Proceedings of the Federated Conference on
Computer Science and Information Systems pp. 659–665

ISBN 978-83-60810-22-4

SimConnector: An Approach to Testing Disaster-Alerting Systems Using Agent Based Simulation Models

Muaz Niazi
University of Stirling,
Scotland, UK
Email:
man@cs.stir.ac.uk

Qasim Siddique
Foundation University,
Islamabad
Email:
qasim_1987@hotmail.com

Amir Hussain,
University of Stirling,
Scotland, UK
Email:
ahu@cs.stir.ac.uk

Giancarlo Fortino
University of Calabria,
Italy
Email:
g.fortino@unical.it

***Abstract*—The design, development and testing of intelligent disaster detection and alerting systems pose a set of non-trivial problems. Not only are such systems difficult to design as they need to accurately predict real-world outcomes using a distributed sensing of various parameters, they also need to generate an optimal number of timely alerts when the actual disaster strikes. In this paper, we propose the SimConnector Emulator, a novel approach for the testing of real-world systems using agent-based simulations as a means of validation. The basic idea is to use agent-based simulations to generate event data to allow the testing of responses of the software system to real-time events. As proof of concept, we have developed a Forest Fire Disaster Detection and Alerting System, which uses Intelligent Decision Support based on an internationally recognized Fire rating index, namely the Fire Weather Index (FWI). Results of extensive testing demonstrate the effectiveness of the SimConnector approach for the development and testing of real-time applications, in general and disaster detection/alerting systems, in particular.**

I. Introduction

IF the past decade were to serve as an example, it is clear that it is perhaps impossible to be over-prepared for disasters. Regardless of the available technologies and preparations, disasters, natural or otherwise are known to hit any part of the world without any prior warning. At times, perhaps getting an early warning might be of little use but at others, even a few additional minutes of warning can make a difference in saving the lives of hundreds, if not thousands of human beings. The earlier a disaster can be detected, the higher the chance of saving precious lives.

However, intelligent detection of disasters is only part of the problem. The other problem is how to effectively test systems which have been developed specifically for responding to disasters. As an example, recently in the case of Mentawai tsunami, because of a lack of an efficient alerting system to go along with the already installed sophisticated detection system, a large number of casualties occurred. According to the Jakarta Post:

"Two days after the disaster struck, the death toll listed 282 and 411 were missing. The tsunami badly damaged 25,426 houses, flattened six hamlets and forced 4,500 residents to evacuate to makeshift shelters".

According to an official report:

"The most sophisticated system currently available needs five minutes to process information from an earthquake before issuing a tsunami warning — and issuing a command to respond to the field would take more than 15 minutes. It would have been too late for Mentawai."

Thus in the Mentawai case, there could perhaps be still possibility of saving a number of lives if there were an efficient early warning and alerting system, which would have generated alerts earlier on. So, if people were given say 10 minutes of warning using possibly almost real-time alerts, these might have helped save at least some of these lost lives. So, the key problem here is that while there were intelligent disaster alerting systems available for early warning, how to test such systems in the absence of actual events.

While any software system needs to be thoroughly tested before it is deployed, the case for testing an entire early warning system end-to-end is extremely strong. However software testing of user-driven software requires user testing. In the case of software driven into action primarily by real world events such as disasters, it can be fairly hard to replicate such events. If such an event is simulated manually by say, a person, generated the event, it is not guaranteed to test the system enough. Although considerable data collection exercises have previously taken place, to the best of our knowledge, there is no direct way of using this data to actually test such a system using it. As such, perhaps, the only way to test these applications is by the use of simulation of such rare events. While there are a number of simulation approaches, agent-based modeling and simulation is well-known to develop models of complex adaptive systems. As such, in this paper, we propose the Simulation Connector Emulator approach. The Simulation Connector approach was engineered in response to a practical problem faced in the testing of an actual forest fire alerting system. This approach is proposed as an advancement in Simulation-based Software Engineering by allowing testing of real-world systems using agent-based simulation models. The idea is based on using a validated agent-based model to simulate disaster events in

real time and subsequently using this real-time data stream to test the latency of the response times of the actual disaster alerting system. In other words, this approach assists in the testing of disaster response systems by allowing for testing scenarios using a simulation of rare and unpredictable real-world events. The parameters of the agent-based simulation can be subsequently validated[3], tweaked and calibrated to match with different disaster scenarios[4].

