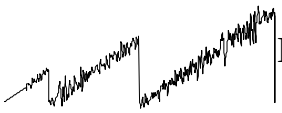
CU-NEES-07-2



NEES at CU Boulder
01000110 01001000 01010100
The George E Brown, Jr. Network for Earthquake Engineering Simulation

Evaluation of OpenFresco and SIMCOR for Fast Hybrid Single Site Simulation

by

Dr. Gary Haussmann

Software Engineer

April 28, 2007

April 2007

Department of Civil Environmental and Architectural Engineering
University of Colorado
UCB 428
Boulder, Colorado 80309-0428

Contents

1 Background and Motivation

The emphasis of the CU NEES facility is on fast hybrid testing, where a typical experiment will link a concurrent computer simulation and physical test specimen into a combined experiment.

Due to the computing and timing demands of fast hybrid testing, the software base at CU NEES has little to no drop-in functionality for distributed hybrid simulation. In distributed hybrid simulation, different parts of the structure are physically located far away from each other, making the "fast" in "fast hybrid testing" difficult or impossible. Many sites perform distributed hybrid simulation without the "fast" portion, opting instead for what is really the well established pseudodynamic tests which function correctly even in the face of large communication delays between sites.

For reasons put forth below, the CU NEES is unable to use these distributed software applications "as is" and meet the requirements of a fast hybrid test. However, the popularity and usefulness of distributed testing suggests that CU NEES should maintain the capability for distributed simulation.

While it might be possible to combine fast hybrid simulation and distributed simulation capabilities into a single software application, the conflicting constraints between distributed and hybrid simulation would produce a convoluted application that may *a priori* be ill-suited for either domain.

CU NEES software is fine-tuned to efficiently perform fast hybrid testing, and has also been used to perform distributed testing. However, there is a strong desire by our site to embrace, in as much as possible, a "community based software" which may become widely used by other sites, streamline our efforts to effectively participate in distributed testing, and which could be integrated with our existing software for fast hybrid testing. Such a single NEES-supported software would require less maintenance from our part, and could be used not only with OpenSEES but also with other software that we may possibly develop.

1.1 Definitions and Concepts

Hybrid Simulation A hybrid simulation combines both numerical computation (a computer model) and a physical specimen. During a hybrid simulation, data from the computer model is used to influence the physical test specimen and vice-versa. Typically the computer is used to model parts of a structure that are impractical to build, while the physical specimen is used to represent components that are difficult to model correctly on a computer.

Distributed Simulation A distributed simulation occurs when the simulation model (for example, a building) is decomposed into multiple parts. Each part is placed on a different computer, and each computer can be physically located a great distance from the other computers. As an example, a building simulation may be split into two halves, with one half of the simulation running on a computer in San Francisco and the other half running on a c omputer in New York. The use of common networking technology (i.e., the Internet) allows the multiple computers to synchronize their simulation state with each other. A distributed simulation may be combined with hybrid simulation.

OpenFresco *OpenFresco* is a software library useful for conducting local and distributed simulations. Using *OpenFresco* it is possible to connect physically separated sites running many types of simulation software. Through the use of "experimental sites" *OpenFresco* enables the user to run a distributed hybrid simulation.

SIMCOR *SIMCOR* (Simulation Coordinator) is a set of software programs and modules that enable distributed and hybrid simulation. Both simulation software and physical specimens can communicate through the Simulation Coordinator in order to exchange data and synchronize simulation events. Like *OpenFresco* , *SIMCOR* is compatible with many versions of simulation software and instrumentation.

Fast Hybrid Simulation In this document the term "Fast Hybrid Simulation" or "Fast Hybrid Test" is used to indicate a Hybrid Simulation (distributed or otherwise) where the test is performed at or near real-time. For example, if an earthquake event is 40 seconds long, than a Fast Hybrid Test would take 40 seconds of wall time to perform in real-time, or 400 seconds at 1/10 scale time. In contrast, Pseudodynamic Hybrid Tests might take hours to simulate the same 40 second earthquake event.

