Runtime Reconstruction of Simulation Models for Dynamic Structure Systems

Fa Zhang[1] and Qiaoxia Zhao[2]

[1] School of Management, Air Force Engineering University, Xi'an 710038, China
[2] School of Information &Navigation, Air Force Engineering University, Xi'an 710077, China
Richter2000@163.com, Zhaoqx2000@sohu.com

Abstract. In many simulation applications, the target system may change its structure unpredictably. The simulation model should adjust in time, to follow the evolving of the target system. In this paper we put forward a new definition of system to accommodate the dynamic structural change, and classify the structural change of system, include change in components, relationships among components and interaction between system and its environment. Then we propose a distributed framework of simulation model, which supports runtime reconstruction. In this framework, connector components work together to integrate and manage computing components. These elements and their relationship are discussed using π -calculus. We study the weak bi-simulation condition in component replacing, and discuss the consistency of transition of system states. Finally, we point out main problems in modeling and simulation for dynamic structural systems.

Keywords: Target System, Structural Change, Runtime Reconstruction, Simulation Model.

1 Introduction

In many applications, the structure of the target system may change dramatically. For example, natural disasters and incidents have great negative effects on the human [1]. Traditional decision tools are not very useful in emergency response, because the progress of event is dynamic. A few simulation models of emergency management were developed [2]. However, the model is predetermined in the early stage of M&S, it's very hard to adjust the structure of model at runtime. This paradigm of simulation is not very suitable for emergency management. First, the type of next event is difficult to predict, sometimes a brand-new type of event will break out. Second, the process of each event may be different. Third, the involved departments, rescue teams, and response policy in each event may be different too. So, what we need in emergency management is a flexible and adaptive simulation system, which can evolve along with the progress of emergency event.

[1] This work is partially funded by National Natural Science Foundation of China (NSFC) under grant 70971106.

T. Xiao, L. Zhang, and M. Fei (Eds.): AsiaSim 2012, Part III, CCIS 325, pp. 27–36, 2012.

In this paper we proposed a simulation framework for dynamic structure system, which can be reconstructed at runtime. In section 2, as an example, the dynamic structure evolving of emergency management was analyzed. In section 3, the structural changes of system were formalized. In section 4, the framework of dynamic reconstruction of model system is proposed. In conclusion we summarized our work and point out problems in this new simulation paradigm.

2 Structure Change in Emergency Management

Emergency management is a SoS (System of Systems), which composed of five interconnected systems, as shown in Fig.1. Each system is composed of many elements, and evolves with time. The relationship among those systems will change rapidly, and the structure and parameters of each system will change too.

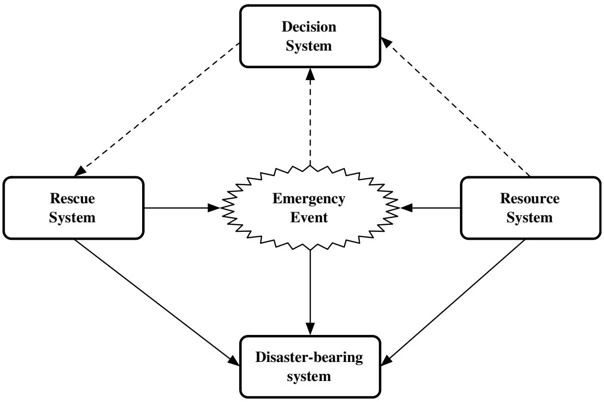

Fig. 1. Emergency management as a System of Systems

As the driven factor of emergency management, an emergency event is time-variant process. Event can be viewed as an evolving network, which starts from a single node of event, after energy released, other nodes may be affected. When some condition is satisfied, other node breaks out. With the break out of nodes, the event spread, derive or transit. Along with the release of energy or with the relief, the state of the node may change. After the energy level of all nodes reduced, the event goes to end. Therefore, event system is a dynamic network, the number of nodes, the type of nodes, and the linkage among nodes are dynamic, and the process of event can't be modeled completely before the end of events. The decision system, rescue system, resource system and disaster-bearing system are also dynamic. What draws special attention is that there are structural changes, not only quantitative change.

