17th World Conference on Earthquake Engineering, 17WCEE
Sendai, Japan - September 13th to 18th 2020

A Distributed Computational Tool for Natural Hazards Simulation

S.Y. Lin[1], A.W. Hlynka[2], L. Xu[3], H. Lu[4], O.A. Sediek[5], S. El-Tawil[6], V.R. Kamat[7], A. Prakash[8], C.C. Menassa[9], S.M.J. Spence[10], J.P. McCormick[11], B. Aguirre[12]

[1] *Ph.D. Student, Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109, USA; email: sylin@umich.edu*
[2] *Research Area Specialist Senior, Office of Research, University of Michigan, Ann Arbor, MI 48109, USA; email: ahlynka@umich.edu*
[3] *Ph.D. Student, Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109, USA; email: lichaox@umich.edu*
[4] *Ph.D. Student, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109, USA; email: harveylu@umich.edu*
[5] *Ph.D. Student, Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109, USA; email: osediek@umich.edu*
[6] *Professor, Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109, USA; email: eltawil@umich.edu*
[7] *Professor, Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109, USA; email: vkamat@umich.edu*
[8] *Professor, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109, USA; email: aprakash@umich.edu*
[9] *Associate Professor, Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109, USA; email: menassa@umich.edu*
[10] *Assistant Professor, Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109, USA; email: smjs@umich.edu*
[11] *Associate Professor, Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109, USA; email: jpmccorm@umich.edu*
[12] *Emeritus Professor, Department of Sociology and Criminal Justice, University of Delaware, Newark, DE 19716, USA; email: aguirre@udel.edu*

Abstract

A novel distributed computing software solution, Simple Real-Time Infrastructure (SRTI), is developed to model the interdependencies that arise in community resilience simulations. SRTI is based on a client-server structure and uses a publish-subscribe pattern to realize data communication between the distributed simulators for different fields. This approach provides a scalable, versatile, and user-friendly solution for integrating multiple discipline-specific models in hazards simulation. Each simulator is treated as a black box that interacts through the SRTI distributed computational platform. In this article, the SRTI is described comprehensively from the high-level structure to its fundamental components including the RTI Server, RTI Lib API, data structure, and the graphical user interface (GUI), as well as optional methods to utilize this framework and its components. Also, several ongoing important updates and improvements of the SRTI are introduced. Lastly, two example implementations that consist of a group of developed simulators for assessing the impacts of earthquakes on community resilience are presented.

Keywords: distributed simulation, hazard simulation, disaster interdependencies modeling, community resilience

1. Introduction

Modeling seismic damage and the following restoration process requires contributions from many specialized disciplines. Transcending the disciplinary boundaries is not a trivial task given the great differences in the types of models used in disparate fields. The problem is complicated by the differing time scales for both phases of the earthquake hazard scenario, i.e. seconds or minutes as the hazard unfolds versus days or months as the recovery process takes place. In addition, it is challenging to model the interdependencies that occur over various spatial scales, e.g. component (meters) versus system level (kilometers), within an integrated framework [1].

Motivated by these needs, a versatile computational tool, “Simple Real-Time Infrastructure” (SRTI), which allows researchers from different backgrounds and fields to link their computational models together to study the effects of natural hazards on community resilience, is developed and presented in this paper. SRTI is based on a client-server structure. The data communication between the distinct simulators in different fields is done through sockets using a publish-subscribe pattern. Each simulator is treated as a black box that interacts through the SRTI distributed computational platform. It provides a scalable, versatile, and user-friendly solution for integrating discipline-specific models in hazards simulation.

Some key features of the SRTI include: 1) it is open-source and free to use, 2) it permits distributed simulations across multiple machines, 3) it provides pre-compiled components to be downloaded and directly used without installation, 4) data messages are parsed as "strings" (JSON format), to allow flexible data representation without re-compilation, 5) it supports different programming languages and different computer systems, and 6) it achieves extensibility and scalability for complex systems in hazard simulation. Beyond this, optional components and extended versions of the SRTI have been developed for specialized functionality.

In the following sections, the high-level architecture of the SRTI and its fundamental components are detailed. Several ongoing important updates and improvements of SRTI are introduced. Lastly, two example implementations that consists of a group of developed simulators for assessing the impacts of earthquakes on community resilience are presented.

