Execution environment for ELECTRE applications

Denis Creusot, Philippe Lemoine, Olivier Roux, Yvon Trinquet
LAN-ENSM, Ecole Nationale Supérieure de Mécanique, Equipe Temps Réel
1, rue de la Noé, F-44072 Nantes Cedex 03, France

Antonio Kung, Olivier Marbach, Carlos Serrano-Morales
TRIALOG Informatique
9, rue du Château d'Eau, F-75010 Paris, France

Abstract

This paper describes an execution environment for reactive systems specified in ELECTRE. ELECTRE allows the specification of a real-time application's temporal behaviour in terms of sequential entities called modules, of events, of relations between modules like parallelism, and of relations between modules and events like preemption. ELECTRE is based on a design and implementation approach enforcing the separation of the sequential part of the application (i.e. module specification), the event part of the application (i.e. event specification), and the control part of the application (i.e. reaction to events). This separation is also reflected at the execution level which includes a control unit, a module unit and an event unit. The execution environment is supplemented by a display system, which can be used for simulation, debugging or monitoring purposes. The display system is a multiwindow facility based on two main types of representations: a structural representation and a temporal representation.

Keywords: Reactive systems, real-time parallel systems, visualisation and monitoring, execution system.

1 Introduction

This paper describes a specification, programming and execution environment based on ELECTRE [Elloy 85], acronym for Exécutif et Langage de Contrôle Temps-réel REparti (Language and Executive for Distributed Real-Time Control), a language allowing the description of the temporal behaviour of real-time control processes.

ELECTRE is one among several approaches intended to model a real-time system during its specification stage in order to be able to perform tasks such as analysis and formal checking of timing and event properties. Other approaches are specific "Real-Time Logics" [Jahanian 86, Ostroff 90], the use of event histories [Dixon 86, Faulk 88], the use
of data-flow diagrams [Ward 86] or petri net analysis [Valette 88], and specific languages [Harel 90, Aeuernheimer 86]. The approaches of CCS [Milner 80] and Lotos [Brinksma 85] have influenced our work. ELECTRE is a specific language allowing the specification of behaviours concerning such notions as process preemption and process blocking. It does not support the description of the sequential process itself. ELECTRE can be classified as an asynchronous language, adapted for the programming of reactive systems, real-time systems in particular.

Among other functions, ELECTRE can be used in the following three ways:

- Validation of temporal behaviour specification. A simulator [Creusot 88] is currently available. It accepts external stimuli representing events, and shows to the user the modification of the state of the ELECTRE specification.

- Monitoring the dynamic behaviour of an application. ELECTRE expressions are used as redundant specifications, either to support debugging as in [Bruegge 83], or in order to provide a fault-tolerant mechanism for limiting detection latency of faults due to synchronization errors.

- Programming specification of the application’s concurrency aspects. ELECTRE is used as a programming language. This is the approach which we have chosen and that we describe in this paper.

The rest of the paper is structured as follows. We first briefly describe ELECTRE. The kind of specification and programming environment in which ELECTRE can be included is then sketched, and the resulting execution environment, with a particular emphasis on display facilities, is presented in detail. Finally a conclusion with a description of current research is provided.

This project is partially funded by the French Ministry of Industry under the STRIN programme. The initial development of a simulator [Creusot 88] within the Electre project has been funded by Renault-Automobiles. It was used as a specification tool in the field of embedded systems.

2 Brief description of the ELECTRE language

Like some other languages [Benveniste 91], the ELECTRE language emerged from research concerning effective ways to express process behaviour and synchronization in reactive systems. [Pnueli 86, Harel 85, Benveniste 91] characterize those systems (e.g. real-time systems) by their reactions to external stimuli (e.g. sensors signals) in order to produce outputs (e.g. actuator commands). Moreover, those systems have to react to and act on an environment that constraints the reaction rate.

ELECTRE is based on path expression theory [Campbell 74]. Path expressions were mainly developed for the synchronization of concurrent processes sharing a resource. They allow the description of how concurrent processes are coordinated in the sharing of a resource. They are a convenient way of expressing constraints that processes must meet in order to guarantee that actions concerning a given shared object are executed in an orderly manner.