As proof of concept of the SimConnector approach, we have developed it in conjunction with a comprehensive forest fire detection and alerting system. Using an agent-based simulation model, we simulate forest fires and extract the Internationally recognized Fire Weather Index (FWI) [5] values from the simulation. The forest fires simulation has been developed to give extensive variability by catering for a large number of factors responsible for the duration and intensity of forest fires. These include tree cover re-growth, snowfall, rainfall, wind speed, fuel types and other parameters. Using the SimConnector approach, the simulation model is connected to the distributed forest fire alerting system; a system which generates alerts for forest professionals by making intelligent decisions based on the FWI values retrieved from the simulator in real-time. Our extensive simulation experiments demonstrate the usefulness and effectiveness of the proposed architecture in the design, development and testing of disaster-alerting systems. Results showing the reduced latency in the forest fire alerting system demonstrate the effectiveness of the proposed architecture in the testing of the disaster alerting for real-time monitoring of rare events.

The structure of rest of the paper is as follows:

In the next section, we provide the background followed by a discussion of the system architecture. In the next section we present a discussion of the simulation experiments and results followed by a conclusion.

II. Background And Related Work

In this section, we discuss basic background information and related work.

A. Tackling Disasters

Disasters can be classified in many different ways[6]. However, regardless of the specific type of disaster, recent examples of disasters around the globe clearly demonstrate the need of care in the designing of alerting systems. Designing software systems with real-time early warning and alerting capabilities can be challenging. These systems cannot be classified as traditional software systems due to a number of reasons. Firstly, such systems require both hardware as well as software to provide decision support to the human end users but they are unlike traditional embedded systems. Secondly, not only do they need to provide real-time monitoring and alerting capabilities, the events which these systems are expected to report are rare events. These events are rare because they may perhaps occur only once in the future of the system (or hopefully never). Thus, while the systems are required to be very robust, most times, there is not a whole lot that can be done to test these systems in real-world situations before an actual deployment. As such, in the absence of real test situations, it is logical that designers of such systems need to rely extensively on novel approaches such as simulation-based software engineering[7]. However, this approach is not as well studied in literature as is desirable[8] especially in this particular use case of disaster alerting systems.

B. Simulation Design of Disasters

Among the various approach to the simulation of disasters for the development of disaster alerting systems, designers are faced with two key possibilities:

- The first possibility is to develop a specialized simulator from the grounds up (e.g. using some high level language such as C++, Java or C# etc.).
- The second option is to use a general purpose methodology to develop a simulator.

While the first option is guaranteed to follow the design and development of the system closely, it can also lead to several setbacks. Firstly, such an approach can result in considerable extra costs, something which might not be available for disaster alerting systems, which are expected to be funded by public funds. Secondly, technically speaking, the reliability of custom-built simulators is, at best, debatable since they will essentially bypass a peer-review or community review, something which could have actually helped improve the design of the system in addition to the discovery and eventual removal of any software bugs. As a result, additional testing might be needed for testing the software raising the costs.

C. Choosing a modeling paradigm

In terms of using a general purpose modeling paradigm, it would be useful if the chosen paradigm qualifies based on certain specific requirements. Unlike modeling of traditional software engineered systems, the peculiar modeling paradigm might entail developing of system models capable of flexibility as well as the representation of complex human social systems as well as computing infrastructures. One such suitable paradigm is the agent-based modeling and simulation (ABM) paradigm[9]. ABM (or ABMS or IBM as it is called in various communities) reflects a specific modeling paradigm which has found extensive use in a large number of scientific domains as described in a recent visualization-based survey on agent-based computing[10]. It has previously been used successfully by researchers in domains as diverse as Biological systems[11], Ecology[12], Social Sciences[13], Business Systems[14] and Computer Sciences[15, 16]. It has been labeled a revolution in the esteemed Journal Proceedings of the National Academy of Sciences [17]. Agent-based models have also previously been shown to model the quantification of complex concepts such as emergence in complex adaptive systems [18]. ABM has also previously been used as a stand-alone simulation tool as

well as for decision support for non-technical Biologist end-users[19]. It has also found extensive use in the discovery of emergent behavior in systems consisting of a large number of interacting entities[20]. PASSIM, a simulation-driven methodology for building Multiagent Systems has been discussed in [21]. However, to the best of our knowledge, ABM has not been previously been used in the context of testing and evaluation of distributed disaster alerting systems.