Real-Time System A real-time system is defined here as a system in which the correctness of a system depends not only on the logical results of the computation, but also depends on the time at which the results are produced.[2]

Hard Real-Time The hard real-time criteria is a criteria typically imposed on specific real-time computing environments such as avionic or industrial control systems, as well as instrumentation or measuring devices. A hard real-time deadline requires that data or a computation result "arrives" (i.e., is completed) at or before a specific time; the data is invalid if it arrives after a specific time. [3]

Soft Real-Time A soft real-time criteria considers the validity of data based on the time that the data arrives, in a manner very similar to that of the hard real-time criteria. However, in a soft real-time environment data arriving after the deadline is not considered invalid, but is considered less important.

Determinism A *deterministic* computing environment is designed so that certain events (such as a data result) can be guaranteed to occur within a predicted timeframe. Deterministic environments are commonly used in hard real-time systems. Typical computers and operating systems such as Windows are not deterministic due to cost and flexibility concerns. Also, typical networking technologies such as ethernet are not deterministic by design to reduce costs.

1.2 Fast Hybrid Testing And The Constraints of Real-Time Simulation

The primary goal of the CU-Boulder *NEES* site involves analyzing the effect of earthquakes using Fast Hybrid Testing (FHT). The primary constraint for a Fast Hybrid Test is time; specifically, the computer simulating a structure in an earthquake must produce updated positions as fast as they would occur in real life. Even with the sophistication of today's computers, this time-constraint is a severe constraint on possible hardware and software to be used in a Fast Hybrid Test.

As an example consider a Fast Hybrid Test where the building simulation is performed in time increments of ten milleseconds (10mS). For real-time simulation, the computation for each 10mS must be completed in less than 10mS, resulting in the following restrictions:

1. The size of the simulation model must be limited in size, so that the computer can finish computing each simulation step in 10mS. If the model is too large the computer will take to long, rendering the Fast Hybrid Test invalid or unstable.

2. Both the network and operating system must be deterministic, meaning there is no unpredictibility in computation and data flow. An operating system such as windows can "hiccup" or freeze for a fraction of a second. Such a hiccup is inconvenient for a typical computer user, but would be disasterous for a Fast Hybrid Test. Similarly, data over an Ethernet network will sometimes experience "lag" or delays in data transfer. Again, for a Fast Hybrid Test lag as small as 10mS could be disasterous.

2 Existing Software Tools

2.1 Current CU NEES Fast Hybrid Testing Software

Most Fast Hybrid Tests at CU-Boulder use the algorithm proposed in [5], which is an implicit transient method. In order to properly handle the nonlinear response, a modified Newton iteration method is employed using instrument feedback each iteration. The iteration process is synchronized to an external trigger source which goes off every 0.97mS, corresponding to a tick rate of 1024 ticks/second. Typical Fast Hybrid Tests are performed with a timestep of 0.97mS and 10 iterations per timestep. Since this algorithm communicates with the physical specimen every iteration, new actuator commands and instrument measurements are generated every 0.97mS, which allows the simulation model to incorporate and react to physical events in less than 2ms.

An example where this fast response is useful is the test of a Magnetorheological (MR) Damper, which allows the damping coefficient to be modified dynamically during damper opration. The change in damping is performed via an electric current, and can happen as fast as 4mS.

All networking is done over SCRAMNet, which is a deterministic network that provides guaranteed data delivery in a finite amount of time. The use of SCRAMNet avoids the problem of random delays and data loss inherent in nondeterministic networks such as ethernet.[4]

2.2 An Examination of OpenFresco

OpenFresco is a a hybrid simulation module designed to easily integrate with OpenSEES. In fact, it uses the same scripting language (TCL) and object design. Many of the data structures and architectural concepts are borrowed from OpenSEES, although *OpenFresco* can freely run as a standealone software application. No support is provided for timing outside of waiting done in experimental control objects. For example, you can't force the simulation to sync to 1mS steps, but you can force the simulation to wait on an actuator and force the actuator to update in 1mS steps. This method can cause problems with multiple independent actuators, since waiting on each actuator in turn could produce incorrect delays. For deterministic timing the synchronization needs to be centralized, which unfortunately would negate the distributed simulation capabilities of *OpenFresco* .