2. Simple Real-Time Infrastructure (SRTI)

2.1 High-level architecture and fundamental components

As shown in Fig. 1, the structure of the SRTI v1.00.00 consists of three key components: the RTI Server, the RTI Lib, and the users’ simulators. Data communication in the SRTI is based on a client-server structure and is done through sockets. The RTI Server is a server-side application that acts as the shared connection point for simulators in a simulation system. It can be run either on the same machine or a separate computer from the individual simulators. Before any simulator can connect to the simulation system, the RTI Server must be launched first and its “hostname” and “portnumber” are required to be referenced by the simulators to connect to it.

The RTI Lib is a client-side API (Application Programming Interface) library that allows a simulator to connect to an RTI Server. The precompiled RTI Lib API must be locally stored and referenced on the same machine as the simulator. The details of socket communication, message creation and parsing are handled within the RTI Lib. Through the use of the RTI Lib API, the connection to the RTI Server and the transmission of messages can be done in a simple manner.

The simulators here refer to the computational models provided by a user that may connect to the SRTI Server. The user needs to modify their simulators to reference the RTI Lib and its API as required. The source code, pre-compiled libraries, documentation, and other information associated with the SRTI are available at [2] and [3].

The RTI Server and RTI Lib are both precompiled and can be downloaded by the user. The RTI Server was written in Java and can be run on systems with the free Java Runtime Environment installed. At this time, the RTI Lib API is available in native Java and C++ so that the simulators written in Java, C++, and Matlab are supported. Both the RTI Server and RTI Lib API are also available as source code and can be rewritten in other compatible languages that support JSON parsing and socket communication.

SRTI has an optional component, i.e. the RTI Server GUI (Fig. 1). It is a graphical user interface that automatically opens when launching the RTI Server. It provides a visual window that describes the “hostname” and “portnumber” of the RTI Server as well as a list of the connected simulators. User can also inspect a live feed and history of data messages received by the RTI Server through the RTI Server GUI.

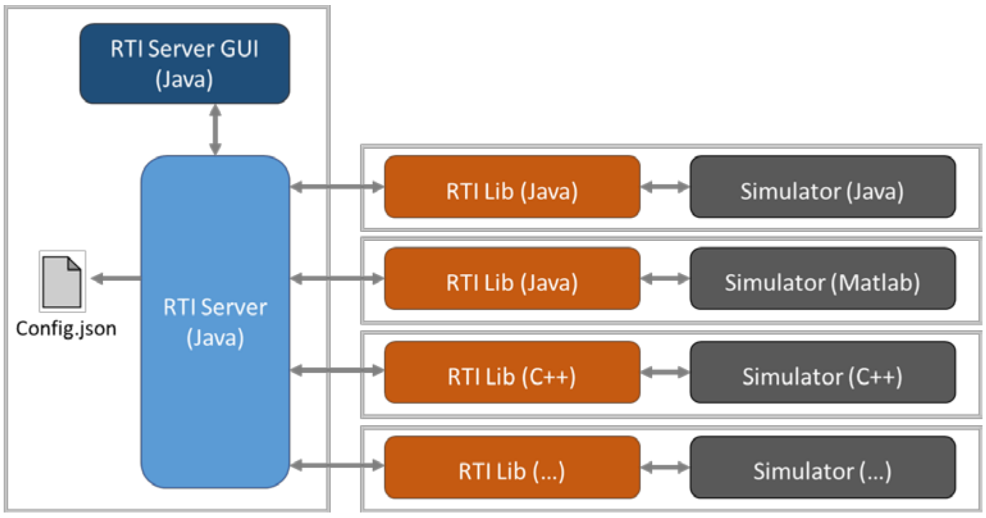

Fig. 1 – High-level architecture of the SRTI v1.00.00

2.2 Message transmission pattern and data format

Message transmission in the SRTI relies on a “publish/subscribe” pattern based on the framework described in [1] and [4]. Each simulator publishes its results in a “message”, and other simulators, which need the information, subscribe and receive the published messages. The RTI Server will listen for any incoming messages, and immediately forward the message to any simulator subscribed to that message. After subscribing to a given message, the RTI Lib API will keep listening for that message while the simulation runs, and it will store the message in a local buffer queue until the message is requested and used by the simulator [2, 3, 5].

