# Implementing Situational Simulations through Distributed Databases

Eddy M. Rojas* and Amlan Mukherjee**

*Assistant Professor, Department of Construction Management. University of Washington. 116 Architecture Hall, Seattle, WA 98195-1610. E-mail: er@u.washington.edu
**Graduate Student, Department of Civil and Environmental Engineering. University of Washington. 116 Architecture Hall, Seattle, WA 98195-1610. E-mail: amlan@u.washington.edu

## Abstract

The Virtual Coach is a web-centric virtual environment that creates temporally dynamic clinical exercises known as situational simulations, with the intention of exposing participants to rapidly unfolding events and the pressure of decision-making. This paper deals with the temporal nature of the databases, which store all the data relevant to situational simulations executed by the Virtual Coach. The first part of the paper describes situational simulations and the relevance of distributed temporal computing. The latter half of the paper focuses on developing the temporal information used to establish temporal consistency of data. It also deals with query processing issues across client-server architectures.

## An Overview of the Virtual Coach

The conceptual framework of the Virtual Coach as explained in Rojas & Mukherjee (2002a) serves as the foundation for the development of situational simulations in the Virtual Coach. The components of this framework consist of three major models: the process model, the product model, and the information model. The process model is a representation of the building process, the product model is a representation of the physical facility, and the information model is a representation of the data environment. The conceptual framework also includes a visualization mechanism to provide process and product feedback to the participant.

The Virtual Coach uses two databases. The *main database* is stored on the server and consists of 'As-Planned' information regarding the construction project. All the information in this database is static. The model for the 'As-Planned' database has been discussed in Rojas & Mukherjee (2002b). The 'As-Built' database remains on the client's machine, and is dynamically updated at the end of each time instant.

## Definition of a Situational Simulation

Situational simulations are temporally dynamic clinical exercises with the objective of exposing participants to rapidly unfolding events and the pressure of quick decision-making. The application of situational simulations provides construction managers and other decision-makers with the opportunity of encountering the wide variety of experiences that the unpredictable nature of the construction industry offers.

The most extensive use of situational simulations is found in the politico-military areas (Allen 1987, Bloomfield & Whaley 1965, Goldhammer & Speier 1959). Relief operations management after natural disasters (Ritchie 1985) also provides areas for application of situational simulations. Situational simulations developed for the construction management domain include AROUSAL (Ndekugri & Lansley 1992). AROUSAL focuses on simulating the management process of a construction company rather than the management process of a construction project. An example of project-centered situational simulation is the game CONSTRUCTO, created by Halpin and Woodhead (1970). This game included the basic components of a situational simulation. However, CONSTRUCTO was based on a process model only and did not include any information or product models, or the visualization of the simulated environment.

Most situational simulations developed so far have been discrete event simulations. Hence, the requirement for temporal databases has not arisen. In addition, since they have not been web-based, they have not had to deal with distributed systems. The Virtual Coach is not a discrete event simulator, but a continuous one, which does not hop from event to event but instead, traverses the whole time length of the project continuously with randomly generated events. This has resulted in the need to deal with issues of computing across distributed databases in quasi-real time.

## Data Flow in the Virtual Coach

Figures 1 and 2 illustrate data flow in the Virtual Coach. Figure 1 explains how a temporal view of the data is queried from the 'As-Planned' database on the server, while Figure 2 follows the flow of data through the Virtual Coach system.

