Mini-Conf.

Statecharts

Un outil de modélisation puissant pour la programmation réactive.

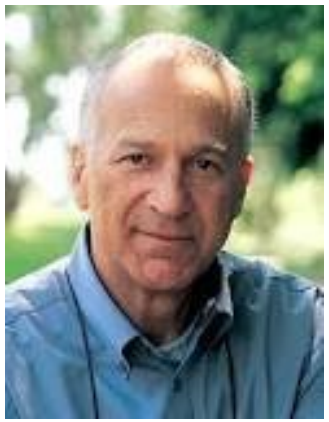
17 juillet 2015

Prof. Jean Demartini

SOFAB
By Telecom Valley

Agenda

Contexte
Contraintes
Problématique
Principes
Représentation
Implémentation



Prof. David HAREL
Computer Science and Applied Mathematics
The Weizmann Institute of Science
Rehovot - Israël

SOFAB By Telecom Valley

Contexte

- Le développement d'un firmware se traduit presque toujours par une pièce de logiciel fonctionnant en mode *réactif*.
  - le comportement du système est la réponse aux stimuli qu'il reçoit de l'extérieur.
  - les systèmes où les traitements de données sont prépondérants sont dits *transformationnels*.
- Le modèle conceptuel correspondant est bien connu : *machine à états finis*.
- Lorsqu'on ne peut pas se permettre de surdimensionner le processeur utilisé ou que les performances à atteindre sont particulièrement élevées, l'implémentation est réalisée en *bare-metal*.
  - pas de noyau temps-réel
- La technique d'implémentation la plus efficace s'appuie sur l'utilisation de fonctions *call-back*.

SOFAB By Telecom Valley

Contraintes

- **Déterminisme** : une MAEF est dite *déterministe* si elle réagit de la même façon à deux séquences d'événements identiques.
    - identiques depuis le démarrage de la machine.
- **Synchronicité** : la durée de traitement d'un événement est supposée nulle.
  - sur un plan pratique on fera en sorte :
    - de ne pas perdre d'événements (avec une probabilité la plus faible possible)
    - de s'assurer que leur ordre d'arrivée est conservé.
- Restriction : il est impossible de faire évoluer deux MAEF *simultanément*.
  - on est donc obligé de fondre les deux machines en une seule ce qui introduit une (explosion) combinatoire des états.
  - mais que veut dire simultanément ?

SOFAB By Telecom Valley

Problématique

- À la base de la MAEF : des états, des événements, des actions.
  - les actions peuvent être associées aux états (Moore) ou aux transitions (Mealy).
- Le concept d'événements « gardés » permet de diminuer le nombre des états nécessaires.
  - la « garde » est un prédicat qui permet d'introduire des variantes d'un même état (états cachés).
- Mais il n'existe pas de mécanisme d'abstraction pour les MAEF.
- Remarque : l'abstraction fonctionnelle est largement utilisée pour les systèmes transformationnels.

Bibliographie

- Les diagrammes de transition d'états deviennent rapidement illisibles et le comportement de la MAEF correspondante est difficile à déduire.
  - il manque un mécanisme d'abstraction qui permettrait de raffiner une description en mode top-down ou bottom-up.
- David HAREL a publié en 1984 l'article :

  STATECHARTS : A VISUAL FORMALISM FOR COMPLEX SYSTEMS
  - une extension du formalisme des MAEF pour décrire des systèmes réactifs complexes (protocole de communication par exemple)
  - pour obtenir une description structurée et concise.
- Remarque : l'abstraction fonctionnelle permet de modéliser efficacement les systèmes transformationnels.

Statecharts motivations

« The literature on software and systems engineering is almost unanimous in recognizing the existence of a major problem in
the specification and design of large and complex reactive systems. A reactive system, in contrast with a transfornational system, is characterized by being, to a large extent, event-driven, continuously having to react to external and internal stimuli.

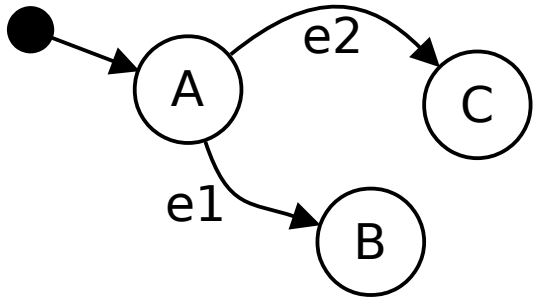
Examples include telephones, automobiles, communication networks, computer operating systems, missile and avionics systems, [industrial control systems,] and the man-machine interface of many kinds of ordinary software. The problem is rooted in the difficulty of describing reactive behavior in ways that are clear and realistic, and at the same time formal and rigorous, sufficiently so to be amenable to detailed computerized simulation [and implementation] ».

