A GENERAL HETEROGENEOUS SIMULATION KERNEL

Martin HRUBÝ, Doctoral Degree Programme (3)
Dept. of Intelligent Systems, FIT, BUT
E-mail: hrubym@fit.vutbr.cz

Supervised by: Dr. Zdena Rábová

ABSTRACT

This paper describes a method of creating a very general heterogeneous simulation machine. Heterogeneous simulation systems are combinations of various modelling formalisms, tools and languages. Described methodology was used in design of a new simulation language called HELEF (**He**terogeneous **L**anguage **E**nvironment)

1 INTRODUCTION

The heterogeneous modelling (or *multi-modelling*) is one of the possible ways how to model very complicated and large systems. We have various mathematical modelling formalizms (Petri Nets of all types, DEVS by Fishwick & Zeigler, differential equations), simulation languages (discrete, contiunous), simulation tools (Matlab) and programming languages (special languages for knowledge modelling).

It is obvious to interconnect them to get a powerfull complex modelling environment which we may call *Heterogeneous modelling and simulation environment*. Makeing such an environment we have to face some problems contextual to their interconnection. This paper offers one of the solutions.

2 HETEROGENEOUS ENVIRONMENT

We are going to interconnect a set of sub-models (or sub-systems). Let us assume that every sub-system operates at its own space (local names) and it needs to access space of the connected sub-systems (global names).

Majority of the computing systems consists of two parts:

Data (memory): every item may be a function $Data_i = function(Local, Global)$

Activities: sequences of statements (procedures). We will call them the *processes*.

In the following theory, data and processes will be called the *simulation objects* (or *objects*).

3 THE AUTOMATIC INFORMATION NET (AIN)

The method of working interconnection between particular sub-models is based on a computing net called AIN. AIN connects all simulation objects of all sub-models together.

The AIN net is an oriented graph:

Definition 1: AIN Net S is a structure $S = [OS \cup \{AINnoevent\}, R]$, where

1. OS is a set of *simulation objects*, $AINnoevent$ is a special object
2. R is a dependence relation between the simulation objects $R \subseteq OS \times (OS \setminus \{AINnoevent\})$.

Let us use a notation similar to the Petri Nets. If $S = [OS, R]$ is a net, then:

- $\bullet o = \{x \in OS | (x, o) \in R\}$
- $o\bullet = \{y \in OS | (o, y) \in R\}$
- if $o_1, o_2 \in OS$ and $(o_1, o_2) \in R$, which means that $o_2 \in o_1\bullet$, then we write: $o_1 \rightarrow o_2$

3.1 THE AIN EVOLUTION

Definition 2: Structure $AC = (OS, R, Marked, Changed)$ is called a simulation context (an evolutionary state) of the AIN net and:

1. $[OS, R]$ is an AIN net
2. $Marked \subseteq OS$, $Changed \subseteq OS$

Definition 3: An AIN context $AC = (OS, R, Marked, Changed)$ is called to be *valid*, when one of the following conditions is true:

1. $Marked = \emptyset \wedge Changed \neq \emptyset$
2. $Marked \neq \emptyset \wedge Changed = \emptyset$
3. $Marked = Changed = \emptyset$

Definition 4: The valid context $AC = (OS, R, Marked, Changed)$ of AIN is being called:

stabilized if $Marked = Changed = \emptyset$ is true

marked if $Marked \neq \emptyset \wedge Changed = \emptyset$ is true

changed if $Marked = \emptyset \wedge Changed \neq \emptyset$ is true

Definition 5: A valid context $AC_1 = (OS, R, M_1, CH_1)$ derives a valid context $AC_2 = (OS, R, M_2, CH_2)$, if:

1. AC_1 is stabilised, then $M_2 = AINnoevent\bullet$ and $CH_2 = \emptyset$ (will be called AC_U)
2. AC_1 is changed, then $M_2 = \bigcup_{i \in CH_1} i\bullet$ and $CH_2 = \emptyset$
3. AC_1 is marked, then $CH_1 = \{j \in OS | reeval(j) = true\}$

and we write $AC_1 \vdash AC_2$. The $\vdash$ operation defines one evolutionary step of a net. We also may write, that AC_1 and AC_2 contexts are in the evolution relation. The *reeval* operation will be defined in the followings.

THE *REEVAL* FUNCTION

Every simulation object contains a method *reeval* which computes its value and communicates with the simulation engine.

Reeval computes a new value v_2 of simulation object o having value v_1. Then it compares v_1 and v_2. Returns:

true — if $v_1 \neq v_2$. And $v_2 := v_1$

false — if $v_1 = v_2$

3.2 SIMULATION RUN

The AIN evolution specifies a way of simulation process in the heterogeneous models.

