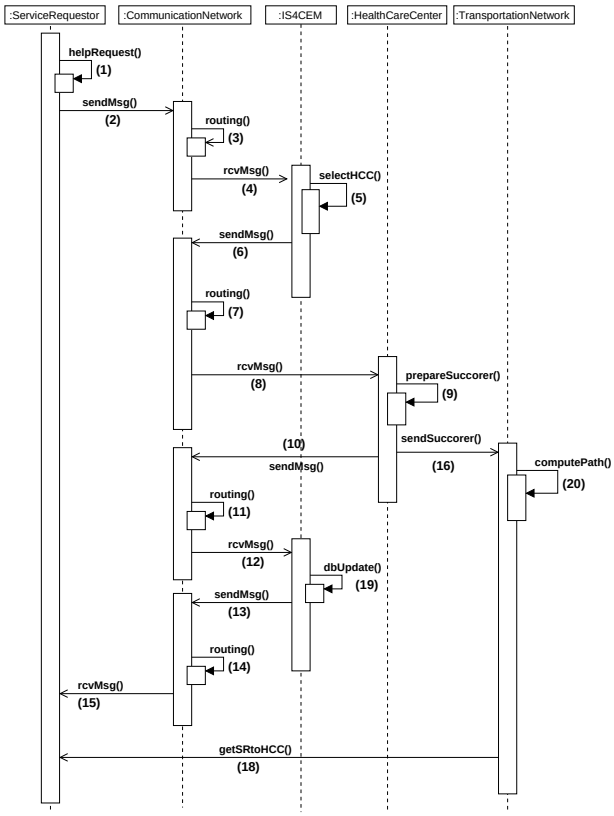

Figure 4. Sequence diagram for first aid request.

Referring to Figure 4, the SR agent asks for help (1), e.g., using its smart phone. The device sends a message to the IS4CEM. Steps (2), (3), and (4) model the use of the communication network and the network routing of the message to the IS4CEM. Once received the request, the IS4CEM agent analyzes the message, determines the best HCC that can serve the help request (5), and sends the first aid request to the HCC (6)-(8). The HCC arranges the succorers team (9) and sends, e.g., an ambulance (16). The HCC also sends a confirmation to the IS4CEM (10)-(12), that updates its database (17), and (optionally) sends a confirmation message or starts providing some online aid to the SR (13)-(15). The succorer reaches the SR and gets her/him to the HCC (18). In the simulator, all the phases (1)-(18) correspond to agents's method invocations.

The agents model

Each agent is characterized by its location, capability, and memory. The agent capability are: perception, behavior, adaptation, cooperation, and autonomy. The SRs, the IS4CEM and the HCCs share some common characteristics: they have their own location in the communication and transportation networks and in the power grid. The power grid, the transportation and communication networks are all complex networks composed of thousands of nodes. We identify the characteristics that differentiate the agents in our system.

Service requestor. We assume that a SR is a human; therefore, it has the intrinsic properties to store and retain information from its experience and to use this information to augment its knowledge. The SR stores information about the used services, e.g., the route to reach one or more HCCs. The SR has the capability to modify its internal state on the basis of the recalled information by observing the state of the other system elements and the environment (perception capability). It also has the capability to act autonomously and cooperate with other service requestors. The SR behavior is determined by the rules to use the services provided by the infrastructures; in addition, the SR has also the capability to adapt its behavior on the basis of the system state. For example, if the requestor has no connectivity to access the IS4CEM and query it to know the route to the nearest available HCC, it will go to one of the HCCs that it is capable to reach using a well known route (without the IS4CEM assistance). The SR behavior depends on the power grid, the communication network, the service provider, and the transportation network.

IS4CEM. The IS4CEM main task is to get and store the state information from the environment and the other infrastructures. It also has the capability to cooperate and interact with the latter by exchanging information. The IS4CEM may also have the capability to percept a change in the system state and, if necessary, to adapt its behavior. For example, if the IS4CEM has no connectivity to the information service of the transportation network, it cannot receive updated information about its state. However, the IS4CEM can provide information about the fastest route using its history and some forecasting algorithm. The IS4CEM behavior depends directly on the power grid and the communication network and indirectly on the transportation network and the HCCs.

Health Care Center. The HCC is a complex system composed of humans and devices. We assume that the HCC has two tasks: to update the IS4CEM and serve patients requests. In the first case, the HCC publicizes its operativity level that is, its actual capacity. In the latter case, we suppose that the HCC has a capacity to serve an average number of health care requests per time unit. The HCC also has the capacity to memorize its state and the capability to adapt its behavior, for example by increasing its capacity during an emergency situation. The HCC may also interact and cooperate with other HCCs. The HCC behavior depends on the communication network, the transportation network, the power grid, the IS4CEM, and the SR behavior.