SOFAB By Telecom Valley

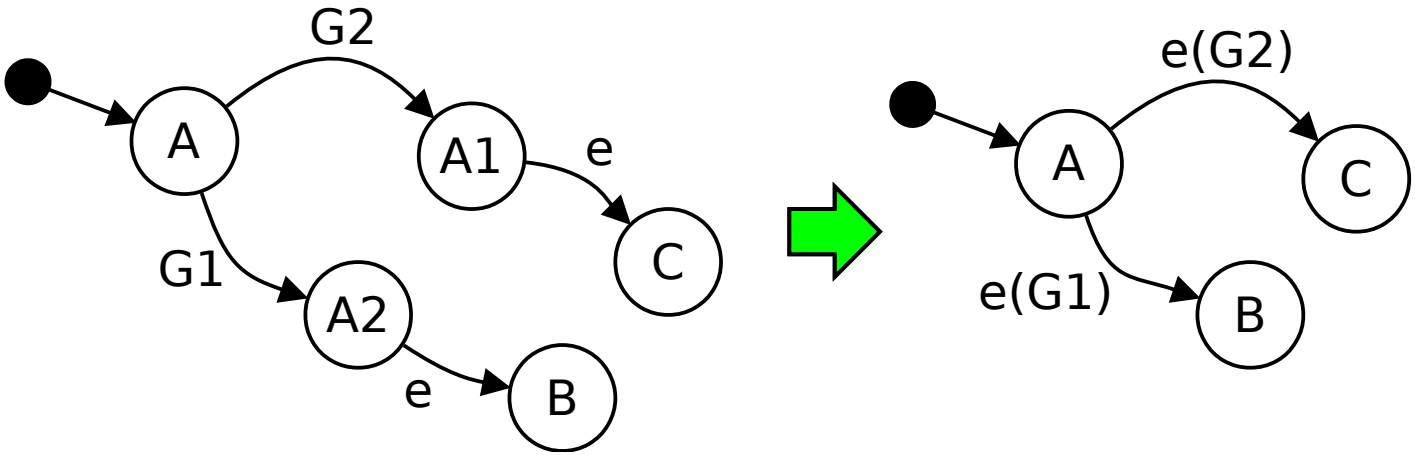
Principes

- Idées sous-jacentes :
  - Définir la simultanéité
    - deux événements seront considérés comme simultanés si le comportement est indépendant de leur ordre d'arrivée.
  - Introduire de « super-états » par *regroupement* (bottom-up) ou par *raffinement* (top-down).
  - Mutualiser certains événements d'intérêt général : ex. Reset, ...
    - particulièrement intéressant pour le traitement des exceptions.
- Objectifs : permettre un description structurée et hiérarchique du comportement d'un système réactif.
  - tout en conservant un support théorique rigoureux.