In the SRTI, all elements of a message are parsed as a single JSON object, which is a readable standard of data representation and can be sent and received as String data. From the user’s perspective, a message must include two key components: “name” and “content,” which respectively refers to the name of the message and the data that a simulator wants to embed in the message. Beyond the name and content, a message must also have the other elements regarding its internal properties, such as “source” (the name of the simulator that published the message) and “timestamp” (the system time when the message was published). These elements can be accessed by the user’s simulator if required.

The RTI Lib API includes functions to help the user create a String value that represents such an object. The “content” would be capable of representing multiple variables within a single message, including integer numbers, floating point numbers, Boolean values, strings, and multi-dimensional arrays. The exact format of the “content” of the message does not need to be pre-defined for the SRTI to receive and send it to the connected simulators, which allows flexible data representation without re-compilation.

3. Ongoing improvements of the SRTI

As discussed above, the SRTI v1.00.00 is a distributed simulation tool permitting users to have high flexibility and control of data transmission. This feature allows it to be adaptable for both simple and complex simulations. However, the responsibility of understanding the name and content of the message falls on the user, as well as ensuring the time-related logic between simulators. To better support the functionality of time management and improve the user experience, a specialized update to the SRTI has been developed, tentatively named SRTI v2.00.00. Users can choose either v1.00.00 or v2.00.00 depending on their needs.

As illustrated in a high-level concept schematic (Fig. 2), an RTI Manager coupled with the original RTI Server is used and additional RTI Wrappers are added to the original data-communication structure to take over much of the aforementioned duty of the users. This approach can greatly reduce the coding effort of the user while they build a large-scale simulation system. Moreover, an optional graphical interface, i.e. SRTI v2 GUI, is developed to help users construct complex simulations more quickly and easily. The SRTI v2 GUI also allows users to execute the RTI Server and simulators from the same location. An example view of the SRTI v2 GUI is shown in Fig. 3

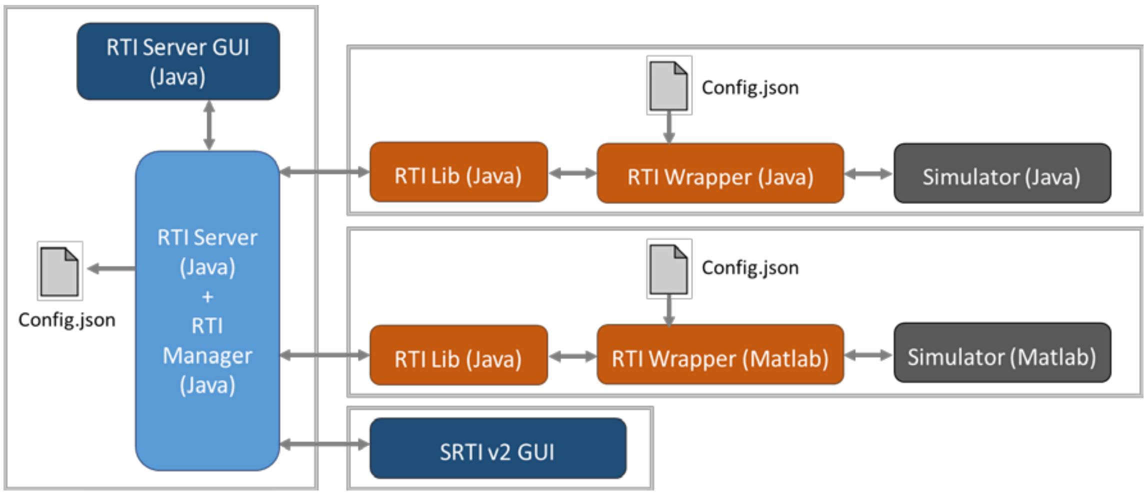

Fig. 2 – High-level architecture of the SRTI v2.00.00

4. Example Implementations

Several applications of the SRTI in hazard simulation have been developed by Project *ICoR: Interdependencies in Community Resilience* [6], such as hurricane and seismic scenarios. For better reusability and extensibility, the discipline-specific models in these simulations are designed to be highly modular, and communicate through the SRTI in a plug-and-play sense. One example implementation of the SRTI in a seismic scenario is the framework described in [7], which quantifies the resilience of communities after a severe earthquake. The schematic diagram of the framework is illustrated in Fig. 4, and more details about the simulators and simulation results are available in [7] and [8]. Another application of the SRTI in assessing seismic resilience is the framework (Fig. 5) proposed for time-dependent analyses of interdependent lifeline systems subjected to successive earthquakes [1]. The description of the simulators, messages, and other details have been documented in [1] and [9].