Agent-based modeling is an advanced programming paradigm where the focus of the simulation design is on modeling individual entities as "agents". In other words, it allows for a more realistic modeling of real-world objects and complex systems. Agent-based development entails development of behaviors of individual agents, which change the internal state of the agents as well as interact with the environment (e.g. In the case of a programming environment agents might consist of patches, an abstraction quite commonly used in Logo-based agent-based modeling environments[22]).

D. Related Work

Rinaldi et al. discuss how models and simulations can provide considerable insight into the complex nature of the behaviors and operational characteristics of critical infrastructures [23]. Bodrozic et al. uses a multi-agent system for data retrieval and processing in forest fire monitoring [24]. Schaeffer et al. discusses the use of extensive simulations to give recommendation on tuning wireless sensor network parameters on the issue of false positive detection due to malfunctioning sensor networks [25]. Angayarkkani et al. approaches the detection of forest fires from spatial data using a combination of data mining, image processing and artificial intelligence techniques [26]. Torii et al. proposes a multi-layer cellular automata based socio-environmental simulation by layering the social interaction scenario on environmental simulation in the domain of fire fighter simulation[27]. One important result in this paper is the control of flow of information between two systems in developing various types of simulations. Hochrainer et al. examines catastrophe modeling and how it implicitly incorporates simulations due to the sheer complexity of the system to be analyzed [28]. Cossentio et al. describe how simulation can be useful in the design and development of multiagent systems [29]. Mata et al. present an approach to forest fire prediction using Case based reasoning[30].

III. System Architecture

This section presents the proposed architecture for the SimConnector in the light of an actual application system. We have chosen the application domain as Forest Fire Detection and Alerting. One key benefit of this domain is the availability of a large amount of historical data as well as a large number of previous studies on forest fire simulations.

The components of the proposed system are shown in Figure 1. The Forest Fire Detection System is made up of several key components such as two separate User Interfaces (web and Windows based applications) front ends, a backend database, a web service and the Simulation based Decision Support System (DSS). DSS is based on the SimConnector in addition to an Agent-based Simulation model for intelligent fire detection.
The individual components are given as follows:

1. Agent Based Model(Forest Fire Simulation)
2. Decision Support System and SimConnector
3. Windows UI & Web UI
4. Web Service

In the following subsections, we give details of these components.

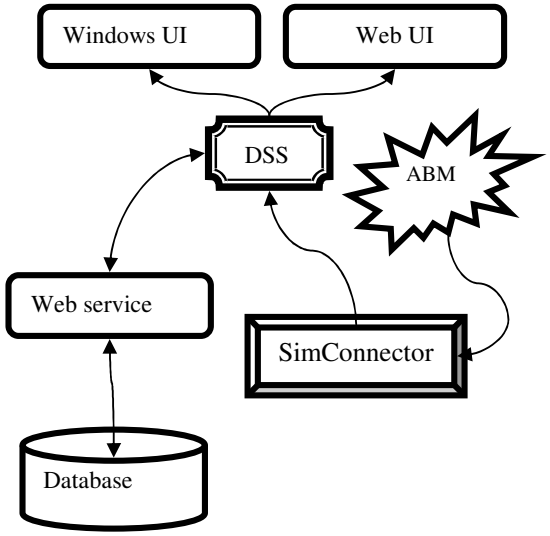

Figure 1 Main Components of Forest Fire Detection System

A. Agent Based Model (Forest Fire Simulation)

The basic implementation of this fire spread was extended from an open source NetLogo model[31]. However the model was extended significantly by adding the measurement of indices as well as numerous factors which can affect actual fires. This model was validated using a Virtual Overlay Multiagent system [4]. The model is based on a Cellular Automata (CA) model of a forest. As time progresses, fires appear in the model. Fires affect a random area based on specified spread rates. However the forest fires are validated using a Forest Fire standard index, namely the Fire Weather Index (FWI). FWI is a means of measuring fire intensity based on a number of different factors. FWI is calculated using complex mathematical calculations. One benefit of using FWI is that it is considered as a standard fire index in a large number of countries and has previously been used as a measure suitable for use by Wireless Sensor Networks for forest fire detection[32]. As soon as the fire occurs, a change in the local temperature is detected using a

simulated Wireless Sensor Network. Forest fire is also affected by snowfall and rainfall, which are also simulated and can be adjusted by means of various parameters.