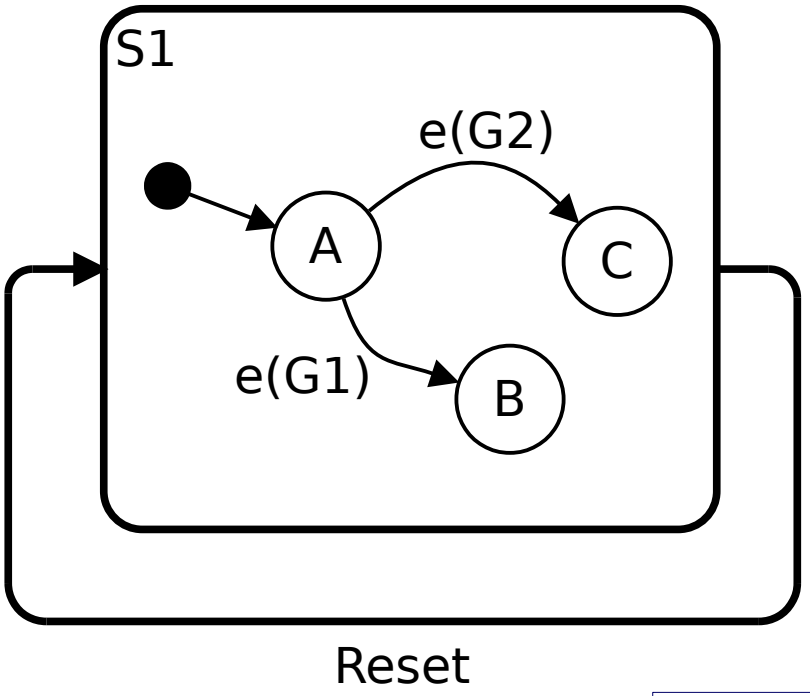
Bases classique

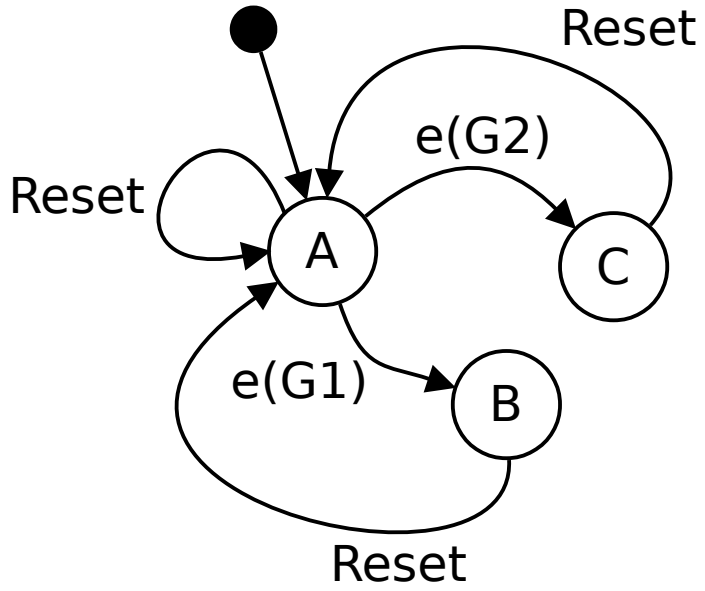


Les événements gardés, mais pas de hiérarchie. apportent beaucoup de concision mais pas de hiérarchie.



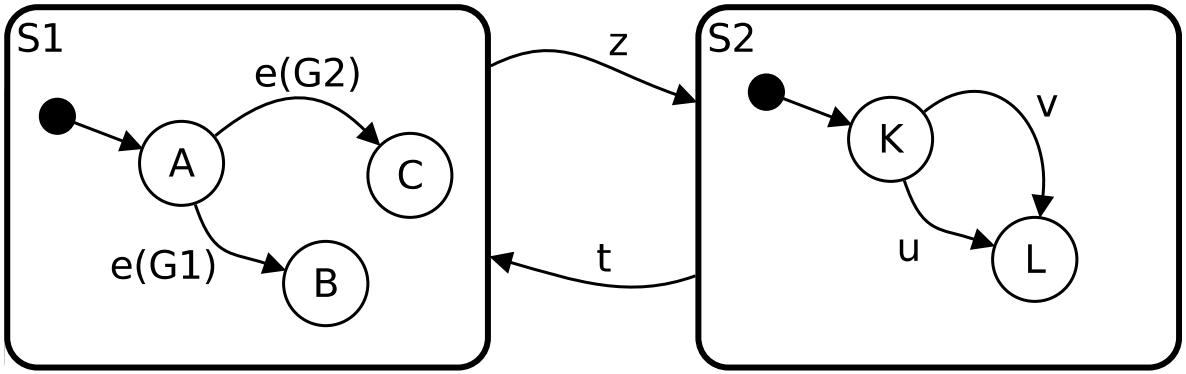
Super-état Mutualisation des événements



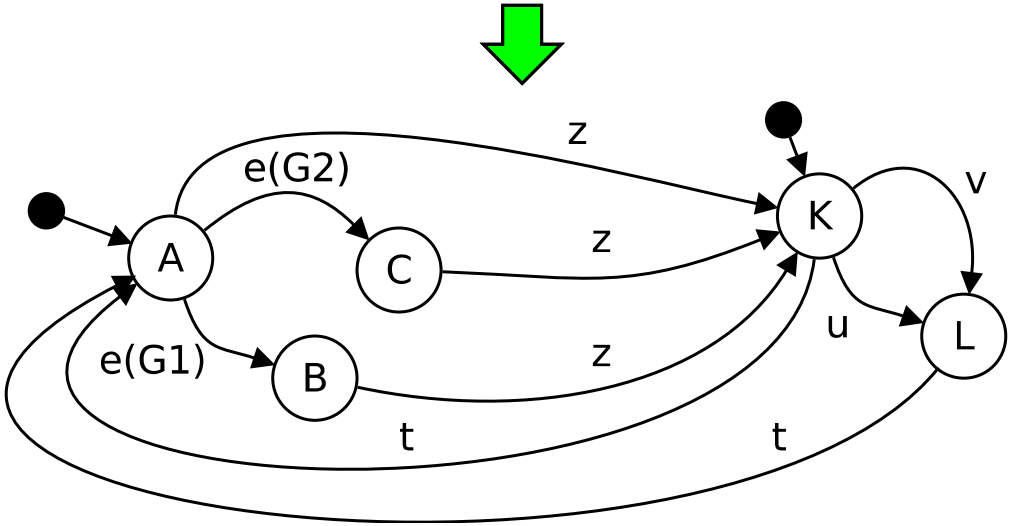


Une entrées beaucoup de sorties. Une description hiérarchique peut être crée.

Hiérarchie



Un super-état peut contenir des états ou des super-états.



SOFAB By Telecom Valley

Guide pour l'implémentation

- Étape 1 : séparer la détection des événements de leur traitement effectif pour conserver leur ordre d'arrivée.
  - détection et mise en file d'attente (FIFO) en foreground (ISR).
    - pas de priorité entre les sources d' interruptions (sauf exceptions).
- Étape 2 : traiter les événements dans leur ordre d'arrivée en tâche de fond
  - gestion de la MAEF
- Étape 3 : vérifier que tous les événements pourront être traités.
  - en principe dans le pire des cas (worst case design)
  - ce qui conditionnera la puissance du processeur à utiliser
- En cas de famine de certains événements on vérifie que certains traitements ne sont pas transformationnels et ne justifient pas un co-processeur dédié.

SOFAB
By Telecom Valley

Questions
Discussion
Échanges