Integrating UML and GPSS for Business Process Modeling and Simulation

Yi Xie
Institute of computer and information engineering
Zhejiang Gongshang University
Hangzhou 310018
China

Abstract: - Simulation is a dynamic modeling technology. However there is little integration of business process static modeling and simulation, which prevents simulation from becoming a mainstream modeling method/tool. Firstly, UML activity diagram is extended using UML extensibility mechanism for BPS, which enhances activity diagram's ability of describing business process and adds some information which the simulation needs. Secondly, the principles and methods that can be used to translate extended UML activity diagram into the GPSS model are proposed, which realizes the integration of the static model and the dynamic model. Finally, a supporting software tool system of integrating UML and GPSS for business process modeling and simulation has been developed based on our proposed methods.

Key words: - Business process modeling; Business process simulation; UML; GPSS

1 Introduction

In order to keep a competitive advantage in fierce and complex competition environment business process re-engineering (BPR) is conducted by many leading enterprises currently[1]. Business process modeling (BPM) is necessary to conduct BPR projects. BPM methods/tools used in BPR projects are commonly classified as two types: static and dynamic. Static BPM methods/tools (e.g. Flowcharter, EasyFlow, IDEF, Process Mapping, UML, Visio and WorkSmart Analysis)[1][2] represent business process by graphical symbols, where individual activities within the process are shown as a series of rectangles and arrows. These methods/tools are easy to use and master. And the built models using these methods/tools are easy to understand and maintain. However most of these methods/tools are not able to conduct 'what if' analysis[1]. Nor are they able to show a dynamic change in business processes and evaluate the effects of stochastic events and random behavior of resources. Business process simulation (BPS) is commonly known as the dynamic BPM methods/tools (e.g. Discrete Event Simulation-based approaches, Petri net-based approaches, Information Systems Architecture-based approaches)[3][4][5]. These methods/tools are able to simulate dynamically entities through business process in the detail physical level of the business process and to conduct 'what if' analysis. So they can evaluate and analyze quantitatively effects of designed business process before implementation. However, there are few effective integrations between static BPM methods/tools and BPS methods/tools. BPS methods/tools usually require a certain amount of expertise to build models and are used by simulation modeling practitioners rather than business analysis specialists[6]. So, BPS methods/tools are prevented from becoming a mainstream BPM tool. Currently

80% of BPR projects used static BPM methods/tools, which lead to the rate of failure in BPR projects is over 50%[1].

So, it has been a popular research field[6][7][8] how to integrate static BPM methods/tools and Dynamic BPM methods/tools. Larry Whitman and Adrien Presley[7] research integration of IDEF0/IDEF3 modeling methods and discrete event simulation and propose the integrated modeling and simulation environment/frame, however, the concrete principles and methods are not discussed and researched. Ning-ke and Niu-dong[8] research business process simulation modeling based on IDEF3. Because of lack of much information which simulation need, the IDEF3 model must be extended in order to generate simulation model from IDEF3 model. However IDEF modeling method itself does not provide the extension mechanism.

UML, which provides the extension mechanisms for specific area of application, has been adopted by OMG and become industrial standard. Discrete event simulation is a dynamic modeling method and has more power in describing business process than other methods (e.g. Petri net)[9]. So, business process simulation modeling techniques based on UML and Discrete event simulation (such as GPSS) are researched in this paper.

2 Extending UML activity diagram for BPS

UML offers the possibility to extend and adapt its meta-model to a specific area of application through the creation of profiles. UML profiles are UML packages with the stereotype «profile». A profile can extend a meta-model or another profile while preserving the syntax and semantic of existing UML elements. It adds elements which extend existing classes. UML profiles consist of stereotypes, constraints and tagged values[10]. A stereotype is a model element defined by its name and by the base class(es) to which it is assigned. Constraints are applied to stereotypes in order to indicate restrictions. Tagged values are additional meta-attributes assigned to a stereotype, specified as name-value pairs.

UML activity diagram is one kind of better process modeling method, but it is a static modeling method in nature and lack of some information which simulation needs. So, UML activity diagram is extended using UML extensibility mechanism for BPS, which enhances activity diagram's ability of describing business process and adds some information that the simulation needs. The further specifications are shown in Tab.1 to Tab.3. The «s_Action» is atomic in the extended UML diagrams. That is say; the «s_Action» uses zero or one resource. Moreover the resource needed by the «s_Action» is seized as soon as the «s_Action» is executed, and the resource is released until the activity is finished. If the conditions are not satisfied, the activity is not a «s_Action», and can be further decomposed.