OpenFresco has built-in support for networked (i.e., distributed) testing, which is problematic for real-time hybrid tests since the networking portion is typically nondeterministic. There also exists built-in support for "local sites" which bypass the networking portion. *OpenFresco* can also use the SCRAMNet deterministic network, although synchronization can still be an issue.

From the point-of-view of the CU NEES lab, *OpenFresco* is attractive because it builds on OpenSEES which is used (in a modified form) by CU for fast hybrid simulation. Because *OpenFresco* melds into the standard object-oriented architecture of OpenSEES and follows the

same general design philosophy, it can be quickly integrated and modified at CU without any learning curve or additional training.

2.3 An Examination of UI-SIMCOR

The software is a "coordinator" software product built on MATLAB which provides no inherent modeling or testing capability. It provides slots for modules, which communicate through the coordinator (a typical "Bus" design pattern [1]). Modules can provide modeling/simulation, or perhaps connect to physical instruments.

Simulation is performed by simulation modules that connect to an external modeling tool (ZEUS, OpenSEES). Experimental modules can communicate with lab sites to drive/measure physical specimens. Different types of modules can be combined in the coordinator; thus a hybrid simulation can be built by combining simulation modules and experimental modules.

The generic module-based approach is very powerful, since each module can be any type of simulation software or any type of experimental interface. Thus, using SIMCOR it would be possible to divide a problem into separate parts, with each part simulated using different software. The separate parts can communicate with each other via *SIMCOR* to provide a combined simulation/experiment, all with minimal modification and customization. In theory tools such as *OpenFresco* could provide similar functionality, by putting simulation software in the SiteServer objects, but in reality *OpenFresco* was generally intended to have a central modeling/simulation program driving multiple "sites" consisting of experimental elements.

Support is built-in to use standard NEES protocols (i.e., NTCP) and in many cases, flexibility and distributed support is favored over efficiency and portability. As such, it would be very difficult to modify SIMCOR to provide real-time hybrid testing. *SIMCOR* is very network-centric, which is a big challenge for real-time systems since typical commercial networks are non-deterministic. The networking software on many computers and operating system is commonly nondeterministic, even on real-time systems. Thus, in order to run *SIMCOR* in a real-time context most of the network-centric source code would have to be modified or removed.

3 A Comparison of Hybrid Software Implementations

Although *OpenFresco* , *SIMCOR* , and the CU NEES custom software all perform hybrid simulation, they have different capabilities and specialities. A great deal of a program's capability comes from its initial design; that is, the set of problems and types of problems the software was original intended to solve. While software in the early stages of development can be easily redesigned or repurposed, established or legacy software is laden with the layers upon of layers of functions and files that have accumulated over years of programming and use. Steering a software monolith toward a purpose it was not originally designed for is possible, but is typically more trouble than it's worth.

3.1 Software Design Goals

OpenFresco appears to be initially intended as a drop-in addon to OpenSEES, although it currently can interoperate with other finite-element programs. *OpenFresco* is not totally divorced from the OpenSEES codebase–you cannot compile *OpenFresco* without certain files from OpenSEES–but the generic element interfaces and usage show that it would be a relatively simple matter to graft any general structural modeling application onto *OpenFresco* . Conceptually *OpenFresco* also inherits many design concepts from OpenSEES, such as the

object-oriented architecture using Nodes, Elements, Integrator objects, etc. Of concern at CU NEES is that OpenSEES was not designed to run well–or at all–on a real-time deterministic system, and so *OpenFresco* inherits that characteristic.