As shown in Fig.4 and Fig.5, both aforementioned frameworks consist of a set of simulators, each of which represents one aspect of a seismic scenario. The communication between the simulators is permitted by the publish-subscribe data management method through the SRTI. Each simulator is considered as a black box that receives inputs from other simulators and provides outputs in a specific format to be used by others. Therefore, any simulator can be replaced with another, which may be designed for a different fidelity or even based on different theory, without affecting the other. Such features highlight the flexibility and scalability of the SRTI.

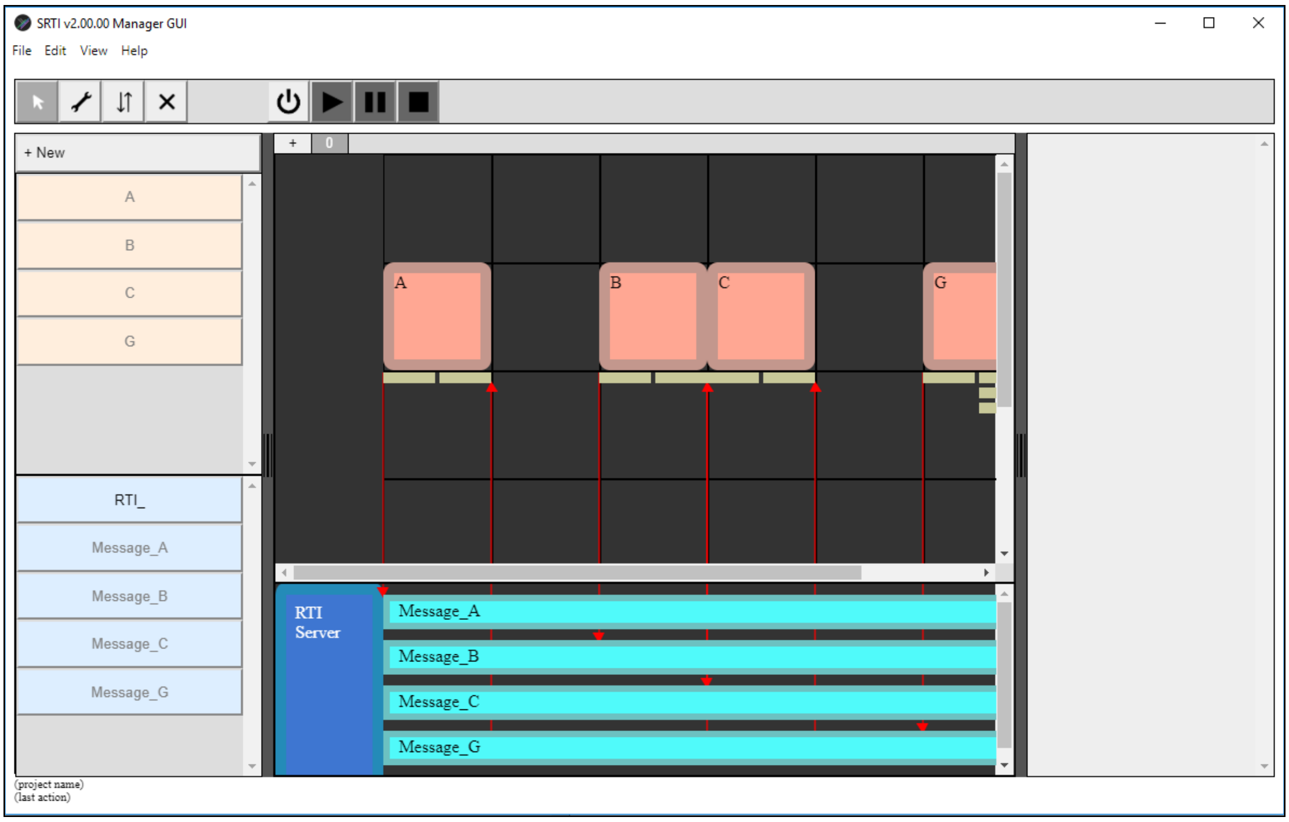

Fig. 3 – Example view of the SRTI v2 GUI