A few researchers noticed the change of system, and proposed several methods, such as online simulation [3], DDDAS(dynamic data driven application system) [4], and symbiotic simulation [5]. In those methods, the simulation application is connected with the real system, and the model can be adjusted based on the continuous arrival of the real data. However, those methods can only deal with the changes of parameters or states, the structural change of system is not taken into consideration. Till now, it is a challenge to construct a simulation model, which can change its model structure along with the evolving of system. A new paradigm of modeling and simulation is necessary. In this paradigm, the simulation system is connected tightly with the target system, the quantitative and qualitative changes of the target system are identified, and under the guidance of human, the model is adjusted at runtime.

3 Formal Analysis of Dynamic Structure System

3.1 A New Definition of System

Complex system may be a network of many elements, and there are nonlinear relations among those elements, as a result the overall characteristics emerge [6]. In formal, a system is a 2-tuple:

$$S = < E, R > \quad ,$$

where E is the set of components, $E = \{e_1, e_2, \cdots, e_n\}$, e_i is a component of the system. R is a binary relation over E. In this traditional definition of system, it is implicit assume that the component set and the relation among those elements keep invariant.

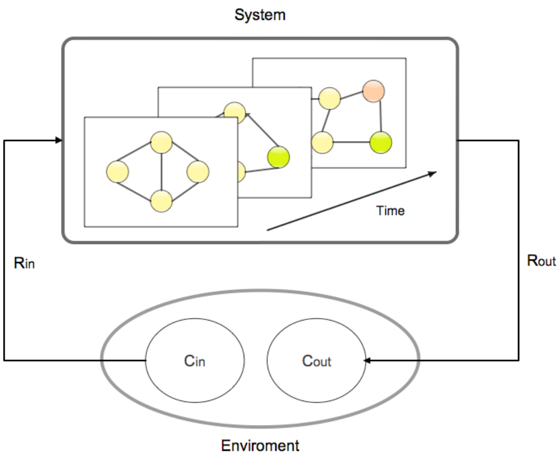

Fig. 2. The concept model of system with dynamic structural change

However, in dynamic situation the above definition of system has limits. Each system has its inner structure, which is determined by the interconnected components, and matter, energy or information may flow across the boundary. During the observing, the components, the relationships among components, and the input/output relation may change. From this viewpoint, a system may be looked as in Fig.2.

To accommodate all those types of change occurred in the life cycle of a system, specially the structural change of system, we should improve the define of system, here we define system as a 6-tuple:

$$S =< E, R, C_{in}, C_{out}, R_{in}, R_{out} > \ ,$$

where E is the set of components, $E = \{e_1, e_2, \cdots, e_n\}$, e_i is a component in the system; R is a binary relation over E ; C_{in} is the set of elements in environment which may send matter, energy or information into the system; C_{out} is the set of elements in environment which may receive matter, energy or information from the system; $R_{in} \subseteq C_{in} \times E$ is the binary relation from C_{in} to E, in other words, input relation between environment and the system; $R_{out} \subseteq E \times C_{out}$ is the binary relation from E to C_{out}, in other words, output from the system to environment.

3.2 Structural Change of System

As illustrated in the example of SoS of emergency management, the structure of the system and the relation between environment and the system may change. In general system theory, only the dynamic of system state have been investigated carefully, the structural change of system has not been studied formally.

The structural change of system only occurs at discrete temporal points; all those temporal points form a set, denoted by $T = \{t \mid t \in R_0^+\}$. A time base $\langle T, < \rangle$ is a structure defined by the set T and an ordering relation < on element of T. $\forall t, t' \in T$, either $t < t'$, $t' < t$ or $t = t'$.

Define temporal relation $\delta = \{< t_m, t_n > \mid t_m < t_n, t_m, t_n \in T\}$ over T, where $< t_m, t_n >$ is a sequential couple. Given a sequential couple $< t_m, t_n >$, for $\forall t \in [t_m, t_n)$, if

$$(E(t) = E(t_m)) \wedge (R(t) = R(t_m)) \wedge (R_{in}(t) = R_{in}(t_m)) \wedge (R_{out}(t) = R_{out}(t_m)) \ ,$$

we said the system is structural stable in segment $[t_m, t_n)$.

Now denote the state of structural stable of the system in $[t_m, t_n)$ as w_m, which is determined by the tuple $< E(t_m), R_m(t_m), R_{in}(t_m), R_{out}(t_m) >$. Specially, denote the structural state at time 0 as w_0.