SIMCOR appears designed from the start to use very generic and flexible protocols and interfaces in an attempt to work with as many programs as possible, and even work with different simulation programs simultaneously! While the flexibility of *SIMCOR* is pregnant with possibilities we must remember that the goal of these hybrid simulation tools is to conduct tests and produce results useful for engineering decisions. So far, we have few indications of cross-application distributed tests which have yielded significant, and cost effective results which advanced the State of the Art or the State of the Practice. Ultimately the suitability of *SIMCOR* for use at CU NEES is based on the overlap between *SIMCOR* 's original design goals and CU NEES. The apparent goals of extreme flexibility and network-centric operation of *SIMCOR* are clearly disjoint from CU NEES goals–more so than the goals of *OpenFresco* .

3.2 Integration Algorithms

The CU NEES Fast Hybrid implementation uses an implicit hybrid method which exchanges data with the physical specimen at intervals of approximately 1mS (in reality, 0.97mS). In contrast, both *OpenFresco* and *SIMCOR* emphasize the alpha operator-splitting method, an explicit method with no iteration. The alpha operator-splitting method works well with distributed hybrid tests since there is no iteration involved for each timestep. However, physical feedback from the instruments is fed back into the simulation at each timestep, which is typically 10mS or higher. In order to handle fast occuring events a 10mS response time is too slow; the timestep would need to be reduced to 1-2mS. However, the delays introduced by the computer network at typically much larger than 1mS, making this choice difficult or impossible for a real-time hybrid test.

1. UI-SIMCOR modeling is done via software such as ZEUS-NL or OpenSEES
2. OpenFresco and UI-SIMCOR timing is performed locally at the lab/control level, with no support for global clocking or synchronization.
3. OpenSEES includes support for parallel computation using remote computers.
4. CU NEES has little experience with using and modifying *SIMCOR*

4 Combined CU/OpenFresco Software

While the current CU NEES Fast Hybrid Test Software works, it has received several updates and maintenance upgrades in order to address current project requirements, newer demands on hybrid testing suggest more modification is required. More complicated hybrid configurations with multiple independent actuators are being required. Larger models are being considered for use in hybrid tests. Finally, the interest in distributed hybrid testing with other sites is increasing. A possible solution is a specific real-time hybrid test application to be used at CU-Boulder.

Based on several observations in the CU NEES Fast Hybrid Test site, it appears that none of the current solutions is satisfactory:

- The current CU Fast Hybrid software has not kept up with evolving demands for larger, more complicated hybrid tests and distributed testing.