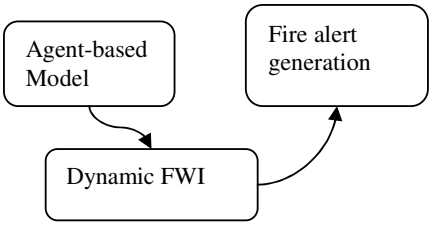

Figure 2 Main components of the Decision Support System

B. Decision Support System & SimConnector

The detection of forest fires in the proposed system is based on the Decision support system. The simulation environment has a number of distributed agents simulating wireless sensor nodes. These sensor nodes (agents) in the simulation environment detect parameters such as current temperature, rain, wind speed, average humidity subsequently used to calculate the FWI. The calculated FWI is sent over to the SimConnector. The SimConnector is responsible for communicating the fire information for onward generation of fire alerts as can be seen in Figure 2.

As the DSS sends fire information to the actual forest fire detection and alerting system, the desktop application depicts a graphical display of the placement of the sensors in the forest as well as the locations where forest fires have been intelligently detected.

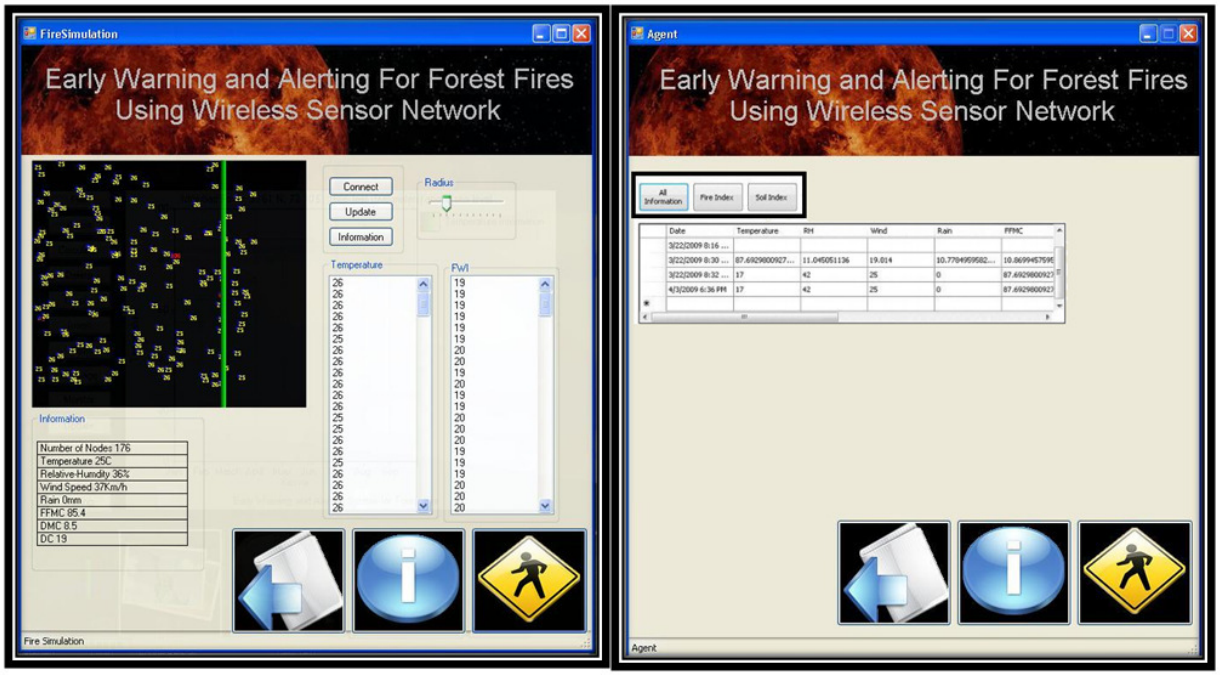

Figure 3(a) Windows UI detecting current location of the agent present in the model (b) UI demonstrating the various Fire Indices calculated dynamically.