If at a temporal point $t_s \in T$, the system switches from structural stable state ω to an other structural stable state ω', then a structural transition happens, denoted as $\omega \xrightarrow{t_s} \omega'$, where t_s is the transition instant.

Therefore the dynamic structural process of a system can be described by a state transitional sequence, which starts from time 0 with state ω_0, denoted as follow:

$$\omega_0 \xrightarrow{t_1} \omega_1 \xrightarrow{t_2} \omega_2 \cdots \ .$$

We analyze the types of structural change of system based on the definition above. The structural change of a system may happen in components, interconnections among components and the relation of input or output.

The change of components in the system includes:

- Adding of a new component, or deleting of a component;
- Aggregation/disaggregation. Multiple components aggregate into one component, or one component decompose into multiple;
- Internal change in a component. A component changes its inner structure or behavior, either keeping interface invariant, or changing interfaces too.

The change in the relationships among components may be:

- Introducing a new type relation into the system;
- Linking two components, or breaking a linkage between two components;
- Adjusting property of a link, e.g. the strength, frequency.

For the input/output relation of system and environment, the elements in environment can't be controlled, but the linkage between system and environment may change. The change may be:

- Addition or deletion of a type of input (output);
- Addition or deletion of a linkage;
- Switching the sending (receiving) elements;
- Adjusting the property of a link.

3.3 Dynamic Graph Model

As mentioned above, the structure of system may change in the observation period. It is necessary to develop a formalization to describe and analysis the structural change of system. Graph can be used to describe the structure of system, however it is static. A (static) graph involves four kind of entity: a set of nodes, a set of edges, map nodes to numbers (weight of node), and map edges to numbers (weight of edge). A dynamic graph is obtained when any of these four entities change over time.

Now we combine the set of elements of system (E) and the related elements in environment (*Cin* and *Cout*) into one set $V = E \bigcup C_{in} \bigcup C_{out}$, then denote the number of elements of V is n. The set of edges is the combination of inner relation of system and input/output relation between the system and its environment. Here we only consider the change nodes and edges. Let number n is the max number of nodes in set V in the observation period.

Define a graph space Ω with the number of nodes n. For a digraph $D = (V, E)$, V is the set of nodes and E is the set of arcs. Define a map $\Phi(t, D)$, for $\forall t \in T$ (T is the time base of the system), $\forall D \in \Omega$, which determine a graph $\Phi \in \Omega$. Then dynamic graph is a map from graph space Ω to itself, that is $\Phi : R \times \Omega \rightarrow \Omega$, and satisfy the following conditions:

(1) $\Phi(t_0, D_0) = D_0, \quad \forall t_0 \in T, \quad \forall D_0 \in \Omega, \quad D_0$ is the initial graph,

(2) $\Phi(t_2, \Phi(t_1, D)) = \Phi(t_1 + t_2, D), \forall t_1, t_2 \in T, \forall D \in \Omega.$

4 Reconstruction Framework of Model System

To quickly reflect the structural change of the real system, we proposed a component-based simulation framework, in which the model system can be adjusted on the fly. In the simulation framework, we can maintenance and reconfigure the model system at runtime. Basic operations include adding a new component, replacing an existed component, changing the linkage among components, etc.

4.1 Process of Model Reconstructing

The process of reconstructing of model system is depicted in Fig.3. Here, the simulation application was coupled with the real system. The application acquires data from the real system continuously; the algorithm of pattern recognition or dynamic data mining identifies structural change. If structural change happened, then generate the requirement of reconstruction. Users may also generate requirement of reconstruction, e.g., the user need a more accurate model, or want to add new function. The requirements management module analyzes the auto-generate or user generated requirements, make judgment on whether the requirement can be satisfied. If not, it may subscribe a model from external library. If the requirement can be implemented, reconfiguration command is send to the reconfiguration management module. Reconfiguration management module impels the running system to a