Definition 6: An evolution of $S = [OS, R]$ is a context sequence $AC_0, AC_1, AC_2, ...$, where:

1. $AC_0 = (OS, R, M_0, \emptyset)$ is a starting marked context
2. $AC_i \vdash AC_{i+1}$ for all $i = 0, 1, 2, ..., i \in N$

The evolution does not have an default defined terminating context. That one must be specified explicitly in a particular model.

Definition 7: Starting context is a $AC_0 = (OS, R, M_0, \emptyset)$ context, so that:

$$M_0 = \{o \in OS | \bullet o = \emptyset\} \setminus \{AINnoevent\}$$

4 PROCESSES-ORIENTED ENGINE DEFINITION

Processes behave in meaning of these basic laws:

- processes have a direct connection to the AIN evolution (they follow the evolution)
- processes are allowed to modify a contents of AIN objects (nodes), and also to add some new objects and to remove (disconnect) the others
- the process running goes together with the AIN evolution, process can be created, suspended and resumed

Definition 8: A process is a simulation object with these attributes:

$$P = (StartC, ContinueC, Body = [cmd_1, cmd_2, ..., cmd_P])$$

where

StartC — is a reserved simulation object which defines a condition specifying a moment (AIN context) when the processes should be started

ContinueC — is a condition (similar to a *StartC*) allowing the process to *run*.

Body — is a process body (list of commands)

4.1 CONTEXT OF PROCESSES

Definition 9: The global context of process management is a triple $GP = (ActP, ReadyP, WaitingP)$.

$ActP$ is an actual running process

$ReadyP$ is a queue of running processes (having their $ContinueC = true$)

$WaitingP$ is a queue of waiting (suspended) processes (with $ContinueC = false$)

Simulation state with no processes is $GP_\varepsilon = (\varepsilon, [], [])$.

Notation: we may write $p \in GP$, if $p = ActP$ or $p \in ReadyP$ or $p \in WaitingP$

Definition 10: The complete simulation context is

$$SIM = (AC = (OS, R, Marked, Changed), GP = (ActP, ReadyP, WaitingP))$$

Let us assume that some AIN does contain an object p of type "process". And p specifies the $StartC_p$ condition. The whole system is in $SIM = (AC_x, GP_\varepsilon)$ and $p, StartC_p \in AIN$ a $StartC_P \in Changed$. HELEF system compiles it so that $StartC_P \rightarrow p$ and so, p will get into *Marked* in the next evolution step. The standard reevaluation of *process* simulation object is done by algorithm:

1. if $p \in GP \Rightarrow false$
2. if $p \notin GP \wedge value(StartC) = true \Rightarrow true$ and the whole system turns to:

$SIM_2 = (AIN, (ActP, [ReadyP, p], WaitingP))$

5 CONCLUSION

Described method privides a formal backround to a new heterougeneous simulation language called HELEF (Heterogeneous Language Environment). HELEF evolves this basic heterogeneity into a next layer - it allows user to specify a structure and a behaviour of the resulting model. More can be found in [2, 3, 5], this paper was very limited in its size, so the AIN theory was presented in its reduced form.

The HELEF simulation system is now being implemented in the Smalltalk language.

ACKNOWLEDGEMENT

This work has been done within the project CEZ:J22/98: 262200012 *Research in Information and Control Systems* and also supported by the Grant Agency of Czech Republic grants No. 102/01/1485 *Environment for Development, Modelling, and Application of Heterogeneous Systems*.

REFERENCES

[1] Beta Book: `ftp://ftp.mjolner.com/BETA/BOOK/betabook.pdf`

[2] Hrubý, M., Rábová, Z.: HELEF: A New Simulation Language, In: Proceedings of the 36th Spring International Conference MOSIS 2002, Rožnov pod Radhoštěm, CZ, MARQ, 2002, p. 39-47, ISBN 80-85988-71-2

[3] Hrubý Martin, Rábová Zdena: Object Oriented Programming Languages in Modelling, In: Proceedings of The 27th ASU Conference, Rattwik, SE, ASU, 2001, p. 55-65, ISSN 1102-593X

[4] Hrubý Martin, Kočí Radek, Peringer Petr, Rábová Zdena: Tools for Creating of Multimodels, In: Kybernetes: An International Journal of Systems & Cybernetics, roc. 2002, c. 9, GB, p. 1391-1400, ISSN 0368-492X

[5] Hrubý Martin, Rábová Zdenka: Modelling of Real-world Objects using the HELEF Language, In: ASU Newsletter: a publication of the Association of SIMULA Users., roc. 28, c. 1, Stockholm, SE, p. 47-57, ISSN 1102-593X

[6] Self Hp: `http://www.cs.ucsb.edu/labs/oocsb/self/`

[7] Smalltalk Hp: `http://www.smalltalk.org/`