Communication network. This infrastructure is a complex network itself, composed of thousands of nodes and communication links, which are characterized by different bandwidth capacities and delays. We use a simple model for

the communication network; we plan to investigate in future a more detailed network model that can be incorporated as internal or external simulation model. We model the network as a routing matrix $N = \{n_{i,j}\}^{m \times m}$. The rows and the columns of the matrix are indexed by the system elements (agents). The matrix element $n_{i,j} = (c_1, d_1, c_2, d_2, ..., c_p, d_p)$ contains all the possible communication network routes (if any) from i to j, and for each route the actual estimated capacity (c) and delay (d) are specified. For example, if packets sent from i to j can be routed in two ways, then $n_{i,j} = (c_1, d_1, c_2, d_2)$. We assume that during a communication all the packets pertaining to the communication use the same route. To model the communication network adaptability, due to the behavior of the communication protocols, a different route will be used whether the selected one becomes useless. The communication network can memorize its state, adapt the path, and route from one node to another on the basis of the current traffic. Therefore, the communication network can react autonomously to changes in the traffic workload. Its depends on the power grid, the transportation network, and the environment.

Transportation network. This infrastructure is a complex network itself, composed of thousands of links, characterized by different capacities and delays. As for the communication network, we simply model the transportation network, letting to future work the adoption of a more detailed model. We model the network as a routing matrix $R = \{r_{i,j}\}^{m \times m}$. The rows and the columns of the matrix are indexed by the system elements (agents). The matrix element $r_{i,j} = (c_1, d_1, c_2, d_2, ..., c_p, d_p)$ contains all the possible routes (if any) from i to j; for each route, the actual estimated capacity (c) and delay (d) are specified in the corresponding matrix element. The transportation network can memorize its state. We assume that it is a static network; therefore, it is unable to reconfigure itself. The behavior of the transportation network depends on the telecommunication network, the environment, and the power grid.

Power grid. This infrastructure is a complex network itself, composed of thousands of links and devices. Also in this case, we use a simple model of the power grid, letting to future work the study of a detailed model. We suppose that there are a set of generators that provide energy to the system elements. A generator may supply energy to all or to a subset of system elements. The power grid is modeled as a matrix $P = \{p_{i,j}\}^{s \times m}$. The rows of the matrix are indexed by the generator and the columns of the matrix are indexed by the system elements (agents). The matrix element $p_{i,j}$ is set to 1 if the system element j is supplied by the generator i, otherwise $p_{i,j} = 0$. The power grid can memorize its state and it is also capable to reconfigure itself (adaptation). It is autonomous and interacts with the other system components. The behavior of the power grid depends on the communication and transportation networks, and the environment.

Environment. The definition of a natural disaster model is definitely a challenge. This model is strictly dependent on the case study and requires a detailed simulation model that considers the spatial topology of the infrastructures. In this paper, we do not consider a natural disaster model, but we rather simply use a failure model described in the next section, supposing that failures can be originated as a consequence of a natural disaster. We also assume that our model implicitly considers a not well defined natural disaster scenario.

The failure model

Failures are one of the most important aspects to model in critical infrastructures. We suppose that there are two main types of failures: *temporary failures* and *permanent failures*. The first have a mean repair time at least one or two orders of magnitude lower than the simulation time. The subsystem restoring can be carried out automatically by the system or manually by an operator. Permanent failures have a mean repair time at least one or two order of magnitude greater than the simulation time and are used to model subsystem disruptions. Both types of failures are technology independent and can be hardware or software. However, the result of a failure is always the same: the system under failure will experiment an inactivity phase, by entering an "off state".

Each failure requires the definition of the distributions of the failure inter-arrival time, of the restore time and the choice of their parameters. We adopt a distributed failure model rather than a centralized one. This means that each agent has associated a failure generator, with its own distribution and parameters. This allow to have completely independent failures model on each subsystem.

THE CRITICAL INFRASTRUCTURE AGENT-BASED SIMULATOR

Our simulator is built using the Recursive Porus Agent Simulation Toolkit (Repast) [9], an agent modeling toolkit. Repast is a discrete event simulator, that has been developed to support the modeling of living social agents, but it can be also used for other goals. It offers a way to specify agents behavior and characteristics, a completely adaptable scheduler, that supports both sequential and parallel discrete event operations and that is responsible for the execution of agents behavior and the simulation task management (e.g., display updating, data recording). Moreover, Repast includes ProActive [6], a Java-based middleware for object- and component-oriented parallel, mobile, and distributed computing. ProActive allows to execute a simulation on a Globus computational Grid in such a way to distribute the load on different nodes.

In the simulator design, we have followed the from-scratch approach. We will explore in a future work the hybrid approach with the integration of network and power grid simulators.