Tab.1 The further specification of «s_Action»

Name	s_Action
Base Class	Action
Tagged Values	ExeTime: UML::Datatypes::String NeedRes:《stereotype》s_Resource NeedQnt: UML::Datatypes::Integer
Constraints	Before and after a s_Action always has a ControlFlow. context Function inv: self.ControlFlow[predecessor]->size()=1 and self.ControFlow[successor]->size()=1

Tab.2 The further specification of « s_Start »

Name	s_Start
Base Class	InitialNode
Tagged Values	ArvlTimeInterval: UML::Datatypes::String FristArvlTime: UML::Datatypes::Integer ArvlTotalNumber: UML::Datatypes::Integer Priority: UML::Datatypes::Integer
Constraints	Before a s_Start has no ControlFlow.and after a s_Start always has a ConrolFlow context Function inv: self.ControlFlow[predecessor]->size()=0 and self.ControFlow[successor]->size()=1

Tab.3 The further specification of « s_Resource »

Name	s_Resource
Base Class	ActivityPartition
Tagged Values	AvailableQnt: UML::Datatypes::Integer
Constraints	A s_Resource is assigned to one or more s_Action. context s_Resource inv: self.s_Action->size()>=1

3 Generating GPSS model from UML Extended Activity diagram

General purpose system simulation (GPSS) is a process-oriented discrete event simulation language and has been widely used in computer simulation area. The executable GPSS model can be generated from the extended UML activity diagram. This includes the following three steps:

3.1 Generating the definition sentences of GPSS

The definitions of storage, queue, function and condition variable in the GPSS model are fulfilled through traversing the whole extended UML activity diagram. The name and capacity of the storage are defined according the serial number and quantity of resource in the extended UML activity diagram. A queue is defined corresponding to every «s_Action». The named rule of queue is *L, here * is the serial number of the «s_Action». Each guard condition is corresponding to one definition of condition variable. The named rule of condition variable is *GC, here * is the serial number of the control flow.

3.2 Generating the simulation sentences of GPSS

This is the key step of the whole model translation. First, the extended UML activity diagram is divided to some kinds of basic structural modules, such as sequence module, merge/decision module, fork module, and join module, shown in Fig.1 to Fig.4. Then each module is translated to the corresponding simulation sentences of GPSS model.

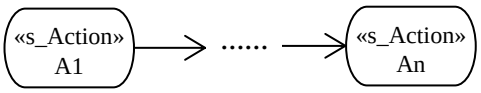

Fig.1 Sequence module

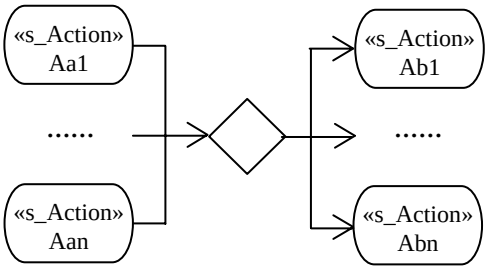

Fig.2 Merge/decision module

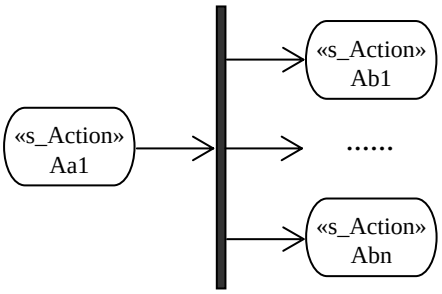

Fig.3 Fork module

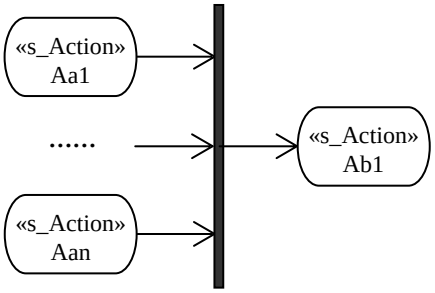

Fig.4 Join module

The translation of single «s_Action» module to GPSS simulation sentence is discussed first, because the «s_Action» is most basic unit of extended UML activity diagram. Suppose the serial number of the «s_Action» is Ai and Ai use resource Sj. The executing time follows the random distribution. The GPSS sentence list is written as PrgmAct (Ai) which is translated from the signal «s_Action» Ai. So the PrgmAct(Ai) is:

```
QUEU  AiL      /* Queuing at AiL */
ENTE  Sj       /* seizing resource */
DEPA  AiL      /* Leaving queue AiL */
ADVA  f        /* «s_Action» delaying */
LEAV  Sj       /* Releasing resource */
```

Where: f is an expression of time according to the different interval distribution. For example, if f is exponential distribution, then f= Exponetial(a,b,c), a is the random number generator entity number; b is the shift value used to position the distribution, c is the compression value used to expand or contract the distribution.

