Consiglio Nazionale delle Ricerche

ISTITUTO DI ELABORAZIONE DELLA INFORMAZIONE

PISA

THE MUTEAM SYSTEM: GENERAL GUIDELINES

F. Grandoni, F. Baiardi, G. Cioffi, P. Ciompi, P. Corsini, A. Fantechi, G. Frosini, L. Lopriore, L. Simoncini, A. Tomasi, M. Vanneschi

L81-02

Estratto da: FTCS-11, the eleventh international Symposium on Fault-tolerant Computing (Portland, 1981)

THE MuTEAM SYSTEM: GENERAL GUIDELINES

F. Grandoni*, F. Baiardi**, G. Cioffi***, P. Ciompi*,
P. Corsini°, A. Fantechi**, G. Frosini°°, L. Lopriore*
L. Simoncini*, A. Tomasi**, M. Vanneschi**

*Istituto di Elaborazione della Informazione, C.N.R., Pisa, Italy.
**Istituto di Scienze dell'Informazione, Università di Pisa, Italy.
***Istituto di Automatica, Università di Roma, Italy.
°Istituto di Elettronica e Telecom., Università di Pisa, Italy.
°°Istituto di Informatica, Università di Ancona, Italy.

The MuTEAM is a prototype system, supported by the National Computer Science Program of the Italian National Research Council (C.N.R.), whose implementation is in progress in Pisa, and it is scheduled to be operative at the end of 1981.

The aim of the prototype is the development of integrated design methodologies for distributed multimicroprocessor systems for real time applications. The main goals are modularity, expandibility, error confinement and robustness: such goals have been approached by the definition of a system organization that unifies hardware configuration, fault tolerance and operating system kernel requirements.

The system is conceived as a hierarchical structure, composed of several virtual levels. Starting from the higher level specifications, the next lower level functionalities have to be excerpted, without any previous binding of specific functions to specific levels. This means, for example, that fault treatment needs not to be delegated only to hardware level. We decided that, in the first phase, a major concern should be survivability. So, as the very first hypothesis, we have to think of a system where the presence of faulty subsystems is normal. Then, the fault treatment strategy is to be integrated in all the system levels.

The higher level is composed by active entities, processes, characterized, in our conception, by being mutually suspicious. Processes do cooperate to accomplish system objectives, exchanging informations among them. Owing to the suspicious attitude, no information is trusted as "a priori" reliable: it must be controlled and validated at each exchange.

Owing to the system layered structure, "underground" interferences must be avoided. To prevent uncontrolled interferences, protective rings must be inserted inside the structure which supports the functionalities at higher level, one for each higher level "user".

Neat separation between entities is a fundamental requirement to execute system diagnosis, that is monitoring of subsystems correctness. This is a basic function of a survivable system, as it is the first step to the organization of available modules to accomplish normal tasks. System diagnosis is assigned to a peculiar kind of processes, diagnostic processes, which form the next layer in the abstract level hierarchy.

In the structuring of diagnostic processes it was recognized that any centralized element would be a single point of catastrophic failure; therefore it was decided not to implement such a vital function on centralized structures, whether logical or physical one. Theoretical foundations for distributed self-diagnostic systems have been established in the literature. The problems arising in the implementation of theoretical models are exposed in /1/ together with the proposed solutions, which will be experienced in the MuTEAM design. Using self-diagnostic techniques avoids the need of a central hard-core to assure the correctness of diagnostic decisions.

Major features of these techniques are:

a) each unit, having a logic identity in the system, is locally responsible of its decision for the system diagnosis, which is globally validated by the cooperation among the units;
b) the behaviour of units is "non-aggressive": that is, no one in the system is given means to, for example, disconnect some other unit from the power bus. Malfunctioning units are simply ignored by closing communication channels connected to them.

The next virtual level, i.e. the operating system kernel, is responsible for managing the logical communication channels. The interprocess communication structure best suited to support explicit information exchange is one based on message-passing techniques, since the shared variables approach lends itself to uncontrolled, "underground" information passing if not very carefully used.