C. Windows UI & Web UI

The design goal of the Web and windows Application are based on a continuous monitoring of the forest using the FWI as an indicator of fire. If the FWI value is detected as within a dangerous level, then the desktop application sends a warning message to the web application.

The Figure 3a shows the Windows User Interface. On the left side of the UI, we can see the location of various simulated wireless sensors. The moving green line represents the scanning of the intelligent DSS as it uses SimConnector to get data about specific coordinates. The update, connect and information buttons are used to select between a continuous update or for refreshing the system if continuous updates is not selected. The Windows UI also depicts the actual values of two parameters (the temperature and fire Weather index) for easy validation of fires by the user. All information is updated in the user interface on a periodic basis. The grid view at the lower left presents the average information retrieved from the sensor nodes deployed in the forest, the average temperature of the forest, average humidity, the wind speed and rain measurements thus

allowing the user to have a complete picture of the current situation in the simulated forest.

We can see the results of the calculated Fire indices in Figure 3b. The system also contains user interface for web and mobile devices for use by emergency support personnel accessing the website. This information is again tied in with information received via the intelligent decision support based on the FWI.

D. Web Service

To have a uniform API for data retrieval, a web service has been developed which connects to the database for storing and retrieving data. The API also allows the user to execute queries against stored FWI Information. Building a client for a web service is easy with the availability of automated tools such as Microsoft Visual Studio.NET which automatically generate client code based on a Web Service Descriptive Language (WSDL) page exported by the web service. By using a standard B2B interface, new clients can easily be developed in the future using other web services or else clients using other languages/technologies.

IV. Results & Discussion

In this section, we evaluate various aspects of the SimConnector implementation using extensive simulation experiments. The goal of using SimConnector is to allow for the performing of real-time black box testing evaluating the timing of the alerts generated by the system for comparison with the time when the forest fires are actually simulated.

A. Experimental design

The goal of the first set of simulation experiments is to empirically discover the latency in the fire detection. The latency measurement is the time taken from the simulated fire to the time the system actually generates alerts which are transmitted to the web and windows User Interfaces.

In addition, the forest fire area was also calculated so that a variety of forest types and scenarios can be evaluated. The calculations are based on finding the total number of trees in the simulation. Once the fire is ignited and the tree begins to burn out after a specific time interval, the number of tree burn out due to the fire allows for the calculation of the forest fire area as given below:

$$A_f = A_{f1} - A_{f2} \tag{1.1}$$

Here, in this equation A_f represents the area of the fire. The area is found by subtracting the area covered by the forest after the fire from the area before the fire. The duration of the fire is subsequently calculated with help of the fire start and the fire finish times. Using the actual time when the system detects the fire, we can then calculate the latency of the system in detecting the fire.

For the evaluation of various scenarios of forest fires, we need to calculate the rate of spread of forest fires as given in the following equation.

$$\frac{dS_f}{dt} = \lim_{t_f \to 0} \frac{A_f}{t_f} \tag{1.2}$$

Where S_f represents the spread of fire and A_f represents the area of the fire. Thus this equation demonstrates the rate of spread of a forest fire.

B. Simulation parameters

A few of the baseline simulation parameters using real forest data from the Islamabad Margalla hills forests are given below in Table 1.

Table 1 System Parameters of Fire simulation model

Parameter	Default Value
Fire delay	40 hrs
Default temperature	16 degrees C
Temperature gradient	±5
Relative Humidity	37%

Here fire delay is a general parameter which allows for calibration of the simulation based on forest fire data. The temperature gradient parameter gives the variation in the temperature (i.e. if the default temperature is say set to 16°C a maximum of 21°C and a minimum of 11°C would be simulated in different sections of the simulation. Another important parameter is relative humidity. Description of other parameters and calculations are given in [5].

C. Results and discussion

In Figure 4, we can see the various simulated forest fires used in the system. Here we can see the relationship between the duration of fires along with the loss of tree cover

(Measured in terms of cells, which represent a unit simulation area i.e. km^2). As can be observed from this plot, a large number of simulations were performed. Duration of simulated fires ranged from less than one hour to close to *12* simulated hours. The loss of tree cover ranged from less than $2000\,km^2$ to $12{,}000\,km^2$. Here, we can see that these random fires allowed covering a large array of scenarios. In other words, if the system performs well within required parameters in these fires, there is a high probability that the system will perform quite well in actual practice as well.