If the «s_Action» does not need resource, the PrgmAct(Ai) can be simplified as follows:

```
ADVA  f
```

(1) Translating of the sequence module

Suppose the serial number of control flow and its successor «s_Action» in the sequence module is CFi(i=1,2,…,n) and Ai(i=1,2,…,n) in order respectively, then the GPSS sentence list is written as PrgmSeq(CF1,A1) which is translated from the sequence module starting at the CF1 and A1. So the PrgmSeq(CF1,A1) is:

```
      TEST_E V$CF1GC,1
      PrgmAct(A1)
      …
      TEST_E V$CFnGC,1
      PrgmAct(An)
```

(2) Translating of Merge/Decision module

Suppose the serial number of Merge/Decision, its predecessor control flow, successor control flow and successor «s_Action» in the Merge/Decision module is MD1, CFai, CFbi and Abi(i=1,2,…,n) respectively, then the GPSS sentence list is written as PrgmMD(MD1) which is translated from the Merge/Decision module. So the PrgmMD(MD1) is:

```
      TEST_E  V$CFa1GC,1
      TRAN    ,Ab1
      …
      TEST_E  V$CFanGC,1
      TRAN    ,Ab1
Ab1   TEST_E  V$CFb1GC,1,Ab2
      PrgmSeq(CFb1,Ab1)
Ab2   TEST_E  V$CFb2GC,1,Ab3
      PrgmSeq(CFb2,Ab2)
      …
Abn   TEST_E  V$CFbnGC,1
      PrgmSeq(CFbn,Abn)
```

(3) Translating of the fork module

Suppose the serial number of fork synchronous bar, its predecessor control flow, successor control flow and successor «s_Action» in the fork module is F1, CFa, CFbi and Abi (i=1,2,…,n) respectively, then the GPSS sentence list is written as PrgmFrk (F1) which is translated from the fork module. So the PrgmFrk (F1) is:

```
      TEST_E  V$CFaGC,1
      ASSI    1,0
      SPLI    n-1,F1,1
F1    TEST_E  P$1,1,Ab2
      PrgmSeq(CFb1,Ab1)
Ab2   TEST_E  P$1,2,Ab3
      PrgmSeq(CFb2,Ab2)
        …
Abn   TEST_E  P$1,n
      PrgmSeq(CFbn,Abn)
```

(4) Translating of Join module

Suppose the serial number of join synchronous bar, its predecessor control flow, successor control flow and successor «s_Action» in the join module is Ji, CFai (i=1…n), CFb and Ab respectively, then the GPSS sentence list is written as PrgmJnt(Ji) which is translated from the join module. So the PrgmJnt(Ji) is:

```
      TEST_E V$CFa1GC,1
      TRAN   ,Ji
      …
      TEST_E V$CFanGC,1
      TRAN   ,Ji
Ji    ASSE   n
      PrgmSeq(CFb,Ab)
```

(5) Generating the arriving and leaving module of dynamical entity

Besides the translation of the basic module, the initial node («s_Start») and final node must be also translated. So the GPSS sentence list of the arriving and leaving of dynamic entity can be generated.

Each «s_Start» corresponds to arriving of one kind of dynamic entity. The translated GPSS sentence list is as follows:

```
GENE a, b, c, d, e
```

Where, a, b, c, d, e is the mean(or function), variance (or amend value of function), the first time arriving, total entity number and PRI respectively when the dynamic entity is arrived. These values may be acquired from the corresponding tagged value of «s_Start». Each final node corresponds to leaving of one kind of dynamic entity. The translated GPSS sentence list is as follows:

```
TERM
```

3.3 Generating the control sentences of GPSS

After the translation of step 1 and step 2, there are short of the GPSS control sentences in the model. So, the sentence "SIMU" must be added at the front of translated GPSS sentence list. At the same time, the sentence for controlling the warm-up and end of the simulation must be add also, which is written as PrgmEnd(PT, RT). Here, PT and RT denote the needed warm-up time and the actual runtime respectively. So the PrgmEnd(PT,RT) is:

```
GENE  PT,,,1
TERM  1
GENE  PT+RT
TERM  1
STAR  1,NP
RESE
STAR  1
END
```

4 The supporting software tool system architecture

The supporting software tool system of integrating UML and GPSS for business process modeling and simulation is aimed to help enterprise managers or business analysis specialists easily generate the business process models based on extended UML activity diagram through adding some necessary specific simulation information to the existent business process models based on UML activity diagram, then according to the business process models based on extended UML activity diagram automatically generate and execute the GPSS-based business process simulation model, finally analyze and evaluate the results of simulation and output the results through GUI. The supporting software tool system architecture is shown as Fig.5.

