# Virtual and Actual Testing of Engineering Structures

**Yan Xiao**

**Dept. of Civil Eng., University of Southern California & Center for Integrated Protection Research of Engineering Structures (CIPRES), Hunan University, China**

**Qing Hu**

**CIPRES, Hunan University, China & FutureNet Technologies Corporation, Monrovia, USA**

**Yurong Guo**

**CIPRES, Hunan University, China**

**P.S. Zhu, Dept. of Civil Eng., University of Southern California**

## INTRODUCTION

The recent rapid development of Internet technology provides new opportunities of revolution in structural experimentation. The Internet based communication enables the transfer and sharing of vast data and particularly the control signals, feedback information among geographically separated laboratories and facilities in an almost real time fashion. Scientific laboratories thus networked can then be utilized for testing more complicated systems to advance our knowledge. Several attempts have been made to carry out collaborative tests on structural systems with remotely located laboratories connected by Internet. One of the most ambitious efforts might be the Network for Earthquake Engineering Simulation (NEES) project, which invested several costly facilities and established a network.

To facilitate the structural research collaborations between the University of Southern California (USC) and the Hunan University (HNU) of China as well as other institutes, a research effort was initiated in 2000 to establish an Internet based network platform, now named as Networked Structural Laboratories, or Networked Scientific Laboratories, and abbreviated as NetSLab. The initial success of the effort attracted a national key project funded by the National Natural Science Foundation of China (NSFC) to the Hunan University. This paper presents an outcome of this project on development of the network platform NetSLab. Basic concepts and features of the platform developed for remote pseudo dynamic testing of substructures and structural elements are described along with the results of tests of bridge pile and pier column system subjected to simulated earthquake input.

## PROPOSED DATA MODEL

The application target of the network platform is for earthquake response simulation of structures over several remotely located laboratories and computers, connected by Internet. Such distributed structural testing can be organized as a series of steps. In one step, each participant completes one event according to the mutual communication among the participants, generates and sends additional result data. In other words, a step can contain several events. In order to make this process work accordingly, some kinds of controller should be introduced. The controller organizes the testing procedure and coordinates the working flow of all participates as well as the data flow in the testing. The authors proposed a generalized data model to abstract such distributed testing procedures and attempted to cover all possible situations in this area. The data model consists of the following three components: *testing results*; *three types of participants*; *communication protocols and dataflow*.

## Testing Results

The ultimate goal of a distributed structure testing is to obtain a series of testing results. According to the proposed data model, the testing results are represented by an array, *TR*, such as,

$$TR=TR(i,j),\ i=1,\ N;\ j=1,\ M \qquad (1)$$

where, $N$ is the maximum number of the steps and $M$ is maximum number of the events within a step. Each row of the array represents the testing results in one step, and each element of a row represents the testing result of a testing event, i.e. $TR(i,j)$ is the result from the step $i$ and event $j$. Note that, $TR(i,j)$ may depend on the results from previous stages. By this definition, the testing process can be considered as how to generate the *TR*, according to certain initial conditions and certain roles in the proposed data model.

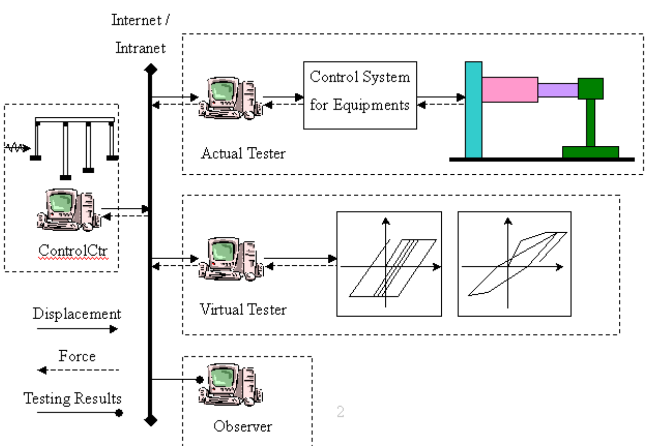


**Fig.1 Participants of NetSLab**

## Three Types of Participants

Based on the definition for *TR*, the participants' tasks in a testing are processing the *TR* with various different means. By their roles in the testing, all participants can be categorized into three groups, as exhibited in Fig.1:

1) Controller: who organizes the testing procedure, controls the testing progress and data communication among all participants, and stores or publishes the testing results. There is only one controller in one testing for current study.
2) Testers: who carry out the detailed task event(s) for each testing step to create new results according to previous results. There may be multiple testers in one testing. The testers can further be categorized into two types: virtual testers, who use computation to provide analytical *TR*, and physical testers, who operate the actual testing equipment to generate results, *TR*.
3) Observers: who monitor the process and share the testing results without any interference of the test processing.

