# THE SPECIFICATION OF A MULTICOMPUTER WITH DYNAMIC NETWORK USING LOTOS

J.A. Gallud and J.M. García
Department of Informática
University of Castilla-La Mancha
Campus Universitario s/n
02071-ALBACETE-SPAIN
e-mail jagallud@eui.ucma.es
TEL.: 34-27-22 86 50
FAX : 34-67-50 66 50

**ABSTRACT**

Formal Methods of specification can play an important role in exploring the behaviour of complex systems, as distributed systems or communication protocols. Lotos, based on algebra of processes, has been chosen as an international standard for specifying many systems. We show in this paper the model for a popular class of parallel machines: multicomputers. This paper establishes a guide to employ the Lotos technique in specifying problems on multicomputer architecture. We focus our attention in a new feature of these machines: the dynamic reconfiguration of the network.

**Keywords**: Software engineering, formal specifications and design, computer architecture.

## 1. Introduction

An important problem in software engineering is the high cost related with developing large software systems. Several aspects affect in every development phase, from informal requirements to executable code. Recently, software engineering has seen a powerful trend towards formal methods, that is, using symbolism with precise mathematical meaning and rules and operation. Such aspects allow designers to reduce the cost of the system development and its maintenance.

Formal Description Techniques (or simply FDT) let us to give a precise relation between initial and final states of any application, without describing implementation details. To specify using formal methods has been considered essential to the standardization and customization process, specially for communication and telecommunication protocols.

In this paper, we present an introduction on specifying a model of multicomputer network. We present this model by using a formal technique called Lotos. Lotos [Bol89] is a formal specification language based on algebra of processes. This allows to designers -through a technical analysis- the verification, validation and development of a system.

The basic idea is that Lotos specifications are described by temporal relations between interactions, that express the observable behaviour of the system. A Lotos specification has two parts: processes algebra and abstract data types.

The important criterion in the definition of Lotos processes is to define the observable behaviour of a process without mentioning how the process works internally. In a process definition its behaviour is specified by means of the observable actions sequence seen through of several interaction points or gates.

The dynamic reconfiguration of the interconnection network is an advanced feature of some multicomputers to reduce the communication overhead. Up to now, we have studied the principles of reconfiguration as well as its limitations and trade offs [Gar93]. In this paper, our work focuses on modelling the dynamic network using a formal language. In this way, we pretend to study the dynamic network in depth and to improve our reconfiguration algorithm.

## 2. Multicomputer network specification

Multicomputers rely on an interconnection network between nodes (processors) to support the message-passing mechanism. The network plays a major role in determining the overall performance of a multicomputer. The goal of the dynamic reconfiguration is to increase the multicomputer performance by minimizing the traffic of messages in the network. Therefore, when the traffic between a pair of nodes is intense, the dynamic reconfiguration algorithm will try put the source node close to the destination node.

We are going to use a step-wise refinement method. A Lotos description of this system could be the next specification:

```
specification multicomputer
behaviour
  ( ( processors [e1, ...,en] || dynamic_network [e1, ...,en] )
       <] change [ctr,pctr] ) >> multicomputer
endspec
```

In this system we can observe three different Lotos processes called *processors*, *dynamic network* and *change* respectively. The first one is parallel composed with the dynamic network process. The system can be interrupted by the singular *change* process. This last process is fired internally on the network by a processor to modify the topology. So that, we have above a model of a multicomputer with dynamic reconfiguration.

The next step is to obtain a specification that describes more details about the system. In first place, we could describe the specification of the network with dynamic reconfiguration.

We are going to suppose that there are m processors each one with l links. Because of this, we have to consider n events in the network specification, where n is the result of the following expression: $n = l * m$. We can see the specification as follows:

```
process  dynamic_network [e1, ...,en,ctr](x[]:mess,con[]:intstr) :=

ctr?con[]:intstr;exit
[](  ( e1? x[1]:mess ; ( [con[1]=1]→(e1!x[1])
                   [con[1]=2]→(e2!x[1])
                        .
                        .
                   [con[1]=n]→(en!x[1]) );exit )
  ||| ( e2? x[2]:mess ; ([con[2]=1]→(e1!x[2])
                   [con[2]=2]→(e2!x[2])
                        .
                        .
                   [con[2]=n]→(en!x[2]) );exit )
       .
       .
```

```
  ||| ( en? x[n]:mess ; ([con[n]=1]→(e1!x[n])
                         [con[n]=2]→(e2!x[n])
                               .
                               .
                         [con[n]=n]→(en!x[n]) );exit ) )
endproc
```

There are two general observable actions: It can appear the internal event associated to a request for changing the initial topology (the event ctr), or the occurrence of messages on gates $e_1$, ..., $e_n$. The control of topology configuration is contained on con[1..n] variable.

On the other hand, it can be noted that we have used a data structure. The dynamic network specification is described more clearly with this data structure, since the occurrence of events $e_1$, ..., $e_n$ means that there are some messages to be sent or received through the network. So that, it is assumed that there is a defined data type called *mess*.

Another Lotos process is the *processors* process. We can specify it as follows:

```
process Processors [e1, ...,en,pctr] :=
 proc_1 ||| proc_2 ||| proc_3 ... ||| proc_M
endproc
```

In this way we can see that the processors are composed by means the parallel operator |||. Each processor can perform three different tasks: it can do its particular computations, to communicate to another processor and to ask for the topology change. The *computation* task depends on the concrete parallel problem, so that it will be discussed in particular cases. The *communication* task can be defined in several ways: one directional, bidirectional, one or several channels and so on.

The last Lotos process to be defined is the *Change* process. This process is the responsible for carrying out the adequate changes on the network topology. Since we are defining a multicomputer system with dynamic reconfiguration, we need a Lotos process that describes concisely the reconfiguration. We have the next specification:

```
process change [ctr,pctr](x:mess,y:intstr):=
 i;pctr?x:mess;makeChange(x:mess) >> ctr!y:intstr;exit
endproc
```

Two observable actions are showed by this Lotos process: the pctr action and the ctr one. When a processor asks for a topology change, it sends a message to the system controller of the network. This request is defined by means the pctr gate and the x variable. At this point all processors are interrupted and waiting for a new network reconfiguration.

## 3. Conclusions

In this paper we have presented a new application of Lotos technique, the multicomputer specification. We focus our attention in the dynamic reconfiguration of the network. We have showed the main features in the development of such specifications.

From this point, we have a model that will help us to study the main features of the dynamic reconfiguration technique. This formal methodology will be used to improve some aspects like stability, scalability, reconfiguration quality, promptness and so on. Moreover, we can use a Lotos tool called Notos [Gal93] to verificate the specifications. This tool simulates the behaviour of a system by means of a prototype. We can test if the specification of the system is the desired.

## REFERENCES


[Bol89] T.Bolognesi and E.Brinksma. "Introduction to the ISO Specification Language LOTOS". The Formal Description Technique LOTOS, results of the ESPRIT/SEDOS project, pp. 23-71. North-Holland, 1989.

[Gal93] J.A. Gallud and J.M. Garcia. "Notos: Un Generador de Prototipos para Especificaciones Lotos". Panel 93, pp. 450-461, Buenos Aires, 1993.

[Gar93] J.M. Garcia and J. Duato. "Dynamic Reconfiguration of Multicomputer Network: Limitations and Tradeoffs", in P. Milligan and A. Nuñez (Eds.), Euromicro Workshop on Parallel and Distributed Processing, IEEE Computer Society Press, pp. 317-323, 1993.