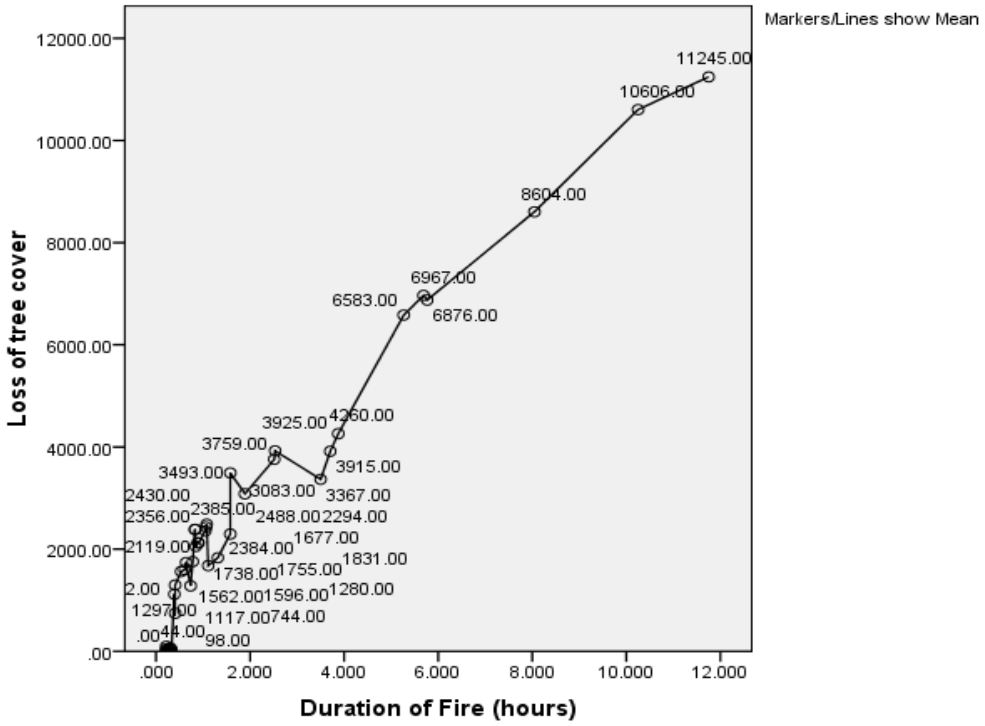

Figure 4 Plot shows the duration of various simulated fires plotted against the loss of tree cover

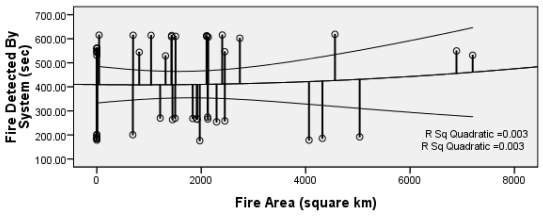

Figure 5 Latency in fire detection with a quadratic fit line plotted in 95% Confident Interval lines

In contrast to this graph, which can be considered as describing the random forest fires generated by the system for inputs to the SimConnector, Figure 5 represents the latency in the forest fires. The x-axis represents the forest fire area in square kilometer while the y-axis represents the actual latency in seconds. The plot itself is drawn in a 95% confidence interval along with quadratic fit lines. As can be seen, in general fire detection is possible in around 400 seconds up to a maximum of around 620 seconds, which amounts to around ten minutes. This is actually a very reasonable time since it is important that the forest fire should be stabilized and classified as a proper fire in contrast to perhaps a small fire which can die down on its own. If the system were to generate too many alerts, it would be completely useless since it would never be possible to find out when the actual fires were there. Domain experts confirmed that this is a reasonable time frame for the detection of forest fires.

