Model-Checking a Group Membership Protocol for TDMA-based Networks with both Static and Dynamic Scheduling

Valério Rosset *, Pedro F. Souto and Francisco Vasques
Faculdade de Engenharia da Universidade do Porto
vrosset@fe.up.pt, pfs@fe.up.pt, vasques@fe.up.pt

Abstract

We report ongoing work on a group membership protocol specially designed to take advantage of the support of both static and dynamic scheduling in new TDMA-based protocols being proposed for safety critical applications, such as Flex-Ray. In contrast with previous group membership protocols for TDMA-based networks, ours does not require the pre-allocation of group membership traffic in every cycle. Currently we are working on the formal verification of its correctness using the UPPAAL model checker. This will provide a higher assurance of correctness of the protocol, which is of foremost importance in safety critical applications.

1. Introduction

It is widely accepted [5] that services such as group membership and distributed agreement (also called interactive consistency) facilitate the systematic development of safety-critical applications. In these applications, group membership services are used mainly to keep track of the operating components (we will use the term processors) of the system. Therefore, they comprise essentially two tasks: failure detection, by which a processor determines the operational state of the other processors in the system, and agreement, by which non-faulty processors agree on the state of all the processors in the system.

Virtually all group membership (GM) protocols [3, 2] for TDMA-based networks perform these two tasks in every TDMA cycle. In these protocols, failure detection is based on missed scheduled messages or on non-valid messages, whereas agreement is done through the exchange of the perceived processors state. Performing the GM protocol in every cycle used to be a straightforward design decision, as virtually all TDMA-based protocols supported only static scheduling and all possible traffic had to be pre-allocated. Recently, however, a new class of communications protocols based on TDMA that supports both static and dynamic scheduling has been proposed. One notable example of these protocols is Flex-Ray, the next-generation protocol for the automotive industry.

In a previous paper [4] we presented a new GM protocol that takes advantage of this scheduling ability to achieve better quiescent performance for the same fault model than previously published GM protocols. This protocol relies on the observation that agreement needs to be executed only when events that may lead to changing the GM occur, and not in every TDMA cycle. Therefore the protocol executes agreement only on the dynamically scheduled part of a TDMA cycle, and, in a quiescent state, the bus bandwidth required for agreement can be used by other aperiodic traffic. This feature has the potential advantage of relieving the pressure on using strong fault assumptions, which require less communication, and of allowing shorter TDMA cycles to meet shorter deadlines.

In order to make the protocol easier to understand, in [4] we presented three versions of the protocol: we started with a basic protocol that executes in every TDMA cycle and that does not support processor reintegration, then we modified that protocol to support processor reintegration, and finally we presented the final version of the protocol that does not require the execution of agreement in every TDMA cycle. This approach was also essential in developing the correctness arguments for the protocol. However, proving the correctness of fault-tolerant distributed protocols is a notoriously hard task, and incorrect protocols, and proofs, have been published in the past. Given that safety-critical applications require a high assurance that they operate correctly, we are currently working on the formal verification of the protocol using the UPPAAL model checker.

2. Model Checking

Model checking allows the verification of the correctness of a system by checking whether the desired properties are satisfied in every reachable state of that system.

*Supported by FCT through scolarship SFRH-BD 19302/2004

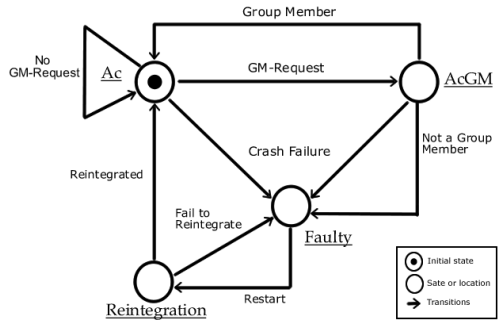

Figure 1. Skeleton of the Processor automata for the final version of the GM protocol.