Figure 5 shows the class diagram of our simulator (we omit methods and attributes for clarity). There is a simulation controller and scheduler, implemented by the Critical Infrastructure Agent-based Model (`CIABModel`) class, which is responsible for initiating and executing the actions of the agents in our model.

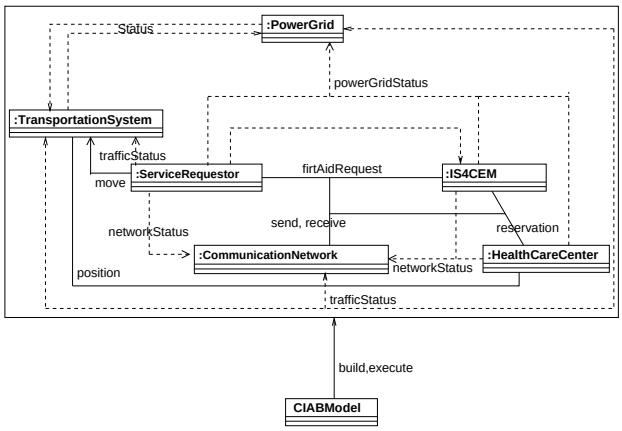

Figure 5. The class diagram representation of the simulator.

Each modeled agent is implemented by a Java class (see Table 1): the class methods implement the *model behavior*, and the execution of the model behavior is organized into the `pre-step`, `step`, and `post-step` methods of RePast. The class attributes implement the *model characteristics*, *status*, and *memory*; *adaptation* is modeled as an exception of the normal behavior, and managed using the standard exception handling mechanism provided by Java (RePast does not provide any specific support for handling exceptions).

Table 1. The Java implementation of the agent model.

Agent model	Java implementation
Behavior	Class methods
Memory	Class attributes
Status	Class attributes
Characteristics	Class attributes
Adaptation	Class exceptions

There are six main classes for the agents: `ServiceRequestor`, `IS4CEM`, `HealthCareCenter`, `CommunicationNetwork`, `TransportationNetwork`, and `PowerGrid`.

The association between/among classes is used to model the direct and explicit interactions. For example, the `ServiceRequestor` class has a bidirectional direct interaction with the `IS4CEM` and `CommunicationNetwork` classes. It also uses the `TransportationNetwork` class: in the class diagram, this unidirectional interaction is represented with a corresponding unidirectional association.

The dependencies among the classes are used to model the system interdependencies. The association between classes is used to model the direct interaction between agents. For example, the SR dependency on the power grid is represented by the unidirectional dependency between the `ServiceRequestor` and `PowerGrid` classes. The interdependency between the transportation network and the power grid is represented by a bidirectional dependency between the two corresponding classes. The use of the IS4CEM by a service requestor is represented by the `firstAidRequest` association, while the use of the communication network by the IS4CEM `send/receive` association.

In the following, we describe more in detail the classes behavior and exception handling.

CIABModel. It is the class controller of the simulation. It inherits all methods and attributes from the SimModelImpl class (the base class of Repast to build a model). The `CIABModel` is responsible for the model setup with the parameters provided by the simulator user; it builds the model during the simulation run and initializes all the agents in the model; it updates the simulation environment (e.g., the matrix of the communication network if a new agent is created); it builds the scheduler. At every tick (time step of the simulator), it executes the behavior of every agent on the basis of the implemented scheduler.

ServiceRequestorAgent. In our model, one or more SR agents can be executed in the same time. In the considered case study, the `ServiceRequestorAgent` can be in six different macro states, depending on its role. If the SR role is succorer, the possible states are: waiting for a first aid call, moving to a place (from the HCC to the location where the wounded citizen is or viceversa), providing first aid. If the SR role is wounded citizen, the possible status are: asking for help, waiting for help, moving to a place (e.g., from the location where the wounded is to the HCC). The behavior adaptation can be needed, depending on the system state. The exceptions are handled when the SR is in the moving to a place state or in the waiting for help state. For example, when in the moving to a place state, the SR normal behavior, that consists in using the route provided by the IS4CEM, can change if the SR decides to use a different route.

IS4CEMAgent. The IS4CEM can be in different states according to its role. We consider the following main roles: First Aid selector, Route selector, Emergency Information provider. The typical behavior is to receive a user request and to provide a response by using some algorithms and getting information from local or remote databases. The exception arises when there is no updated information to get a decision. In this case, the adaptation capability of the IS4CEM is exploited by executing a different algorithm that takes a decision on the basis of some historical data.

HealthCareCenterAgent. In our model, we can have one or more instances of the HCC agent, depending if we simulate a case with only one HCC or more HCCs. The main tasks of

the `HealthCareCenterAgent` are: to reply to IS4CEM requests by sending a first aid team and to serve the citizen requests. Its adaptation capability consists in the increase of the HCC capacity (using a larger staff) in case of emergency.

