The design of distributed, dependable, real-time systems using a functional paradigm

A. Bondavalli$^1$, L. Simoncini$^2$ and C. Bernardeschi$^2$

1 CNUCE-CNR, Via S.Maria, 36, Pisa, Italy e-mail: andrea.bondavalli@cnuce.cnr.it
2 Department of Information Engineering, University of Pisa, Via Diotisalvi, 2, Pisa, Italy e-mail: simon{cinzia}@iet.unipi.it

1 Abstract

Distributed real-time computer systems are replacing conventional control systems in many applications. Most of these systems are highly critical, e.g. design of a traffic flight control or an industrial process control system. In addition to the specified functional capabilities, these applications demand predictable timeliness and a high level of dependability which includes non-functional attributes such as availability, reliability, safety and maintainability [Lel]. The systematic development of fault-tolerant real-time systems with guaranteed timeliness and dependability requires an appropriate system architecture and a rigorous design methodology.

2 Design methodology

To achieve high level of dependability, the classical techniques of fault avoidance, fault tolerance, fault removal and fault forecasting must be combined with the use of formal methods. In this way faults in the design and development of the system can be eliminated or reduced. Dependability must be designed in a system and cannot be considered as an add-on after the system has been developed. The key notion in dependability is that reliance must be justifiably placed on system service [Lal]. This means that we need explicit and testable requirements and specifications to be used in a top down refinement design of the system. Therefore, a design methodology, in addition to the structural description of the system, should include more information to allow proving typical design properties such as timing and dependability. The use of a functional language, together with a dataflow computing model for the design of dependable large-scale parallel computing systems, have been shown to be a basis of a design description language satisfying many of the challenging requirements of predictable distributed systems [BS1]. In [BS2] a design methodology has been defined which provides:

- a design description language and some extension for dealing with dependability issues, with the creation of a library of fault tolerance schemes which can be used for modular insertion of redundancy;
a set of tools as part of a design development environment allowing the designer to proceed in successive interactive steps, each of which can be validated;
- a set of design constraints on the supported architecture.

The design description language is based on coarse-grained dataflow notation which, quite apart from its natural ability to describe highly parallel applications, has other very interesting advantages:
- modules composing a system have functional behaviour and locality of effects; these imply composability which puts in direct relation the general behaviour of a system from its constituent parts;
- modules are atomic and satisfy referential transparency; they work in local environments and two executions of the same module with the same input data produce equal results, simplifying re-execution or recovery of a module;
- systems structured in this way are inherently fault-tolerant in the sense that simple failures may be tolerated by simply resubmitting the same inputs to a module describing the same function;
- structural models for software reliability assessment can be applied since all data necessary to their use can be obtained by a simple instrumentation of software code;
- the data-flow graph representing the system can easily be translated, with proper techniques, into some sort of timed Petri-net for timeliness analysis purposes.

The construction of an interactive design environment, populated by a set of tools for the automatic insertion of fault-tolerance in the system, for the automatic translation into analysable timed Petri-nets and into analysable stochastic Markov systems for reliability analysis, and for the evaluation of structural models for software reliability, is feasible. The interactive approach is used for calibrating the design so that dependability and timeliness requirements can be attained with a high level of predictability.

References


This article was processed using the \LaTeX macro package with LMAMULT style