Viewpoint Oriented Real-Time Parallel Systems Development

ZHANG Lichen, GALINDO Michel, MARQUIE Daniel and RAYNAUD Yves
IRIT/SIERA, UPS
118 Route de Narbonne
31062 Toulouse,France

ABSTRACT: In this paper, we propose a new approach to real-time parallel systems development which explicitly avoids the use of a simple framework to design complex systems. Instead, this new methodology is proposed according to five views: environment, function, behavior, performance, material and besed on the partition of the system deveopment life cycle into six stages. In each stage, specific techniques are applied.

KEY WORDS: software engineering, viewpoint, real-time, parallel processing.

1. INTRODUCTION

Typical real-time systems involve complex sequences of events, actions, conditions and information flow, often with explicit timing constraints. Failure to meet timing constraints could cause a catastrophe. Distributed systems will likely be the best accepted solution for throughput enhancement of large real-time applications composed of highly concurrent processes, but this does not mean that timing constraints will be met automatically. In fact, the following types of constraints and requirements arise in the development of real-time parallel systems: decomposition of system into tasks, timing constraints, resource constraints, precedence relationships, concurrency constraints, communication requirements, allocation the elementary functions on parallel architecture, even utilization of each processor.

It is fair to say that the problem of finding good methods to aid in the development of such systems has not been satisfactorily solved. Some methods have been proposed for real-time system design [1][2][3][4][5][6], but they are not suited for complex system.

These methods have the gap beween the levels of specification, software architecture and hardware architecture. In addition, they do not provide a solution for guaranting the deadline of real-time tasks. It is in this context that we have been studying real-time parallel system design since 1989 under contract supported by French National Space Study Centre (in French, Centre National d'Etudes Spatiales (CNES)). The project is organized into three phases: state of the art, proposition of an original design methodology, and validation of the solution proposed on an application of CNES.

This paper is structured as follows. The section two depicts our modeling methodology by emphasizing the main aspects and techniques of each stage. The following section overviews the application that is used to validate the proposed methodology.

2. DEVELOPMENT METHODOLOGY

In the second phase, we have proposed a design methodology of parallel real-time systems according to five views: **a view of environment of system, a view of function of system, a view of behavior of system, a view of performance of system, and a view of material of system.** The methodology is based on the decomposition of the system development life cycle into six stages. In each stage, specific techniques are applied.

Stage 1: definition of requirements: The first stage is concerned with determing what is to be built and delivered, how fast il must operate, how good it must be , with what it will interface, how much it will cost and most important, what part the software will perform. Techniques for specifying requirements formally do exist but do not suit for complex real-time system. Instead, requirements will be expressed in **structural nature language [14].**

Stage 2: analysis of the environment of system: The second stage is concerned with: stating the objects concerned by the environment of system, determing the operations, events, informations concerned by the objects, describing the behaviour of each object.

The objects of environment have the following characteristics: internal strength cohesion and weak coupling between the objects. Therefore, **the HOOD [5] /ESTEREL [6] integration or HOOD/SDL [7] integration** are adapted for analyzing the environment. HOOD is used for the identification of objects of the environment, and ESTEREL or SDL are used for the behavioural description of the elementary objects. ESTEREL is reserved for critical applications, whereas SDL is adapted for less critical applications.

Stage 3: refinement of the system: The refinement strategy is globally top-down and consists in the execution of a set of basic steps where a given function is broken down into sub-functions. Each such sub-function is in turn defined and broken into other lower level components in following refinement step. A top-down decomposition of a system stops up to elementary functions. In this stage, we make **the extension on the method SADT [13]** to permit expression of three types of relations between functions: synchronization by event, represented by the line ——————, transfer of information by variable of state, represented by ----------, transfer of information by channels, represented by ———.

Stage 4: analysis of the behavior of system: We describe the behavior of system with its environment. The control activites can be specified by the behavioral view. These must be present on any level of the activity hierarchy, controlling that particular level. These controllers are responsible for specifying when, how and why things happen as the system reacts over time. We make use of the **Statecharts [4]/PRTIL [9] integration** to analyze the behavior of system. Statecharts, extensions to Finite State Machines (FSM), were recently proposed by Harel. Statecharts make it even easier to model complex system behavior without ambiguity. The extensions provide a notation and set of conventions that facilitate the hierarchical decomposition of FSMs and a mechanism for communication between concurrent FSMs, the transitions depend on global conditions and on being in particular states. PRTIL is used for specifying the condition that guards the transition.

Stage 5: evaluation of performances: During this stage, the analysis and simulation models of the system may be developed, based on a user defined multi-processors or a hypothetical multi-processors. These models help determine performance bottlenecks in the system and assist in selecting the hardware configuration. Rough estimates of processing times and communication times for all the elementary functions are developed, generally based upon lines-of-code estimates. These estimates are tallied to verify whether system fits within harware and processing timeliness constraints.