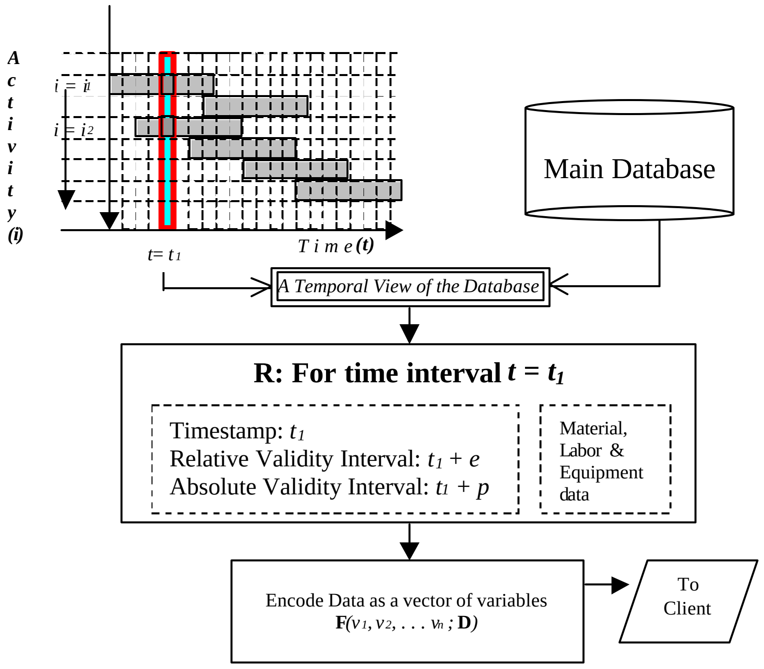
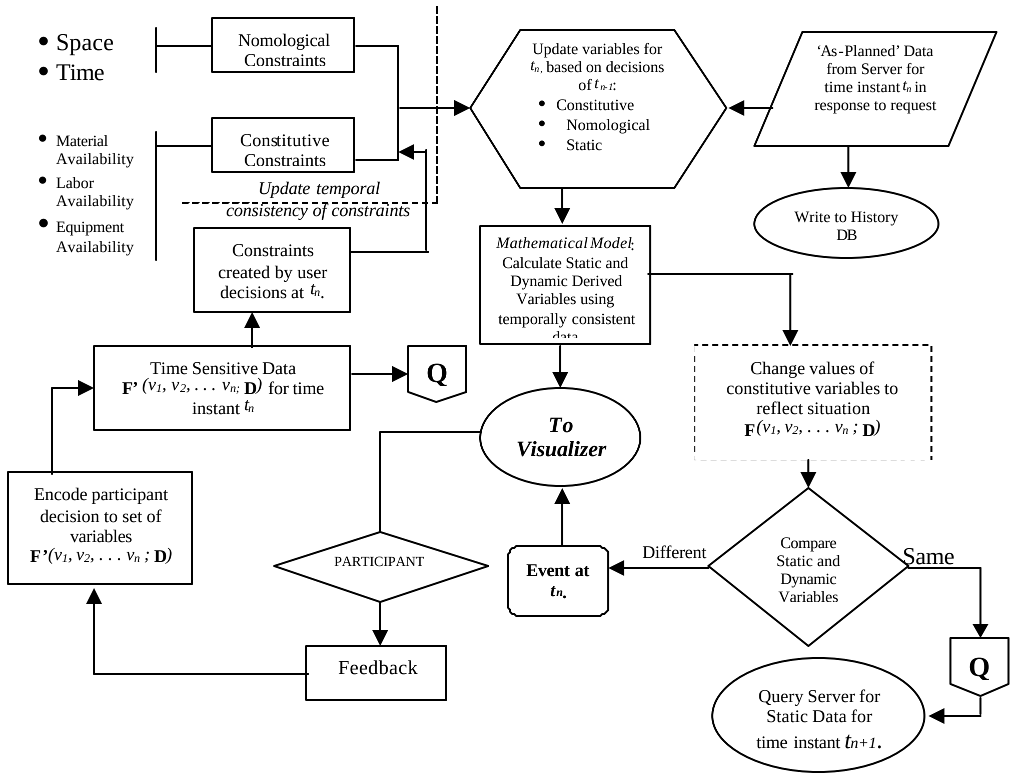

Activity (i)
i = i1
i = i2
t= t1
Time (t)
Main Database
A Temporal View of the Database
R: For time interval t = t1
Timestamp: t1
Relative Validity Interval: t1 + e
Absolute Validity Interval: t1 + p
Material, Labor & Equipment data
Encode Data as a vector of variables F(v1, v2, . . . vn ; D)
To Client