(1) Business process modeling module: providing users with a GUI modeling platform. The users create a business process model based on extended UML activity diagram.

(2) Setting experimental frame module: adding some necessary specific simulation information (such as run time/end conditions, run modes) through GUI.

(3) Model translating module: translating a business process model based on extended UML activity diagram into an executable business process simulation model based on GPSS.

(4) Executing model module: executing business process simulation model based on GPSS.

(5) Analyzing and evaluating module: analyzing and evaluating the results of simulation

(6) Result output module: outputting the simulation data and the result of analysis and evaluation using a variety of forms (such as text and chart) to the users.

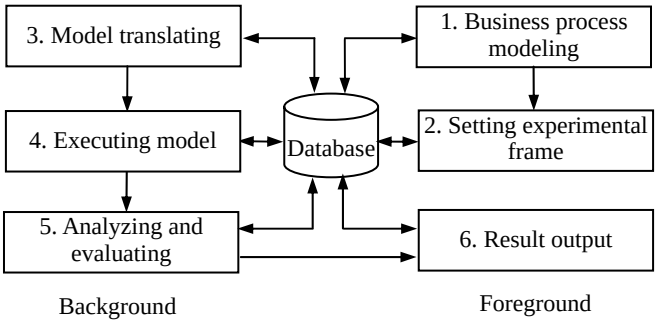

Fig.5 The supporting software tool system architecture

According to hereinbefore discusses and researches, a supporting software tool system of integrating UML and GPSS for business process modeling and simulation has been developed using object-oriented technology and been applied in some

real BPR projects, which show the feasibility and efficacy of the approaches proposed.

5 Conclusions

Because the mapping model and simulation model is integrated in the method of business process modeling and simulation based on UML activity diagram and GPSS, a mass of enterprise existence static mapping models such as UML activity diagram can be used to develop dynamical simulation models easily, which improves the utility of static mapping models, reduces the workload on the developers of business process simulation models and modeling cost, and ensures consistency and integrity between static business process models and dynamic business process models. This method is easy to be mastered and used. The built models using this method not only facilitates users understanding and communicating but also can be simulated and support business process dynamical mapping, quantitative analysis and evaluation, which provides decision support for enterprise managers, business analysis specialists or BPR project manager and enhances the probability of success in BPR project.

References:

[1] Paul, R.J., Hlupic, V., Giaglis, G.M. Simulation modelling of business processes. In the Proceedings of the UK Academy for Information Systems 1998 Conference. Lincoln: 1998, pp.311-320

[2] Wendy Currie, Vlatka Hlupic. Simulation modelling: the link between change managenment panaceas. Proceedings of 2000 Winter Simulation Conference. Orlando: 2000, pp.2022-2028

[3] Xie Yi, Tang Renzhong. Research on Event-driven Process Simulation Technology Based on OMT. CHINA MECHANICAL ENGINEERING, 2004, 15(12): pp.1069-1072

[4] Kerim Tumay. Business process simulation[A]. Proceedings of 1996 Winter Simulation Conference. California: 1996, pp.93-98

[5] GONG Shi-hao, YANG Ji-jiang, CHAI Yue-ting, LI Mei-ying, LI Fang-yun. BUSINESS PROCESS MODELING BASED ON COLORED PETRI NETS. Infornation and Control, 2000, 29(01): pp.1-5

[6] Charles R. Harrell, Kevin C. Field. Integrating process mapping and simulation. Proceedings of 1996 Winter Simulation Conference. California: 1996, pp.1292-1296

[7] Larry Whitman, Adrien Presley, Brian Huff. Structured models and dynamic systems analysis: the integration of the idef0/idef3 modeling methods and discrete event simulation. Proceedings of 1997 Winter Simulation Conference. Atlanta: 1997, pp.518-524

[8] NING Ke, NIU Dong, LI Qing, SHEN Hui, CHEN Yu-liu. IDEF3-based Business Process Simulation Modeling. Computer Integrated Manufacture System, 2003, 9(5): pp.351-356

[9] S.R. Nidumolu, N.M. Menon, B.P. Zeigler. Object-Oriented Business Process Modeling and Simulation: A discrete event system specification framework. Simulation Practice and Theory, 6 (1998): pp.533-571

[10] Object Management Group, Inc.: UML 2.0 Superstructure http://www.omg.org