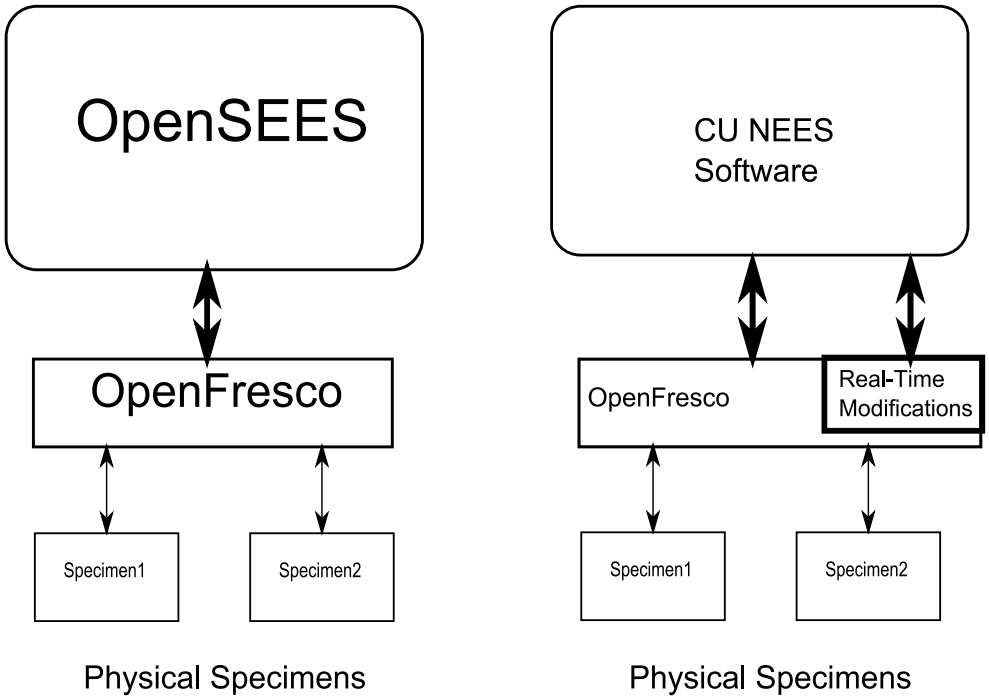


Figure 1: Proposed FHT software using OpenFresco but replacing OpenSEES with CU NEES simulation software

- OpenFresco and SIMCOR are not–and are not intended to be–a turnkey solution for fast hybrid testing.
- OpenFresco, while generally used with OpenSEES, is useful on its own for hybrid simulation.
- OpenFresco does not provide the critical timing and synchronization needed to connect multiple physical specimens running from a common clocking mechanism.

Thus, from a long-term view the best route for CU NEES fast hybrid testing is to combine OpenFresco with simulation software that is targeted at real-time systems and fast hybrid testing. In this setup, the modeling and simulation would be done using customized software designed to run an utilize the special features of a real-time operating system, while the hybrid control interface would be done through a modified version of OpenFresco.

The software package OpenFresco is a separate software module that is intended to easily support hybrid testing both locally and remotely. Although OpenFresco is generally distributed and demonstrated with OpenSEES, the connection between OpenFresco and OpenSEES is fairly weak from a software engineering perspective. Furthermore, OpenFresco is generic enough and small enough that it can be easily modified and attached to other simulation software. Although OpenFresco suffers from the same lack of timing support and remote configuration/control as SIMCOR, its smaller size is more amenable to rapid modification and customization.

Henceforth, ideally the CU-NEES software module should have the following features:

- Parallel implicit transient and be connected to a cluster of computers
- General finite-element models

- Transient analysis-no frequency analysis is planned
- Minimal use of libraries/functions that are not accessible in real-time environments.
- Specialized experimental elements for use with OpenFresco
- Operates with a modified version of OpenFresco to handle timing and synchronization of the fast hybrid test.

As previously mentioned, this software package is mainly intended for viable fast hybrid testing. Therefore the emphasis is less on distributed and pseudodynamic testing and more on reliable performance and hard real-time compliance. The timing impact of various software design choices is important, and any software algorithm that would require a nondeterministic amount of computation time is unusable. The determinism requirement excludes certain iteration methods and material models that have variable iteration counts, and some beam or truss models would certainly have to be excluded as well.

5 Summary of Capabilities

Shown below is a comparison of the current hybrid simulation tools as understood by CU NEES, as well as the capabilities of the proposed CU NEES software built by combining custom CU FHT software with *OpenFresco* .

	OpenSEES	OpenSEES/ OpenFresco	UI-SIMCOR	OpenSEES/ CU	CU FHT/ OpenFresco
Modeling	y	y	y[1]	y	y
Hybrid Simulation	n	y	y	y	y
Hybrid Timing	n	implicit[2]	implicit[2]	y	y
Uses real-time OS	n	n	n	n	y
Distributed	y[3]	y	y	n	y
Distributed Hybrid	n	y	y	n	y

[1] *SIMCOR* provides modules to talk to analysis software and uses MATLAB for computations involving the stiffness matrix.

[2] Both *OpenFresco* and *SIMCOR* rely on the experimental controllers to synchronize timing with their corresponding actuator(s) and instrumentation.

[3] OpenSEES provides parallel processing capability via PETSC.