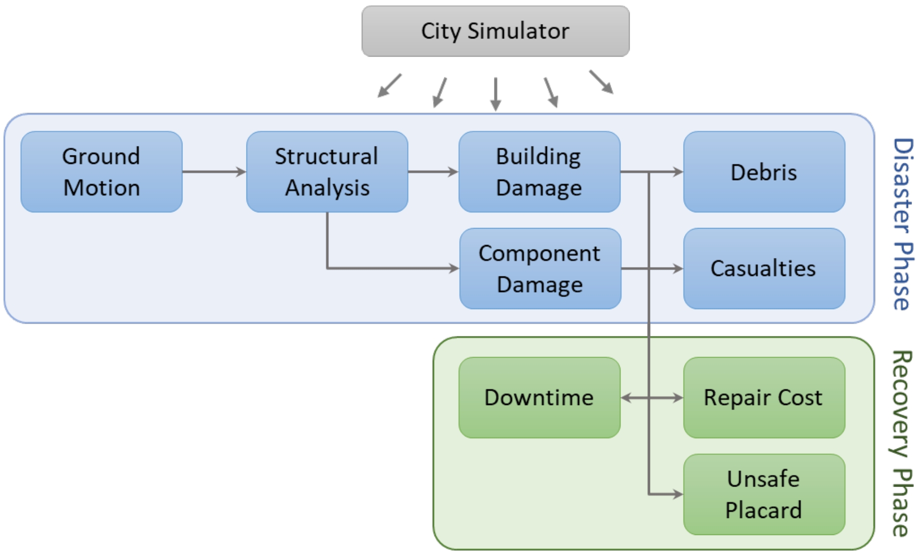

Fig. 4 – Seismic resilience assessment framework proposed by Sediek et al. [7, 8]

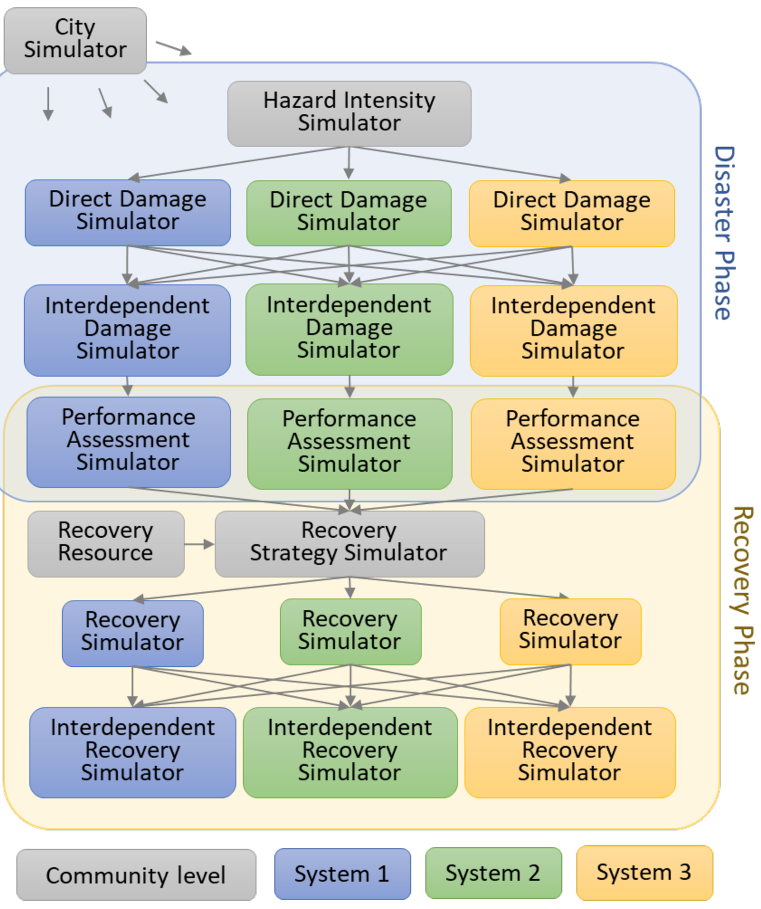
City Simulator
Hazard Intensity Simulator
Direct Damage Simulator
Direct Damage Simulator
Direct Damage Simulator
Interdependent Damage Simulator
Interdependent Damage Simulator
Interdependent Damage Simulator
Performance Assessment Simulator
Performance Assessment Simulator
Performance Assessment Simulator
Disaster Phase
Recovery Resource
Recovery Strategy Simulator
Recovery Simulator
Recovery Simulator
Recovery Simulator
Interdependent Recovery Simulator
Interdependent Recovery Simulator
Interdependent Recovery Simulator
Recovery Phase
Community level
System 1
System 2
System 3