**Figure 1**: **Temporal View of Data Queried from the Server**



• Space
• Time
Nomological Constraints
• Material Availability
• Labor Availability
• Equipment Availability
Constitutive Constraints
Update temporal consistency of constraints
Constraints created by user decisions at $t_n$.
Update variables for $t_n$, based on decisions of $t_{n-1}$:
• Constitutive
• Nomological
• Static
'As-Planned' Data from Server for time instant $t_n$ in response to request
Write to History DB
Mathematical Model: Calculate Static and Dynamic Derived Variables using temporally consistent data
Time Sensitive Data F' $(v_1, v_2, \ldots v_n; \mathbf{D})$ for time instant $t_n$
Q
To Visualizer
Change values of constitutive variables to reflect situation F$(v_1, v_2, \ldots v_n; \mathbf{D})$
Encode participant decision to set of variables F'$(v_1, v_2, \ldots v_n; \mathbf{D})$
PARTICIPANT
Event at $t_n$.
Different
Compare Static and Dynamic Variables
Same
Feedback
Q
Query Server for Static Data for time instant $t_{n+1}$.

**Figure 2**: **Flow of temporal data in the Virtual Coach**

Figure 2 traces the flow of logic, which handles the temporal data item that comes into the system, queried from the client.

The Virtual Coach system is built on a specific number of data variables. These data variables encode the requirements and availability of material, labor and equipment for a particular time-activity element *(i, t).* Dynamic variables reflect the availability of resources in the simulation at a point of time *t*, while static data values of corresponding variables, describe the state of the simulation, had it proceeded 'As-Planned' and reflect resource requirements. A random Event Generator may also change the values of the dynamic variables to randomly create events.

An activity is allowed to occur only when the availability of resources satisfy the requirements for a particular time-activity element *(i, t).* This lays the foundation for logical consistency of the system. When resource availability does not satisfy resource requirement, an event is created, and participant reaction is solicited to reallocate resources, so that logical consistency can be maintained and the simulation can continue with the activity.

Figure 2 illustrates the ideas explained above. $\mathbf{F}(v_1, v_2, v_3, \ldots v_n; \mathbf{D})$ is the data vector that is queried from the server and then updated to reflect temporal constraints from previous events in the system. After the update, the data vector represents dynamic data that is fed into the mathematical equations to get derived data for the visualizer. The information is also compared to the queried static data to check if availability matches requirements. It may also be passed through an Event Generator. In the process, if an event is directly or indirectly generated due to logical inconsistency, participant feedback is solicited and encoded as fractions of the same variables in the data vector **F'**($v_1$, $v_2$, $v_3$, … $v_n$; **D)**. This data vector is processed to create temporal constraints that are applied to update future static queried data. The **D** in the data vector encodes the temporal information tagged to the data. This information helps in maintaining temporal consistency. Its components are discussed in detail in the following section.

## Temporal Consistency of Data in the Virtual Coach

All data in the simulation will be tagged with temporal information, which will define the temporal status of the data item. Since situational simulations will run in real time, the time sensitivity of the information will need to be considered. All information will need to be logically and temporally consistent at all points of time during the simulation.

Ramamritham (1993) states that temporal consistency arises out of the need to preserve temporal validity of data. Data is said to be temporally valid when it reflects the state of the environment that is being controlled by the system. The need to maintain temporal validity gives rise to temporal constraints on the transactions that process time sensitive data. All transactions must be completed satisfying temporal constraints so that deadlines specific to the system can be met. Hence the goal of the system is to complete transactions in accordance with specific time constraints.

Ramamritham further goes on to describe a typical real time system to consist of a controlling system and a controlled system. In this case the controlling system consists of the backend logistics and the human participant, while, the virtual environment, which simulates the construction project, can be viewed as the controlled system. The controlling system interacts with the environment based on data that is queried from the main database and the information, which is provided by the participant's reactions to the environment. Hence, it is very important that the controlling system appropriately perceive the state of the virtual environment. If differences arise between the perception of the environment and its actual state, inconsistencies will arise in the visualization of the information at hand.

Data inconsistencies can be classified as logical and temporal. A logical inconsistency could arise out of a requirement violation. For instance, a construction activity without necessary material and labor requirements might be allowed to occur, when it should have raised a situation. Temporal inconsistencies will occur when information queried from a particular time interval is inappropriately visualized at a wrong interval or if user input constraints are applied to queried data even when they have ceased to be valid.