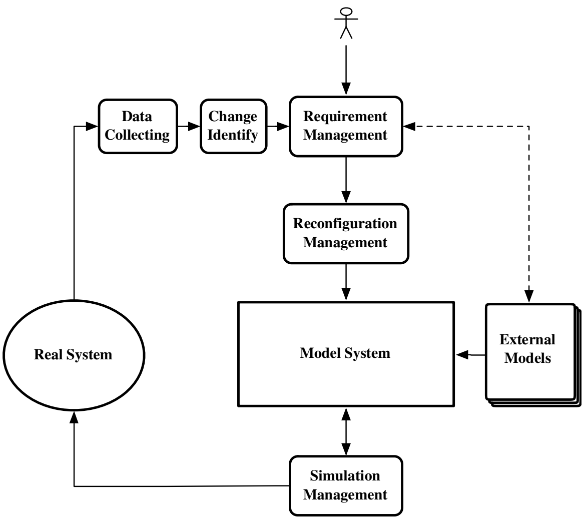

Fig. 3. The process of reconstruction of simulation models

reconfigurable state, and then adjusts the model structure according to reconfiguration algorithm. In the process of reconfiguration, the consistency of the system must be ensured. After reconstruction, the simulation management module resumes the simulation, validates the changed model when needed, and then acquires measures applying to the real system.

4.2 Reconfiguration Management

Reconfiguration management plays the key role in the reconstruction of simulation application system [7]. There are two types of software architecture for dynamic reconfiguration, centralized and distributed [8]. The centralized dynamic reconfiguration management (DRM) is easier to implement, while its efficiency may decline when the system scale increases. In distributed DRM, the DRM function is emerged from the coordination of numbers of components. Distributed architecture is flexible, extensible, and scalable. In the model system, a kind of components, named connector, is introduced. Connectors mediate the addition and deletion of components, and connecting or disconnecting components. In distributed DRM, the model components are partly self-management, such as finding a connector, to register/unregister, etc. Connector is responsible for operation of reconfiguration, while in a large-scale system a composite task may need the coordination of multiple connectors.

As a dynamic coupled system, the reconfiguration of simulation model can be described with process algebras. Each component is a process, the interaction among components is communication of processes, and the message is transmitted in channels [9]. Model components is a computing unit with capability of partly self-management, which is defined as a 4-tuple C=(*Id, P, M, E*), where *Id* is the identifier, *P* is the port, *M* is the internal implement of computing, *E* is the dynamic evolution.

Connector is responsible for the dynamic linkage among model components. We define connector as 7-tuple L=(*Id, P, Ch, MCh, Kc , Kl , E*), where *Id* is the identifier, *P* is port, *Ch* is message channels, *MCh* is management of channels, *Kc* store information of connected model components, Kl store information of other connectors, *E* describe the evolution of dynamic linkage.

Connectors integrate model components. In this architecture, the addition and deletion of model components, the dynamic linkage among model components are performed. A simple example of distributed DRM of simulation model system is shown in Fig.4.

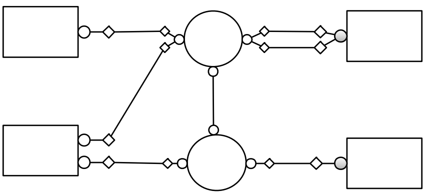

Fig. 4. A simple example of distributed DRM

A basic operation in DRM is addition of a new component. The process can be described with π-calculus. In one side, the new component finds connector, and register to the connector. In other side, the connector acquires the id and service description s of the model component, and then stores it into the library *Kc* .

The registration of the model component is:

$$reginfo(id,s) = (id,s)(\overline{registor} < id,s >)$$

The connector acquires and stores the related information of the model components, as:

$$creginfo(x,y) = (x,y)(registor(x,y))$$

One other basic operation of dynamic reconfiguration is to connect two model components. One component provides service s, and the other wants to use this service. In the process of connecting, at least three components are involved, the two model components and one or more connectors. Those components are all abstracted as a process.

When model component A need a service s, first it request service from connector C, as:

$$requestservice(i,r,l) = \overline{i} < s > .r(z).\overline{z} < l >$$

where A send message about request s on channel i, then wait for response message z from the connector on channel r , and then send the address of the request service component to service component on channel l.

When the connector C received the request, it searches the service information s in the library Kc, (or may ask other connectors for the address of service), and then return the address of the service-providing component to A.

$$queryservice(i,r,p) = i(y).([y = a]\overline{r} < p > +[y \neq s]\overline{ask} < y >)$$

The component S provides service to A, as:

$$provideservice(p,s) = p(x).\overline{x} < s >$$

Where S receives the request address x on channel p, then send service s to A on x.