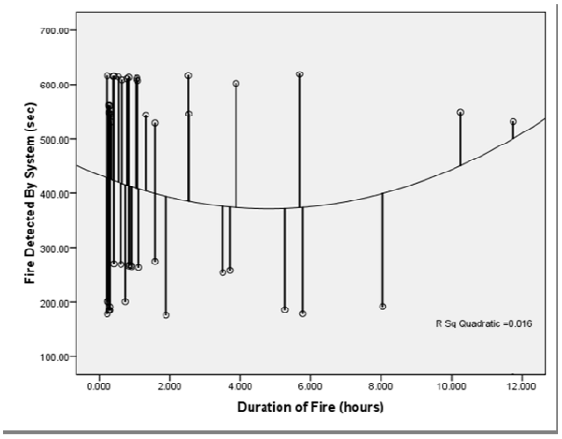

Figure 6 Fire duration vs. time taken by the system to detect and generate alarms

In Figure 6, we see a plot of the fire duration plotted against the latency of the proposed system in detecting fires. Here, we can observe that even for very small fires, the system aptly detects fires. While the fire may range from less than an hour to even *12* hours duration, the detection of the smaller fires is still efficiently performed by the system. By seeing the plot which also gives a quadratic fit line, we can observe the efficiency of the proposed system. Essentially detecting forest fires which are of shorter duration are important because personnel can be put on the lookout in the region just in case the fire goes out of hand. However, in real life, such fires are difficult to detect in large forests. The

proposed approach thus offers a better way of testing disaster alerting systems, in general, and forest fire systems in particular.

V. Conclusions and Future Work

In this paper, we have presented the SimConnector approach to using agent-based modeling for the testing of a real-world disaster alerting system. As proof of concept, we have applied this approach to a prototype application for the detection of forest fires. The architecture of the forest fire system consists of two user interface front ends, a web service, a backend database as well as an agent-based model. We have demonstrated the application of the SimConnector approach by implementing a system using an agent-based simulation model for the validation. Using a simulated Wireless Sensor Network in a forest fire, the system demonstrates the effectiveness of the SimConnector approach. The latency of fire discovery using intelligent decision support based on the international standard Fire Weather Index is minimized using the proposed approach. In addition, extensive simulation experiments also demonstrate the effectiveness of the approach in fires of both shorter durations as well as longer durations. The key goal of the system is to provide early detection of forest fires and generate alerts for forest fighters. The application allows users to monitor and test various parameters in the forest fire model such as intensity, location, date and time of the fire whereas the web application. While in this paper, we have developed the system in the domain of forest fires, we believe the approach is equally valid for develop other types of disaster early warning and alerting systems. In the future, we plan on demonstrating the SimConnector approach in other domains of disaster alerting systems.

References

[1] N. Kerle and C. Oppenheimer, "Satellite Remote Sensing as a Tool in Lahar Disaster Management," *Disasters,* vol. 26, pp. 140-160, 2002.

[2] S. B. Jb, "Mentawai tsunami death toll triples," in *The Jakarta Post*, ed. Padang, 2010.

[3] M. A. Niazi, *et al.*, "Verification &Validation of Agent Based Simulations using the VOMAS (Virtual Overlay Multi-agent System) approach," presented at the MAS&S 09 at Multi-Agent Logics, Languages, and Organisations Federated Workshops, Torino, Italy, 2009.

[4] M. Niazi, *et al.*, "Verification and Validation of an Agent-Based Forest Fire Simulation Model," presented at the SCS Spring Simulation Conference, Orlando, FL, USA, 2010.

[5] C. E. Van Wagner, "Development and structure of the Canadian forest fire weather index system," *Forestry technical report,* vol. 35, p. 37, 1987.

[6] W. H. Rutherford and J. de Boer, "The definition and classification of disasters," *Injury,* vol. 15, pp. 10-12, 1983.

[7] B. Boehm, "Software engineering economics," *Software Engineering, IEEE Transactions on,* pp. 4-21, 2009.

[8] L. Duclos, "Simulation cost model for the life-cycle of the software product: A quality assurance approach," *Dissertation Abstracts International Part B: Science and Engineering[DISS. ABST. INT. PT. B-SCI. & ENG.],* vol. 43, 1983.

[9] R. Axelrod, *The complexity of cooperation: agent-based models of competition and colloboration.* Princeton, NJ: Princeton University Press, 1997.

[10] M. Niazi and A. Hussain, "Agent-based Computing from Multi-agent Systems to Agent-Based Models: A Visual Survey," *Springer Scientometrics,* vol. In-press, 2011.