In [4] we used the set of group members (M-SET) maintained by every processor to specify the GM problem. We considered two properties: Agreement and Validity. Agreement states that all non-faulty group members must compute the same M-SET. Validity states that the M-SET of non-faulty processors must mirror the operational state of processors in a timely fashion. The protocol proposed to solve the stated problem, consists of two phases that are executed in every TDMA cycle: the Failure Detection phase (FD-phase) and the Group Membership phase (GM-phase). Every processor is required to transmit in the FD-phase so that failures can be detected in a timely fashion. Agreement is achieved in the GM-phase. In the final version of the protocol, if no group membership change event occurs the GM-phase does not require the exchange of any message – stated in a different way, the GM-phase is optional.

2.1 Model Overview

UPPAAL is a toolbox for the verification of real-time systems, based in timed automata, a generalization of finite state-machine [1]. Our model of the protocol is composed of three types of automata: RoundSlotCounter, Verifier and Processor. The RoundSlotCounter simulates the behavior of communication protocol and controls the length of slots. The Verifier is an observer automaton that is used to check the Agreement and the Validity properties at the end of every TDMA cycle. The Processor automaton models the behavior of one processor an is the core of the model.

Figure 1 shows a simplified version of the Processor automaton for the final version of the GM protocol, i.e. for the GM protocol with process reintegration and agreement only when needed. It is is composed of four main *locations* (the circles) Location Ac is active if the processor is a group member, independently of being faulty, and the protocol is in the FD-phase. Location AcGM is similar to location Ac, the difference is that the processor is executing the GM-phase of the protocol. The Faulty location is active if the process is faulty and does not consider itself a group-member, or if it crashed. Finally, the Reintegration location is active when a process is trying to become a group member. The figure also shows *edges* between locations. These edges are labeled with the *guards* that enable the transition from one location to another. Note that it is possible to transition from any location to the Faulty location if there is a crash fault. Otherwise, transition to the Faulty location occurs at the end of the GM-phase, either if the process was a group member or if it is trying to reintegrate, when the processor diagnoses itself faulty.

As we mentioned above, this is a simplified version of the automaton developed. The complete automaton includes additional (committed) locations, and corresponding edges. Taking into account faults is the main source of complexity of the model: although we consider the fault model adopted in related work, our protocol requires a slightly stronger fault model to satisfy the Validity property. Because of this we have actually developed two models, one to verify the Agreement property, and the other to verify the Validity property.

2.2 Preliminary Results

We run the UPPAAL verifier on the model developed for the Agreement property and checked it successively for configurations with three, four and five processors.

Checking the Validity property has proved more difficult, because the model is more complex and the verifier runs out of main memory (we used a PC with 1 Gbyte of memory) even for configurations of 3 processors. Currently we are investigating how to simplify this model.

References

[1] G. Behrmannm, A. David, and K. Larsen. *A Tutorial on UPPAAL*. Department of Computer Science, Aalborg University, Denmark.

[2] P. D. Ezhilchelvan and R. de Lemos. A robust group membership algorithm for distributed real-time systems. *Proceedings of the 11th Real-Time Systems Symposium*, pages 173 – 179, 1990.

[3] K. H. Kim, H. Kopetz, K. Mori, E. H. Shokri, and G. Grünsteidl. An efficient decentralized approach to processor-group membership maintenance in real-time lan systems: the prhb/ed scheme. In *Proceedings of the 11th Symposium on Reliable Distributed Systems*, 1992.

[4] V. Rosset, P. F. Souto, and F. Vasques. A group membership protocol for communication systems with both static and dynamic scheduling. In *Proceedings of the 2006 IEEE International Workshop on Factory Communication Systems (WFCS'06)*, 2006.

[5] J. M. Rushby. Bus architectures for safety-critical embedded systems. In *Proceedings of the 1st International Workshop on Embedded Software*, pages 306–323, 2001.