## COMMUNICATION PROTOCOLS AND DATAFLOW

The data to be circulated among all participants correspond to all kinds of testing results and control information. In order to keep the testing running smoothly and efficiently, it is very important to specify how these data to be circulated. We used the Finite State Automata (FSA) model to describe the dataflow during the testing process. In general, an FSA is decided by its state transfer function, defined as,

$$F(C, u)=N \qquad (2)$$

where, $C$ is the current state, $u$ is input and $N$ is the next state.

In the proposed data model, the state transfer function $F$ is defined as,

$$F(C, u, H)= (N, \square) \qquad (3)$$

where $H$ is the historical data (partial *TR*), i.e. each node has a certain memory to maintain some historical data, and □ is an action which specifies what to send and send to whom.

According to the definition and properties of action, three types of data communication protocols are introduced in the proposed data model:

1) Centralized sequential communication: controller sends control data (called request) to each tester one by one and a tester responds back to controller only after it finishes controller's request. The order of data transfer is pre-determined, i.e. $F$ is independent to $u$ and $H$, and at any moment, only one peer to communicate.
2) Centralized parallel communication: controller sends requests to multiple testers simultaneously and waits for their response. As soon as obtaining the responses from the testers, the controller decides and starts next step to send out next round of requests.
3) Distributed parallel communication: controller only sends out the start signals and initial data to certain testers. When a tester receives the partial *TR* and request, it executes certain action and decides its own action to send request and testing result to someone else. At certain point, a tester may report the results back to the controller.

## IMPLEMENTATION OF NETSLAB

Since the final testing system will be used by many non-computer professionals for many different needs of structural testing, it is very important to keep the simplicity of the platform for understanding, developing and maintenance. Many tests may require a near real-time response which also requires the final system to operate as quickly as possible under the given network environment. Therefore, we introduced two new concepts to fulfill these goals.

1) Dynamic Unified Data Packet (DUDP). This data packet covers all communication related components: current stage data of *TR*, controller's request, testers' current status and communication requirements, etc. With such concept, only one kind of data packet is needed for traveling among all participants, providing the simplicity and reliability to the system. DUDP Parser is the software module to parse a given DUDP and to help any entity (controller, tester or observer) understand DUDP efficiently and correctly.

2) Generalized Data Communication Agency (GDCA). This is the unique communication components in our testing system, which can be attached to its master entity, such as a controller, testers or observers. It accepts DUDP from its master entity, sends it to the given destination and activates the corresponding entity if necessary. On the other hand, any incoming DUDP also goes through this agency to activate its master entity.

Based on these two concepts, data communication in a testing procedure becomes clear and simple. Any entity, whether it is a controller, tester or observer, always contains a DUDP Parser and a GDCA. For a tester, it is always waiting there, and an incoming DUDP or a user command can activate this entity. It then parses the DUDP to understand the request and corresponding data, carries out all necessary operations (including using the actual testing equipment or theoretical model with the input data to create testing results *TR*), determines the destine entity, generates the new DUDP and submits it to GDCA. All the testers have the same procedure for data communication and the differences are their own operations. This scheme is very helpful for simplicity, reliability and efficiency. The controller also can be established on the tester model. Besides the tester's function, the controller has more functions, such as, designating the participants for a given testing, organizing the communication protocol, specifying the initial conditions, evaluating testing results and deciding the termination of the testing.

In NetSLab, DUDP is represented by an XML formatted structure data, which provides a good flexibility for data representation, well defined data format standard and a large number of processing tools. The proposed GDCA can be considered as an ISO layer 7 protocol, which has both data presentation and communication function. GDCA receives the standard XML formatted data DUDP from the application programs. GDCA can be configurable to support the three communication models mentioned previously. The implementation of GDCA is fully open to system designers. In NetSLab, GDCA has been implemented based on a general interface engine, UniPipe, which provides many sophisticated lower level communication functions and a very flexible ActiveX control (including event sink) interface. Using UniPipe's diagram based script language, Action Tree, the design and maintenance of GDCA become much easier. As an ActiveX controller, it naturally supports any application program language with standard ActiveX function, such as VB, VC++, VJ, etc. Other issues, such as data repository, fault tolerance, security and multimedia capability are also considered in NetSLab.

## APPLICATION PROGRAMMING ISSUES OF NETSLAB

### Network Environment

The development of NetSLab is targeted to the flexibility and suitability in various network environments. This is considered to be one of the important factors for the technology transfer and the promotion for wide applications. A simple speed testing software was developed to test the data transfer speed using GDCA. At one end of the network, the program generates and sends a series of data for variable, *x*, and records the return time needed for the data of another variable *y*, calculated as the function of *x* at the other end of the network. Results of speed testing for roundtrip data transfer over different network environments are shown in Fig.2. As shown in Fig.1, the transfer speeds are fairly smooth with an average below 0.08 sec, for the networks of 1Mbps to 10Mbps, which are the environments within or among most academic units in the US and China. Interesting data were also recorded for the situation of a PC connected from a hotel room in Changsha, China to another PC located at USC using a slow dialup connection with a speed of only 24kbps. The roundtrip speed fluctuated between 0.4 to 0.7sec, however was not too drifted from the recorded average speed of about 0.4sec based on the MS Ping test. This may be considered as the possible worst-case scenario of NetSLab application, in terms of network speed.