[11] R. Mukhopadhyay, *et al.*, "Promotion of variant human mammary epithelial cell outgrowth by ionizing radiation: an agent-based model supported by in vitro studies," *Breast Cancer Res,* vol. 12, p. R11, Feb 10 2010.

[12] D. Goulson, *et al.*, "Effects of land use at a landscape scale on bumblebee nest density and survival," *Journal of Applied Ecology,* vol. 47, pp. 1207-1215, 2010.

[13] V. Quera, *et al.*, "Flocking Behaviour: Agent-Based Simulation and Hierarchical Leadership," *Journal of Artificial Societies and Social Simulation,* vol. 13, p. 8, 2010.

[14] M. J. North and C. M. Macal, *Managing business complexity: discovering strategic solutions with agent-based modeling and simulation*: Oxford University Press, USA, 2007.

[15] M. A. Niazi and A. Hussain, "A Novel Agent-Based Simulation Framework for Sensing in Complex Adaptive Environments," *Sensors Journal, IEEE,* vol. 11, pp. 404-412, 2011.

[16] M. Niazi and A. Hussain, "Agent based tools for modeling and simulation of self-organization in peer-to-peer, ad-hoc and other complex networks," *IEEE Communications Magazine,* vol. 47(3), pp. 163 – 173, 2010.

[17] S. C. Bankes, "Agent-based modeling: A revolution?," vol. 99, ed: National Acad Sciences, 2002, pp. 7199-7200.

[18] M. A. Niazi and A. Hussain, "Sensing Emergence in Complex Systems," *IEEE Sensors Journal,* 2011.

[19] A. Siddiqa, *et al.*, "A new hybrid agent-based modeling & simulation decision support system for breast cancer data analysis," in *Information and Communication Technologies, 2009. ICICT '09. International Conference on*, 2009, pp. 134-139.

[20] S. Bandini, *et al.*, "Agent Based Modeling and Simulation: An Informatics Perspective," *Journal of Artificial Societies and Social Simulation,* vol. 12, p. 4, 10/31/ 2009.

[21] M. Cossentino, *et al.*, "PASSIM: a simulation-based process for the development of multi-agent systems," *International Journal of Agent-Oriented Software Engineering,* vol. 2, pp. 132-170, 2008.

[22] U. Wilensky, "NetLogo," *Center for Connected Learning Comp.-Based Modeling, Northwestern University,* vol. Evanston, IL, 1999.

[23] S. M. Rinaldi, "Modeling and simulating critical infrastructures and their interdependencies," in *System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on*, 2004, p. 8 pp.

[24] L. Bodrozic, *et al.*, "Agent based data collecting in a forest fire monitoring system," 2007, pp. 326-330.

[25] S. E. Schaeffer, *et al.*, "Decision making in distributed sensor networks," presented at the Proceedings of the Santa Fe Institute Complex Systems Summer School, Santa Fe, NM, USA, 2004.

[26] K. Angayarkkani and N. Radhakrishnan, "Efficient Forest Fire Detection System: A Spatial Data Mining and Image Processing Based Approach," *IJCSNS,* vol. 9, p. 100, 2009.

[27] D. Torii, *et al.*, "Layering social interaction scenarios on environmental simulation," *Multi-Agent and Multi-Agent-Based Simulation,* pp. 78-88, 2005.

[28] S. Hochrainer, "Catastrophe modeling and simulation," in *Macroeconomic Risk Management Against Natural Disasters*, ed: DUV, 2006, pp. 105-144.

[29] M. Cossentino, *et al.*, "Simulation-based design and evaluation of multi-agent systems," *Simulation Modelling Practice and Theory,* vol. 18, pp. 1425-1427, 2010.

[30] A. Mata, *et al.*, "Forest Fire Evolution Prediction Using a Hybrid Intelligent System," in *Balanced Automation Systems for Future Manufacturing Networks*. vol. 322, Á. Ortiz, *et al.*, Eds., ed: Springer Boston, 2010, pp. 64-71.

[31] U. Wilensky, "NetLogo Fire model," *ed. Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University,* 1997.

[32] M. Hafeeda and M. Bagheri, "Wireless sensor Network for early detection of forest fire," presented at the In Proc of the International Conferences of Mobile Adhoc and Sensor System, 2007.