Titre : Machines d'exécution pour langages synchrones.

Résumé : L'approche synchrone est d'un grand intérêt pour la conception de systèmes réactifs temps réel sûrs. Toutefois, en raison des hypothèses synchrones, les programmes issus de cette approche ne sont pas directement exécutables. Leur implantation nécessite une *machine d'exécution* dont le rôle est d'établir le lien entre le modèle idéal synchrone et le monde réel asynchrone. La machine d'exécution est souvent considérée comme un détail d'implémentation et traitée au cas par cas. Notre objectif est de définir une *architecture générique* de machines d'exécution qui permette d'avoir une approche unifiée pour l'implantation des processus synchrones.

L'architecture proposée est basée sur le modèle de *boîtes réactives*. Ce modèle permet de concevoir des machines d'exécution qui partagent une structure générique commune, mais chacune répondant à des besoins particuliers en termes de connexion à l'environnement, de mode de fonctionnement et de traitement des anomalies. L'exécution de traitements externes non instantanés (*tâches* du langage ESTEREL) en parallèle avec le code synchrone est aussi traitée et intégrée dans l'architecture.

Nous avons développé des outils de *génération automatique* basés sur cette approche. Ils permettent la configuration et la production de machines exécutables pour le système Solaris 2. Les bibliothèques sont écrites en `C++` et reposent largement sur les *objets synchrones* de F. Boulanger.

Nous illustrons l'utilisation de cet environnement sur l'exemple d'un ATM (*Automatic Teller Machine*). Nous nous intéressons aussi à la *vérification de propriétés logiques*, d'abord du programme synchrone puis de l'implémentation elle-même (propriétés "de bout en bout" portant sur les signaux physiques).

Mots clefs : systèmes réactifs, systèmes temps réel, langages synchrones, ESTEREL, machines d'exécution, implémentation, architecture logicielle, tâches, objets.

Title: Execution machines for synchronous languages.

Abstract: The synchronous approach is suitable for designing dependable real-time reactive systems. However, because of the synchronous hypotheses, synchronous programs are not directly executable. Execution of such programs requires an *execution machine* whose role is to bridge the gap between the synchronous ideal model and the asynchronous real world. Execution machines are often considered as implementation details. Our goal is to define a *generic architecture* for execution machines that brings a unified approach to synchronous program implementation.

The architecture we propose is based on the *reactive box* model. This model allows the design of execution machines based on the same generic structure, but customized according to the specific needs in terms of connection to the environment, functionning mode and exception handling. Execution of non instantaneous external computations (ESTEREL's *tasks*), which run concurrently with the synchronous code, is also considered and integrated into the architecture.

We have developed tools for *automated generation* based on this approach. These tools support configuration and generation of executable machines on the Solaris 2 system. Libraries are written in `C++` and rely on F. Boulanger's *synchronous objects.*

We illustrate the use of this environment on the example of an ATM (*Automatic Teller Machine*). We also take interest in *logical properties verification*: first, properties of the synchronous program, then properties of the actual implementation (*"end to end"* properties concerning physical signals).

Keywords: reactive systems, real-time systems, synchronous languages, ESTEREL, execution machines, implementation, software architecture, tasks, objects.