In order either to simplify the model so that an analytic method is made applicable, or to reduce the number of event to be simulated, **aggregation methods [10]** are particularly well suited to parallel architecture modeling, since these architecture are very often composed of similar, even idential, components that constitute choice aggregates.

Stage 6: implementation: Firstly, some constraints related to design solution are introduced for refining the functional structures. Secondly, the repartition between hardware and software is determined according to the results of evaluation of performances, the hardware must be specified. After the repartition between hardware and software, the elementary functions are allocated among parallel processors to achieve following goals: meeting all timeliness constraints, meeting precedence constraints between functions, minimizing communication cost between processors, balancing utilization of each processor. The above allocation meets certain optimization criteria without specifing the order of execution of functions on a processor, we can use such an allocation pattern, in conjunction with heuristic rules for ordering task allocated to each processor and the communication and synchronization between functions on a processor can be realized by hardware or software. The realization of the communication and synchronization between processors depends on the nature of relation between the functions.

The heuristic method is to provide fast and effective algorithms for a suboptimum solution of allocation. It is useful in application where an optimun solution is not obtainable within a critical time limit and is also applicable to larger dimensional problems. We make use of a function allocation model based on a **zero-one programming technique, the branch-and-bound method [11]**. The design of a model for allocation involves the following steps: 1) formulate the cost function to measure the interprocssor communication cost and processing cost by using the results of evaluation of performance, 2) formulate a set of constraints to meet the diverse requirements by using the results of evaluation of performance, 3) derive an iterative algorithm to obtain a minimum total cost solution.

The relation "variable of state" between functions is realized by memory. The mutual exclusion is commonly employed when using shared variables. The relations " synchronization " and " transfer of information by channels" are realized by interrupt or boolean variables or semaphores or physical channels.

3. EXAMPLE OF APPLICATION

An application has been proposed by CNES for validation of the methodology proposed in the second phase, and the prototype is constructed by use of Transputers and OCCAM [8].

(1) SYSTEM REQUIREMENTS

The simulator must be able to provide the transfer frame that complies to the CCSDS Recommendations and is identical with the frame which is produced by an onboard generator. The primary requirements for this simulator are as follows: 1) to simulate application data and to create source packets, 2) to multiplex these different packets on a virtual channel, 3) to generate the transfer frame, 4) to add CRC or R-S code, and insert the Synchronisation marker and serialize message for generating Physical Channel Access Protocol Data Unit. The Figure 1 specifies the primary functions of system by method SADT (13).

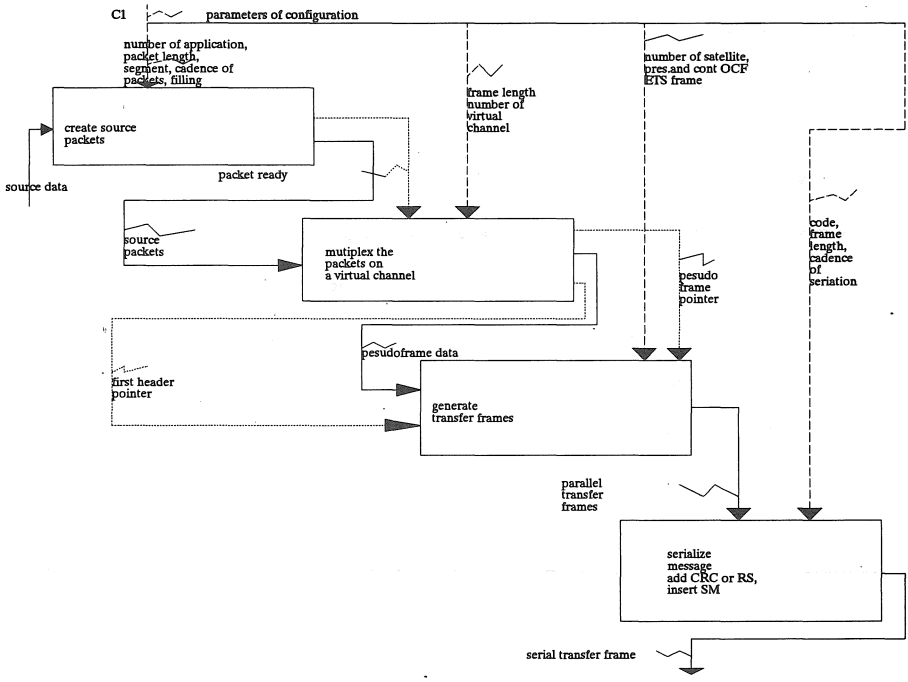

Fig. 1 primary requirement of the simulator

(2) HARDWARE AND EXECUTIVE SOFTWARE

In our simulator, the system consists of five transputers made up from a Volvox 1/A board [5], which fits inside an IBM PC AT, containing one T425 processor and a Volvox 4/A containing four T425 processors. The executive software is written in OCCAM language. In the simulator, the fourth function is realized by a serial generator. This hardware communication with Transputer network is by an Inmos C012 link Adaptor.