A temporal inconsistency might in turn give rise to a logical inconsistency. For instance a decision regarding whether a construction activity should occur or not is taken by a comparison of available resources with resource requirements of the activity at that point of time. Now, because of previous events, a lag $\delta$ in time may be created between the simulation time $t$, and the time of the 'As-Planned' database. This would mean that for the time-activity element $(i,t)$ the resource requirements of the time–activity element *(i, t-*

*d)* would need to be queried. A temporal inconsistency would arise if the value of $\delta$ were not appropriately updated to reflect the real delay in the controlled environment. This inconsistency would in turn give rise to a logically inconsistent action by the controlling system. It can be argued that all logical inconsistencies arise from temporal inconsistencies, and that all temporal inconsistencies will eventually give rise to logical inconsistencies.

Each piece of data would be tagged with three pieces of information to reflect its temporal status. To start with each data item would have a *timestamp* that would indicate the time interval at which it was created or to which it belongs. Hence, when resource availability data for the activity $i$ is created during simulation time interval $t$, then the data item is said to have the timestamp $t$. As discussed earlier, there may be a lag $\delta$ between the simulation time and the static time of the database, which is being queried. Hence the time stamp of the data item that would need to be queried would be $(i, t\text{-}d)$.

Data items created at time interval $t$ would also have a validity interval. The validity intervals would give rise to constraints that would need to be followed in order to maintain temporal consistency. For example, if a participant decides to cut labor by half for the next $n$ intervals for the activity $i$ during the time interval $t_1$ then he/she immediately inputs information that has a timestamp $t_1$ and an absolute validity interval $n$. This would mean that the data variables that encode his/her decision of cutting labor by half is valid only till the time interval $t_1+n$ for the activity $i$. This data may be applied to labor availability data for the activity $(i, t)$ only as long as the following temporal constraint is fulfilled:

$$t - t_1 <= n$$

Where $t$ is the current simulation time and $t_1$ is the time stamp of the data.

Finally, there will also be a relative validity interval, which as the name denotes will conserve temporal constraints between different data items. Hence, if an activity $j$ is constrained to start at least $n$ time intervals after the activity $i$ completes, then $n$ would be a relative validity interval between the data items pertinent to the activities $i$ and $j$. A delay in the activity $i$ by $d$ time intervals would mean that the timestamp of the activity $j$ would need to be updated as well. This could be expressed by the following constraint:

$$timestamp_j - timestamp_i >= duration_i + n$$

Where:

$$duration_i = duration_i^P + d$$

at any time $t$:

$$timestamp_i <= t <= timestamp_i + duration_i + n$$

The $duration_i^P$ is the planned duration for the activity $i$ as queried from the ‘As-Planned’ database and $duration_i$ is the dynamically changing value of the duration of the activity $i$ which reflects the delay $d$ in the activity.

The temporal status of each data item is reflected by the tagged temporal information consisting of a timestamp for the data and relevant relative and absolute validity intervals.

## Query Processing Issues

The Virtual Coach utilizes a strict client-server protocol for its database systems. In general the client-server protocol allows one site to send a request to another site, the server, which sends an answer as a response to this request. In a strict client-server system each site has a fixed role of always acting either as a client or a server. In such cases the server is the data source and the client is the query source.

In the Virtual Coach, the client is the participant's workstation and the server contains the 'As-Planned' information regarding the virtual construction project that is being simulated. The client queries the server on demand. The advantages of choosing a client-server system for the purpose could be listed as follows:

- Only data on server machines need to be backed up.

- Security issues can be addressed by controlling only the server machines and client-server communication links.

- Client and server machines can be equipped according to their specific requirements. Hence, while the clients can be simple desktop PCs with good support for GUIs, the servers could be more powerful machines with higher processing capacity and better I/O performance.

The point of concern remains on how to distribute the computing load between the client and the server.

Kossmann (2000) surveys the commonly used approaches in exploiting client resources, and discusses the trade-offs between them. He presents the alternatives as Query Shipping, Data Shipping and Hybrid Shipping. While the query shipping technique conducts query processing on the server and sends the results to the client, the data shipping technique ships all the relevant data tables to the client and conducts the query processing on the client. Hybrid shipping offers the flexibility to execute query operators on both client and server.