CommunicationNetworkAgent. Its main tasks consist in computing the packet delivery time and adapting the packet routing if the network topology or load condition changes. The adaptation capability depends on the routing algorithm used at the network level. The current network model is simple and we consider only a static adaptation capability. The use of a more sophisticated network model (e.g., by enabling the interaction with an external network simulator such as NS2 or Omnet) will allow us to consider a more detailed network model and dynamic network topologies.

TransportationNetworkAgent. Its main tasks are to compute the trip time from a source to a destination location, and to advertise exceptions such as route interruptions or disruptions and traffic jams. The transportation network is static and no adaptation capability is considered. The use of a more sophisticated network model (e.g., interacting with a transportation network and traffic external simulator) will allow us to model some traffic management policy.

PowerGridAgent. It simulates the power provisioning at different sites. The power grid, similarly to the communication network, should have a reconfiguration capability (even if limited), but we do not currently model it. The use of a more detailed power grid simulator will allow us to model exceptions and fault reaction policies. For example, we could model an automatic capacity replanning in case of overload or a failure detection and recovery policy.

CONCLUSIONS

During critical operating conditions, which are typical of natural disaster or terrorism attack scenarios, the technological, social, and natural system interdependencies pose the entire system under severe stress. To determine such interdependent effects and to study the effect of emergent phenomena, ABM&S seems to be the most appropriate technique.

In this paper, we have proposed an UML-based approach to model critical infrastructures and their interdependencies through agents. We have discussed the most relevant issues regarding the agent-based simulator design and implementation by considering as a case study the IS4CEM application. The simulator development offers us some degrees of flexibility to customize the target system according to the specific case under study. We can add or remove agents and/or infrastructures. For example, we can extend our case study by considering other actors that are currently generalized as service requestors, such as police men and fire men. At the contrary, we can simplify the current case study by eliminating the succorer role. Furthermore, the failure model and/or environment model can be changed. For example, by modeling the agent spatial position, we can refine the environment model by considering events such as a landslide or a flooding.

We have designed the current agent-based simulator for the IS4CEM application by following the from-scratch approach. In future work, we will consider the hybrid approach, as this choice will allow us to use more complex existing models of some system components such as the communication and transportation networks and the power grid.

REFERENCES

[1] I. Abele-Wigert and M. Dunn. *International CIIP Handbook*. Center for Security Studies, ETH Zurich, 2006.

[2] V. T. da Silva, R. Choren, and C. J. P. de Lucena. A UML Based Approach for Modeling and Implementing Multi-Agent Systems. In *Proc. of 3rd Int'l Conf. on Autonomous Agents and Multiagent Systems*, pages 914–921, 2004.

[3] S. Eubank. Social Networks and Epidemics. In *Proc. of Agent Based Modeling and Simulation Workshop*, pages 57–66, Nov. 2003.

[4] S. Filatyev, A. Chaturvedi, J. Gore, A. Mellema, and A. Hanna. Fire Evacuation Modelling Using Multiple, Bridged Simulations. Technical report, Purdue Homeland Security Institute, May 2005.

[5] O. Gursesli and A. Desrochers. Modeling Infrastructure Interdependencies using Petri Nets. In *Proc. of Int'l Conf. on Systems, Man and Cybernetics*, Oct. 2003.

[6] K. Hopkinson, X. Wang, R. Giovanini, J. Thorp, K. Birman, and D. Coury. EPOCHS: a Platform for Agent-based Electric Power and Communication Simulation Built from Commercial Off-the-shelf Components. *IEEE Trans. Power Systems*, 21(2):548–558, May 2006.

[7] K. Kavi, D. Kung, and H. Bhambhani. Extending UML for Modeling and Design of Multi-Agent Systems. In *Proc. of Int'l Workshop on Software Engineering for Large-Scale Multi-Agent Systems*, May 2003.

[8] R. Miles and K. Hamilton. *Learning UML 2.0*. O'Reilly, Apr. 2006.

[9] M. North, N. Collier, and J. Vos. Experiences Creating Three Implementations of the Repast Agent Modeling Toolkit. *ACM Trans. Model. Comput. Simul.*, 16(1):1–25, 2006.

[10] J. Odell, H. Parunak, and B. Bauer. Extending UML for Agents. In *Proc. of Agent-Oriented Information Systems Workshop*, pages 3–17, 2000.

[11] S. Rinaldi. Modeling and Simulating Critical Infrastructures and their Interdependencies. In *Proc. of 37th Annual Hawaii Int'l Conf. on System Sciences*, pages 5–8, Jan. 2004.

[12] S. Rinaldi, J. Peerenboom, and T. Kelly. Identifying, Understanding, and Analyzing Critical Infrastructure Interdependencies. *IEEE Control Systems Magazine*, 21(6):11–25, Dec. 2001.