6 Conclusion

While both *OpenFresco* and *SIMCOR* provide software capabilities for hybrid simulation and testing, neither software package is a suitable solution for performing fast hybrid simulations at the CU NEES facility. While creating a complete fast hybrid simulation application from scratch is possible, the more efficient process would be to take already functional code and augment or customize it where needed.

Addendum and Comments

A Response from Stephen Mahin and Andreas Schellenburg

Our main points:

OpenSees was not formulated for real time operating systems, but it may be possible to do so. Perhaps you have examined the available realtime operating systems to identify the best ones for hybrid simulation. Alternatively, this is something that could be done. Perhaps what you are using is the best, but I got the impression from my conversations with Andreas that you were not overly satisfied with the capabilities of the one you were using.

OpenSees appears to be quite efficient, and there are several clustered implementations I am aware of and I understand that are quite fast. I am not sure what is needed to try to implement openSees on a real time multi-processor environment or in a multi- core, multi-core system. However, In my experience, the speed of the computation is dependent on many additional variables. Many of these relate to the selection of the time step, integration method and iteration procedure, and various parameters used to set maximum number of iterations, error tolerances, etc. I generally get a factor of 100 or more in difference between execution times for different students trying to solve the same problem. Thus, some study of the efficiency of various integration methods, and the necessary geometric and material modeling complexity might be appropriate.

OpenSees now includes a wide variety (perhaps a dozen) of very general integration operators. There is no preference in OpenSees or OpenFresco for any one of these versus another. Your write up suggests that we prefer on type. Personally, I prefer non-iterative Newmark explicit methods if the problem allows it. OpenSees does have some limitations regarding integration operators – It does not have force-based, mixed force/displacement or force/displacement switching integration operators or ones based on energy. These limitations are important for stiff systems and other kinds of special systems.

OpenSees has now more than a dozen implicit and explicit equation solvers installed so that nearly any displacement based integration method can be used in conjunction with a wide variety of iteration techniques.

OpenFresco is not properly referred to as hybrid simulation software, as it cannot do hybrid simulations at all, ever. It is middleware, and provides services needed to carry out hybrid simulations locally or on a local or wide area network. It can be extended to model in a hybrid fashion other aspects of an analysis, not just elements.

OpenFresco is equally adapt at local as well as distributed hybrid simulations. You suggest that it was developed for geographically distributed tests. This is not a true statement of its origin.

OpenFresco is now entirely separate from OpenSees, and the downloadable package includes all of the various files from OpenSees that may be needed. In general, the need for these openSees files is associated with analysis of the specimen, and we hope in the future to generalize this to the need for any complete finite element analysis program (such as OpenSees).

The OpenFresco framework should be separated from its current implementation. For example, OpenFresco's classes can be expanded to consider other aspects of

an analysis that may want to interact with physical or external virtual entities. It is unclear in what ways OpenFresco needs to be deterministic. Clearly, there are aspects of it that can take variable amounts of time (for example, if a nonlinear geometric transformations are used to compute the actuator displacements in complex set ups). In any event, Andreas suggests that this is relatively straight forward if we use a time step equal to the clock speed for the controller.

With regards to Fast Hybrid Simulation have a slightly different set of terminology. In general, I like the term "fast," especially in comparison to "real time" testing which I think is confusing. However, for fast testing it seems that there are two approaches, (1) how to get the test to run at at a specified rate (which is what you focus on), and (2) how to account for real dynamic effects in the test set up when solving the equations of motion. For the first problem, we basically have the same problem as a quasistatic hybrid simulation, but as we have rapid movements we need to insure that the computations are done on schedule, and that the actuators track properly. It is the first of these that you seem to focus on, but I am concerned about the later issue may be in practical problems be equally or more difficult to solve (as inertial and kinematic interaction of the actuator, control system, specimen and lab set up may become more important as the rate of testing increase). If the specimen has significant mass or damping on its own right, there will be real inertial and damping forces in the specimen that need to be accounted for in the numerical portions of the hybrid model. In any event, these all require discussion, and the issues of how to establish determinism of the numerical and operational aspects of the problem is a good starting point.