The Virtual Coach aims at shifting most of the computing load onto the client and using the server only to host the 'As-Planned' database for the simulated project, which will be queried on demand. Since the database will, under typical situations, contain up to half a million data points, shipping data from the server to the client is not a very good idea. Hence even though all computations dealing with the queried information will be carried out on the client, the query processing will occur on the server. Precedence information regarding the project schedule will be stored in a Directed Acyclic Graph (DAG) (Cormen et al, 2001), which will be dynamically updated during the simulation, as it will contain relative validity intervals between activities. The timestamps for data relevant to activities subject to delay will need to be dynamically updated to maintain temporal consistency. The Hybrid-shipping alternative turns out to be the most useful under the circumstances. Therefore, while in response to requests from clients the server will only ship the relevant results queried from the 'As-Planned' database, the DAG for each project coding the precedence information will be shipped from the server to the client at the beginning of each simulation.

As explained earlier, during the simulation a time lag of δ may arise between the simulation time *t* and the time of the 'As-Planned' database. Hence, all data queried at the server will return results that will need to be updated at the client to maintain temporal consistency. This will follow a compensation based online query processing technique as discussed by Srinivasan and Carey (1992) instead of updating the information on the server. The 'As-Planned' data stored on the server is always static and never changed. However, updates will be propagated on a need basis on the DAG, which codes the activity precedence information on the client. This further justifies the usage of the Hybrid Shipping query-processing alternative.

## Conclusions

The situational simulation runs in accelerated real time. A project, which spans over a year in real life, is typically scaled down in the Virtual Coach environment to a period of a few hours. As the simulation proceeds and the events arise, participants are provided with limited time bounds to decide on effective strategies to deal with the situation. This time bound is not unlimited, but is directly proportional to the time a construction manager would have for similar situations in real life.

The distributed nature of the Virtual Coach will provide a framework for future research to extend the system to cater to multi-participant construction projects. The participants involved could be distributed across different geographical locations and yet collaborate and learn from each other's experiences within a shared virtual environment.

Future work involves the development of a detailed algebra to describe and explain the functioning of the algorithms that maintains the temporal information pertinent to each data item and also dynamically defines and updates the temporal constraints.

## References:

Allen, T. (1987). "*War Games*." McGraw-Hill, New York.

Bloomfield, L. & Whaley, B. (1965). "The Political-Military Exercise: A Progress Report" Orbis. Vol. 8, No. 4, pp. 854-870.

Cormen, T.H., Leiserson, Charles E., Rivest, Ronald L., Stein, C. (2001) "An Introduction to Algorithms" 2/e, McGraw-Hill, New York.

Goldhammer, H. & Speier, H. (1959). "*Some Observations on Political Gaming*." World Politics. Vol. 12, No. 1, pp. 71-83.

Halpin, D. & Woodhead, R. (1970). "*A Computerized Construction Management Game*." Department of Civil Engineering, University of Illinois at Urbana-Champaign. December.

Kossmann, D. (2000) "*The State of the Art in Distributed Query Processing*" ACM Computing Surveys, Vol. 32, No. 4, p. 422-469. December 2000.

Ndekugri, I. & Lansley, P. (1992). "*Role of Simulation in Construction Management*." Building Research and Information, Vol. 20, No. 2, pp. 109-115.

Ramamritham, K. "*Real-Time Databases*", (invited paper) International Journal of Distributed and Parallel Databases 1 (1993), pp. 199-226, 1993.

Ritchie, G. (1985). *“Atlantis: The Basis for Management Simulation Development.”* Simulation/Games for Learning, Vol. 15, No. 1, pp. 28-43.

Rojas, E & Mukherjee, A. (2002a) *“Modeling the Construction Management Process to Support Situational Simulations.”* Journal of Computing in Civil Engineering, ASCE. *(In print)*

Rojas, E & Mukherjee, A. (2002b) *“Data Modeling for the Virtual Coach.”* ASCE Conference on Information Technologies in Civil Engineering, Washington, DC, November 2nd- 3rd, 2002. *(In print)*

Srinivasan, V. & Carey, M. (1992) *“Compensation based on-line Query Processing”* Proceedings of the ACM SIGMOD Conference on Management of Data, San Diego, CA, June, pp. 331-340.