Fig. 5 – Seismic resilience assessment framework proposed by Lin & El-Tawil [1, 9]

5. Conclusion

SRTI is a computational tool for distributed data communication between distinct simulation models used in natural hazards research. It is open-source and free to use, and permits distributed simulation across multiple machines. The extensibility and scalability of the SRTI make it versatile and well-suited for complex simulations. The key features of the SRTI include providing pre-compiled components to be downloaded and used without the requirement of installation or re-compilation, as well as supporting multiple programming languages and different computer systems. The high-level concept, fundamental components, and some ongoing improvements of the SRTI were discussed and two example implementations were presented to show the ability and flexibility of SRTI.

6. Acknowledgements

This research was supported by the University of Michigan and the US National Science Foundation (NSF) through grants ACI-1638186.

7. References

[1] Lin SY, El-Tawil S (2020): Time-Dependent Resilience Assessment of Seismic Damage and Restoration of Interdependent Lifeline Systems. *Journal of Infrastructure Systems*, 26(1): 04019040. https://doi.org/10.1061/(ASCE)IS.1943-555X.0000522

[2] SRTI (2019): Simple Real Time Interface (SRTI). https://github.com/hlynka-a/SRTI (Accessed Dec. 15, 2019)

[3] Lin SY, Hlynka AW, Xu L, Lu H, El-Tawil S, Kamat VR, Prakash A, Menassa CC, Spence SMJ, McCormick J, Aguirre B. Simple Real-Time Infrastructure (SRTI): A Distributed Computing Software Solution for Simulating Natural Hazards Scenarios. *Natural Hazards Review*, under review.

[4] Lin SY, Chuang WC, Xu L, El-Tawil S, Spence SMJ, Kamat VR, Menassa CC, and McCormick J (2019): Framework for Modeling Interdependent Effects in Natural Disasters: Application to Wind Engineering. *Journal of Structural Engineering*, **145** (5): 04019025. https://doi.org/10.1061/(ASCE)ST.1943-541X.0002310

[5] Abdelhady, AU, Lin, SY, Xu L, Sediek OA, Hlynka AW, El-Tawil S, Spence SMJ, McCormick J, Kamat VR, Menassa CC (2019): A Distributed Computing Platform for Community Resilience Estimation. Proceeding of the *13th International Conference on Applications of Statistics and Probability in Civil Engineering, ICASP13*, Seoul, South Korea, May 26-30, 2019.

[6] ICoR (2019): Interdependencies in Community Resilience (ICoR) Project. https://icor.engin.umich.edu/ (Accessed Dec. 01, 2019).

[7] Sediek OA, El-Tawil S, McCormick J (2019): A Scalable Framework for Assessing Seismic Resilience of Communities. *Structures Congress 2019*, Orlando, Florida, USA, April 24–27, 2019.

[8] Sediek OA, El-Tawil S, McCormick J (2019): Quantifying the Seismic Resilience of Communities: A Distributed Computing Framework. *International Conference in Commemoration of 20th Anniversary of the 1999 Chi-Chi Earthquake*, Taipei, Taiwan, September 15-19, 2019.

[9] Lin SY, El-Tawil S (2019): Time-Dependent Computation of Multiscale Interdependencies between Lifeline Systems Subjected to Seismic Events. *International Conference in Commemoration of 20th Anniversary of the 1999 Chi-Chi Earthquake*, Taipei, Taiwan, September 15-19, 2019.