(3) COMMUNCATION PERFORMANCE ANALYSIS

One important characteristic of the application is the great communication amount between the Transputers. This causes message transfer delay. We evaluate the influence of various parameters on message delay. The following factors are of great importance to message delay (Fig. 2, Fig. 3, Fig. 4): the number of processes on a processor, message size, probability of transfer message of processes.

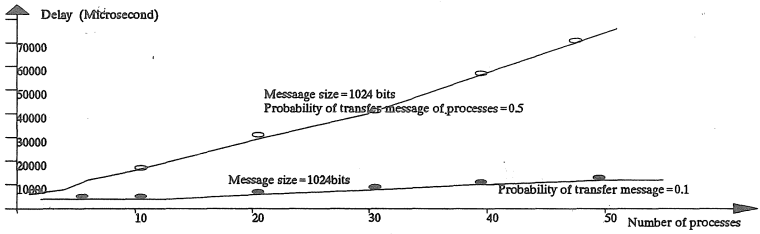

Fig. 2 Delay in function of number of processes

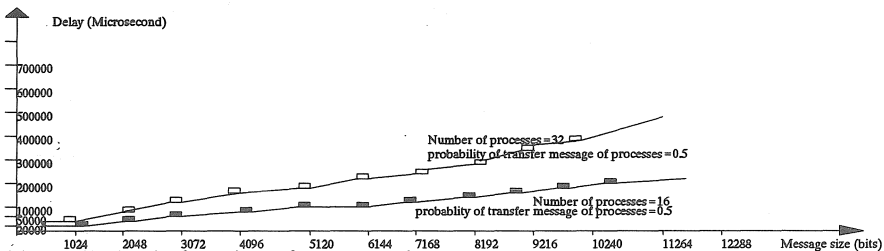

Fig.3 Delay in function of message size

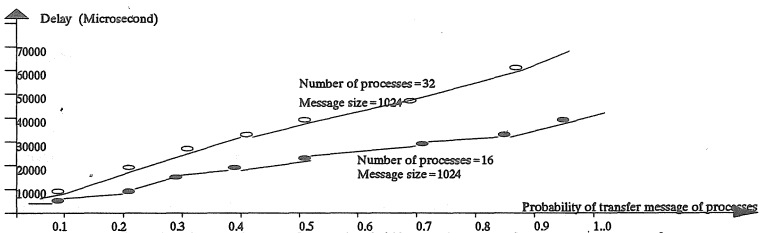

Fig. 4 Delay in function of probability of transfer message of processes

(4) ALLOCATION

Any allocation must first satisfy the communication requirement between the Transputer because the communication cost is greater than the processing cost in our application. Since the processing cost and communication cost is scenario dependent, the allocation can be optimized only for a given scenario.

4. CONCLUSION

In conclusion, we may say the our methodology is adatpted for real-time parallel system design because we do not attempt to use a simple framework to design complex system. The application has shown that the proposed methodology is complete, the results of evaluation of performances are useful for allocation of processes on multiprocessor and provide the solution for garanting the deadline of a real-time task.

REFERENCES

1. M.Jackson, System development, C.A.R HOARE series, Prentice-Hall 1983.

2. Joint IECCA and MUF Committe on MASCOT (JIMCOM), The Official Handbook of MASCOT. Her Majesty's Stationery Office (June) 1987).

3. P.T. Ward and S.J.Mellor, Structured development for real-time systems, Yourdon Computing series-YOURDON PRESS-Prentice-Hall 1985.

4. D.Harel, Statecharts, a visual formalism for complex systems, Science of computer programming North Holland, Vol.8, 1987, P.231-274.

5. HOOD Working Group, HOOD Eeference Manual, draft B issue 3.0 (June 1989).

6. G.Berry and G.Gonthier: "The synchronous language ESTEREL: Design, Semantics and Impementation" INRIA report 842, 1988.

7. CCITT: "Fonctional specification and description language (SDL)", Recommandations Z.101-Z.104, Vol. VI, Fasc. VI.7, Genova 1981.

8. E.Hirsh, Les Transputers: Application à la programmation concurrente, Editions Eyrolles 1990.

9. K.T. Narayana and A.Aaby, Specification of real-time systems in real-time temporal interval logic, Proceeding of real_time systems symposium, IEEE Computer Society Press, 1988.

10. G.Pujolle and S.Fdida, Modèles de systèmes et de réseaux,Editions Eyrolles, Paris, 1989.

11. P.Richard, Y.Edward and M.Tsuchiya, A task allocation model for distributed computing systems, IEEE Transactions on Computers 31(1): 41-47,1982.

12. CCSDS 102.0-B-2, "Recommendation for Space Data System standards. Packet Telemetry.", CCSDS Recommendation, Blue Book, Issue-2, January 1987 or later issue.

13. I.G.L. Technology, SADT, Un langage pour communiquer, Editions EYROLLES, 1989.

14. D.J.Hatley and I.A.Pirbhai, Strategies for real time systems, Rourdon computing series-Yourdon Press Prentice Hall 1985.