A message-passing approach permits, instead, a neat encapsulation of each environment, since only explicit interactions are possible. The natural characteristics of the adopted approach are

Reprinted from FTCS-11, THE ELEVENTH ANNUAL INTERNATIONAL SYMPOSIUM ON FAULT-TOLERANT COMPUTING, June 1981

stressed by the model of the programming language adopted: each interaction is subject to the mutual control of the communicating processes; moreover, messages are typed, and this is a mechanism to implement message validation. The MuTEAM kernel may be seen as the run-time support for the mentioned language. All system software will be programmed in this language. The description of the MuTEAM kernel is in /2/.

The next level is constituted by hardware configuration.

The first parameters to be taken into account are performance figures, e.g. throughput, response time, interprocess data rate, etc. Among the initial specifications of the MuTEAM project there was the need for a high communication rate, attainable only through a commmunication structure based on a parallel bus. It was recognized that high speed information exchange is required only inside groups composed of a limited number of processes, while inter-group communications occur at lower rates. The overall system architecture was so based on a cluster structure /3/. The first step in the project development has been the design of the cluster structure. To each Computer Element of the cluster a fixed set of processes is assigned and the virtual machine supporting their executions is given by the operating system kernel. Interprocess communications are based on a single structure, composed by the parallel bus and the shared memory. As discussed before, protective rings must be inserted in the basic elements.

At the processor element level there is the need of a ring for each process, as well as for the kernel.

Protective rings in the communication structure give rise to a major architectural influence, since the shared memory space cannot be conceived as a uniform, freely accessible space. It must be given instead a logical structure, relying on a rigid physical implementation, which can be conveniently pictured by the segmentation concept. The global address space (or virtual address space), i.e. the set of addresses of all cluster shared memory locations, is to be partitioned into (logical) segments. Each segment has to be given access control mechanisms. A communication channel between two logical subjects is established by giving to both subjects some access rights for a single segment.

The management of access rights is physically and logically distributed. The shared memory is organized in blocks, one block per Computer Element. The access to each memory block is controlled by a specific protection mechanism, which, in turn, can be managed only by the associated processor. With this organization, no single entity is allowed to access any shared space without the granted permission of a "controller". The management of the protection mechanism is local and a malfunction in one processor can only influence the memory banks which are phisically associated to it.

The insertion of an intelligent communication controller for the managing of interprocessor interrupts, allows to cope with those failures which cause continuous interrupts coming from a failed processor.

The protection mechanism and the use of the intelligent communication controller allows us to operate a logical reconfiguration inside a cluster.A failed Computer Element is logically disconnected by associating to each shared segment it can access a NoRight access right and refusing any attention it may request.

The next phase in the development of the MuTEAM Project will be the evaluation of the proposed solutions for the operating system and for the distributed diagnostics. Moreover logical reconfiguration will be implemented on the first version of the prototype.

Problems related to the insertion of hardware monitoring for on-line fault detection, modification of the common bus communication structure, to avoid singularities and a global approach to the recovery of the system will be the main research arguments in the next years.

ACKNOWLEDGMENTS

The MuTEAM project is the result of the joint efforts of several people and institutions. Among them, we should like to thank: M. Boari, A. Natali of the Istituto di Automatica, Università di Bologna; P. Velardi of Istituto di Automatica, Università di Roma; A. Orlandi of Oto Melara S.p.A., La Spezia; R. Cardini, A. Ciuffoletti, M. La Manna, L. Strigini of Selenia S.p.A., Roma. A particular acknowledgment to Mrs. L. Orsucci of Selenia S.p.A., for the typing effort and her patience.

REFERENCES

/1/ P. Ciompi, F. Grandoni, L. Simoncini, "Distributed Diagnosis in Multiprocessor Systems: the MuTEAM Approach", this issue.

/2/ F. Baiardi, A. Fantechi, A. Tomasi, M. Vanneschi, "Mechanism for a Robust Multiprocessing Environment in the MuTEAM Kernel", this issue.

/3/ G. Cioffi, P. Corsini, G. Frosini, L. Lopriore, "MuTEAM: Architectural Insights of a Distributed Multimicroprocessor System", this issue.

These works have been supported by the National Computer Science Program of the Italian C.N.R.