4.3 Consistency in Reconfiguration

In dynamic reconfiguration, it is very important to ensure the model system consistency, in other words, to avoid illegal evolution. Ensuring consistency is a complicated issue, which can be discussed at different level as syntax, semantic or application. For reconstruction of simulation models, the basic constrain is to ensure the behavioral consistency of computing component. When a component is replaced by another, what need are not only the compatibility of their interface, but also the consistency of their observable behaviors.

For example, replacing one component with a new one. Even though the two components are compatible in interface, the system may be failure because the interactive protocol is not compatible. Therefore, we should also define interactive protocol. The condition of behavioral consistency is:

Suppose component C evolved into C', their behavior are formed as process P and P'. To ensure consistency of behavior, whenever P and P' are weak bi-simulation. Define R is a binary relation over process, process $P_1, P_2, (P_1, P_2) \in R$. R is a weak bi-simulation relation if and only if, for any operation $a \in Act$, satisfy

(1) Whenever $P_1 \xrightarrow{a} P_1^{'}$, then there exists some $P_2^{'}$, such that $P_2 \xrightarrow{\hat{a}} P_2^{'}$ and $(P_1^{'}, P_2^{'}) \in R$;

(2) Whenever $P_2 \xrightarrow{a} P_2^{'}$, then there exists some $P_1^{'}$, such that $P_1 \xrightarrow{\hat{a}} P_1^{'}$ and $(P_1^{'}, P_2^{'}) \in R$;

where $\hat{a}$ is all behaviors in a, except the unobservable behaviors τ in a.

For a running system, it's probability to save and restore the states of some components in the process of reconfiguration. The evolution operations are transactional, one atomic operation must be finished, or if it's interrupted, the state of the system must be rolled back. Ensuring state consistency is necessary when components evolve. First, the involved components must be drive into a special state, in which they are not and will not launch or participate any transaction. Then DRM collect and save all state information, and transfer state information to target components. After reconfiguration, the system resumes from the correct state and continue to run.

5 Conclusions

In many applications, e.g. emergency management, the structure of real system will change, thus it is necessary to evolve the model system along with the evolving of events. It is a general feature of many complex systems in dynamic environments. We analyzed the internal components, the relation among components and the input/output of such systems. A framework of simulation system was proposed to deal with the structural change of target system. We proposed a dynamic reconstruction architecture, which composed of computing components and connectors. Connectors are responsible for operation of reconfiguration. This distributed architecture is flexible, scalable and easy to achieve load balancing. Dynamic reconstruction of model system appeals a new paradigm of simulation, there are many problems to solve, such as design the algorithm of reconfiguration, ensure consistency, dynamic model validation, etc. Further study remains to be continued.

Acknowledgements. This work was funded by National Natural Science Foundation of China (NSFC) under grant 70971106. We are grateful to Professor Xuan Hui-yu for her good advice, and Dr. Mu Chen for his excellent research assistant.

References

1. Wein, L.M.: Homeland security: from mathematical models to policy implementation. Operations Research 57, 801–811 (2009)
2. Tang, A.P., Wen, A.H.: An intelligent simulation system for earthquake disaster assessment. Computers & Geosciences 35, 871–879 (2009)
3. Mitchell, B., Yilmaz, L.: Symbiotic adaptive multisimulation: An autonomic simulation framework for real-time decision support under uncertainty. ACM Transactions on Modeling and Computer Simulation 19, 1–31 (2008)
4. Celik, N., Lee, S., Vasudevan, K., Son, Y.-J.: DDDAS-based multi-fidelity simulation framework for supply chain systems. IIE Trans. 42, 325–341 (2010)
5. Aydt, H., Turner, S., Cai, W., Low, M.: Research issues in symbiotic simulation. In: Winter Simulation Conference, pp. 1213–1222 (2010)
6. Mitchell, M.: Complexity: A Guided Tour. Oxford University Press (2009)
7. Bouassida, I., Guennoun, K., Drira, K., Chassot, C., Jmaiel, M.: A rule-driven approach for architectural self adaptation in collaborative activities using graph grammars. International Journal of Autonomic Computing 1, 226–245 (2010)
8. Taylor, R.N., Medvidovic, N., Oreizy, P.: Runtime Software Adaptation: Framework, Approaches, and Styles. In: IEEE/IFIP Conference on Software Architecture, pp. 455–461 (2009)
9. Sangiorgi, D., Walker, D.: The π-calculus: A theory of mobile processes. Cambridge University Press (2001)