Anyway, we are hopeful that we can work with you and Victor on these issues.

Stephen Mahin

Comments and corrections by Andreas Schellenburg:

> Andreas: *Section 2.2: True, but if you use one single ExpControl for all the actuators this problem (with independent multiple actuators) can be avoided.*

CU Response: Yes, this is certainly a valid approach to produce a hybrid test where all actuators are synchronized to a single timing signal.

> Andreas: *I think if you can force the Simulation Application (e.g., OpenSEES) to sync to 1mS timesteps (and you do not use nonlinear ExpSetups) the OpenFresco should be able to run deterministic.*

CU Response: This is basically the method used at the CU NEES site at the moment, so we hope that modifying *OpenFresco* to do the same will be a straightforward endevour.

> Not true; we can use whatever integrator the FE software provides. Particularly, if you decide to use OpenSees we have many explicit, several operator splitting and several implicit (including Benson's) integrators available. (refers to the text "... *OpenFresco* and *SIMCOR* emphasize the operator-splitting method")

CU Response: The real issue being brought up in this section is that the implicit method used by CU [[5]] requires extremely fast updates and data turnaround (less than 1mS), whereas the methods commonly used by *OpenFresco* and *SIMCOR* perform updates much less frequently in a typical hybrid test. As an example, the first distributed real-time hybrid test performed by Berkeley (see *http://nees.berkeley.edu/Events/*) involved an update rate of around 20mS, presumably using an operator-splitting method. In addition, these tests are "soft real-time" since the data and computation results are not guaranteed to arrive in the specified time (20mS). The CU NEES site tests are typically done in a "hard real-time" context where data results and computation occur every 1mS (actually 0.97mS) without fail. The enforcement of hard real-time constraints make certain activities much easier, such as stability analysis and programming of the hybrid integration. However, we realize that acheiving hard real-time deadlines in a geographically distributed test is very hard and prohibitively expensive.

Because many hybrid tests at other NEES sites have used such (relatively) long update times, these sites have not frequenty observed the issues seen by CU with its 1mS update intervals. Certain key issues seen at CU are due to the 1mS update rate, and becuase they are not observed at other sites these issues are not considered by other software authors when designing their hybrid simulation libraries.

> Andreas: *We are planning to do this (add the ability to combine multiple modeling tools in OpenFresco)*

B Response from Bill Spencer

Review of "Evaluation of OpenFresco and SIMCOR for Fast Hybrid Single Site Simulation"

This document reviews the current state of UI-SimCor, OpenFresco, and Fast Hybrid Test (FHT) with the goal of evaluating the potential of UI-SimCor and OpenFresco to improve existing CU NEES software for FHT. The document also describes the current state of UI-SimCor, OpenFresco, and Fast Hybrid Test (FHT) very well.

The main contents are about the fast distributed (multi-site not single-site) hybrid testing. If CU wants to conduct only single-site fast hybrid testing, then they do not need to use UI-SimCor or OpenFresco, because the existing CU NEES software is already well-developed to efficiently perform fast hybrid testing without external network communications.

The report states that the current CU NEES software is not ready for distributed testing. Therefore, CU seeks to develop or improve the existing software by employing "community based software such as UI-SimCor and OpenFresco" as much

as possible. However, neither UI-SimCor nor OpenFresco are suitable for performing fast hybrid simulations, because they rely on the experimental controller to synchronize timing with their corresponding actuator(s) and instrumentation, which is a critical point for FHT. Moreover, neither SIMCOR nor OpenFresco in their current forms include all DAQ, data, video, and camera functions, all of which will delay communications and undermine the fast rate aspect.