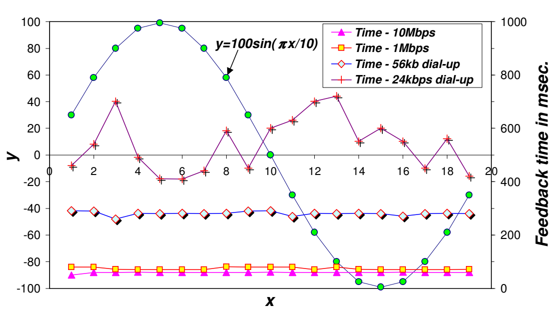


**Fig.2. Speed test results in different network environments.**

### Remote Simulation Analysis of Earthquake Response – Virtual Test

Non-linear time history analysis is generally accepted as the best method to study the earthquake response of a structure. However, due to its complexity and large amount of computation, this method is currently mainly used in research or earthquake designs of some important and complex structures. For a non-linear analysis program, one of the most difficult parts is how to construct the hysteretic models for various structural elements. Since different types of structural members (beams, columns, and walls) use different materials and have many differences between their characteristics of restoring force, there exist many types of models to simulate earthquake response of structures. To develop a non-linear time history program, one needs to construct analysis model for structural system, select appropriate numerical integration method and equation solver, and also deal with structure hysteretic models. Based on the characteristics of non-linear time history analysis, we can separate the hysteretic modeling from the main program. Computers with programs handling one or several types of hysteretic modeling is considered as virtual testers in NetSLab, and can be located remotely. A remote analysis to simulate the earthquake response of a structure can be considered as a virtual remote testing.

**Remote Testing of Structures**

Trial applications of NetSLab were carried out with development of several specific application programs for simulating earthquake response of structures. Current study is a collaboration effort between the University of Southern California and the Hunan University, both have adequate structural testing facilities. One of the initial trial applications of NetSLab for remote pseudo dynamic testing was carried out using the large-scale testing frame at USC on a full-scale model of precast prestressed concrete pile to pile-cap assembly. The tests successfully validated the adequacy of NetSLab.

Currently, one of the applications of NetSLab is to simulate the system responses of single column bent bridges to transverse earthquake motions. The experimental concept for simulating the earthquake response is shown in Fig.3. The super structure is analyzed as a concentric mass which includes all the tributary masses from the neighboring spans. The bridge column is simulated by analysis which includes a hysteresis model and the lateral force and deformation envelop.

Currently, the authors are taking an endeavor to collaborate with researchers at the Taiwan Uiversity to carry out the first bridge testing using NetSLab and the platform developed by Taiwan University researchers. A model bridge will be tested on the Internet based platforms virtually across the Taiwan Strait, however using the testing facilities at both Taiwan University and at the Hunan University.

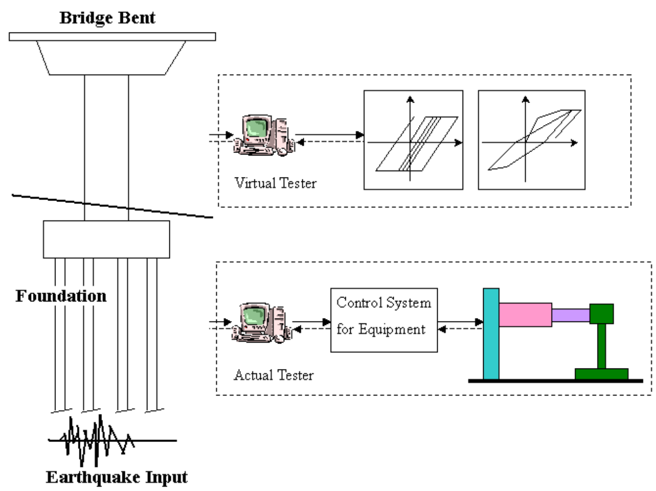


**Fig.3. Experimental concept with virtual and actual testers**

## CONCLUDING REMARKS

A network platform (NetSLab) was developed for remote response simulation of structures and structural elements under earthquake inputs, among geographically distributed structural laboratories or computer facilities. The adequacy of the NetSLab platform was validated by several remote virtual and actual tests at and between two universities in the US and China. NetSLab provides researchers a powerful tool to conduct research with efficiently shared use of laboratory resources. One of the most significances of current research may be to show a role model to avoid or eliminate the redundant investment of laboratory facilities.

(Acknowledgements--Multiple sources of supports were received from the National Natural Science Foundation (NSFC) under the National Key Project (Grant No. 50338020); the Cheung Kong Scholarship and the 985 Project from the Ministry of Education of China, the Hunan University and the University of Southern California. FutureNet Technologies Corporation, Monrovia, California, USA and its China subsidiary division provided supports under an agreement for joint development, academic promotion and applications.)