The main problem mentioned in the document with using UI-SimCor for FHT is regarding communications; typical commercial network (Ethernet) communication (e.g., TCP/IP) is nondeterministic, meaning that there maybe unpredictability in computation and data flow. Most of these problems are also present for OpenFresco. The only difference is that OpenFresco can employ SCRAMNet, which is deterministic and compatible with existing CU NEES software for FHT. Note that UI-SimCor can also communicate with equipment which is implemented through SCRAMNet. Last August, a team from UIUC visited UCB to implement UI-SimCor with their -NEES facility, which uses SCRAMNet and anxPC real-time target to impose data and read the feedback signal. This API for -NEES at UCB receives and sends data through TCP/IP network which is nondeterministic. However, when OpenFresco is used in the distributed hybrid test, the same problem arises.

In summary, CU wishes to upgrade their NEES software for fast distributed hybrid testing by combining existing CU NEES software with OpenFresco rather than UI-SimCor, because of compatibility issues with their existing software and hardware (maybe existing CU NEES software for FHT is based on the OpenSees and the hardware is implemented through SCRAMNet) rather than technical issues.

via B.F. Spencer *(evaluation written by Oh-Sung Kwon and Kyu-Sik Park)*

A few statements in the CU document require modification to reflect accurately UI-SimCor features, as below:

> Page 5 original: *"The separate parts can communicate with each other to provide a combined simulation/experiment, all with minimal modification and customization."*

UIUC Comment: *The separate parts in UI-SimCor do not communicate with each other. Instead, they communicate through UI-SimCor.*

CU Response: Document corrected.

> Page 6 original: *"So far, we have few indications of cross-application distributed tests which have yielded significant and cost effective results which advanced the State of the Art or the State of the Practice."*

UIUC Comment: *We believe that there are few results about cross-application distributed tests, because most of software for distributed hybrid testing is not compatible across applications and hardware. This fact should not lead one to the conclusion that there is no need, or that having such capabilities is of little use. In fact, research has shown that flexibility of the software for hybrid simulation provides much more benefit than having a single platform. (there is a good number of papers already published and under review on applications on soil-structure interaction for bridges and*

> *for multi-resolution assessment of very high-rise buildings; research that could not have been done on one analysis platform. For further details, contact Amr Elnashai at: aelnash@uiuc.edu.*

CU Response: while some cross-application tests may be useful, the bulk of hybrid tests involve one or at most two different analysis tools. Although the flexibility to combine, for example, five different analysis programs in one combined test could be considered useful to some researchers, the functionality supported would be the lowest common denominator supported by all five applications. In the support of such flexibility, functions such as hard real-time support are defenestrated because they are not supported by all platforms and applications.

> Page 8 original: *"SIMCOR provides modules to use simulation software but provides no simulation capability by itself."*
>
> UIUC Comment: *The pseudo dynamic simulation capability is an intrinsic feature of UI-SimCor; the modules represent only restoring forces. OpenFresco, on the other hand, is a set of middleware communication services (similar to NTCP and the forthcoming NHCP) that is intended to facilitate hybrid simulation; it cannot perform hybrid simulation itself.*

CU Response: The footnote is a bit vague and actually refers to model analysis, which in SIMCOR is done by an external application or by MATLAB. The footnote does not refer to the integration process used by SIMCOR. The reference to *OpenFresco* in the aforementioned table is actually referring to "OpenSEES/OpenFresco" which does provide simulation and modeling capabilities.

References

[1] Gregor Hohpe and Bobby Woolf. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions.* Addison-Wesley Professional, 2003.

[2] John Stankovic. Misconceptions about real-time computing. *IEEE Computer*, 10 October 1988.

[3] John Stankovic. *Deadline scheduling for real-time systems: EDF and related algoritms.* Klewer Academic Publishers, 1998.

[4] Eric Stauffer. Assessment of opensees for use in realtime and fast hybrid testing. CU NEES Number CU-NEES-07-1, 1 January 2007.

[5] Zhong Wei. *Fast Hybrid Test System for Substructure Evaluation.* PhD thesis, University of Colorado, Boulder, 2005.