Heurs et malheurs de la conception par modèles en automatique

Paul Caspi *Verimag-CNRS*

caspi@imag.fr **http ://www-verimag.imag.fr**

21 novembre 2003

Trois petites histoires …

- **Airbus de SAO à Scade**
- **Lustre et les langages synchrones**
- **Jusqu'où iront Matlab/Simulink/Stateflow ?**

… et une conclusion

Airbus de SAO à Scade

Un choix stratégique au début des années 80 :

Quel langage informatique choisir pour mener à bien l'A320, premier avion commercial à « commandes de vol électriques » (logiciels critiques) ?

- **Pascal, Modula, C, ADA, réseaux de Petri ?**
- **SAO, formalisme « maison », recyclage des schémas-blocs analogiques en outil de spécification puis de génération automatique de code**

Intérêts de SAO

Double :

- **Passage direct par compilation, des modèles d'automatique au code :**
  beaucoup moins d'erreurs

- **Langage graphique, proche de la culture des ingénieurs aéronauticiens, pilotes d"essai, … :**
  facilite la communication dans l'entreprise
  accélère la mise au point
  capitalise le savoir-faire et facilite la transition de technologie

SAO participe au succès de l'A320

De SAO à Scade

Mais :

- **SAO est un outil maison, supporté en interne, cher à maintenir**
- **SAO est de plus en plus coupé de la culture informatique**

Dans les années 90, Airbus investit dans Scade :

- **quelques calculateurs de l'A340-600**
- **le formalisme choisi pour l'A380**

D'où vient Scade ?

Lustre et les langages synchrones

Au début des années 80, quelques chercheurs, automaticiens et/ou informaticiens, constatent la mauvaise adéquation des langages et systèmes informatiques pour le temps-réel et l'automatique

- **Milner (Edinburg) (SCCS),**
- **le groupe Grafcet**
- **Harel et Pnueli (Weizmann) (Statecharts)**
- **Berry (Ecole des mines) (Esterel)**
- **Benveniste et Leguernic (Inria) (Signal)**
- **Caspi et Halbwachs (Imag) (Lustre)**
- **…**

Ils proposent des formalismes et des langages pour y remédier

pourquoi tant de Français ?

Lustre et les langages synchrones

Ces langages et formalismes sont

- **fondés sur une notion de temps synchrone, différente de la tradition asynchrone de l'informatique et théorisée par Milner puis Berry**

- **en général rigoureux et bien fondés des deux points de vue, automatique et informatique,**

Sous l'impulsion de Berry et Benveniste se met en place une communauté de recherche française dite des langages synchrones

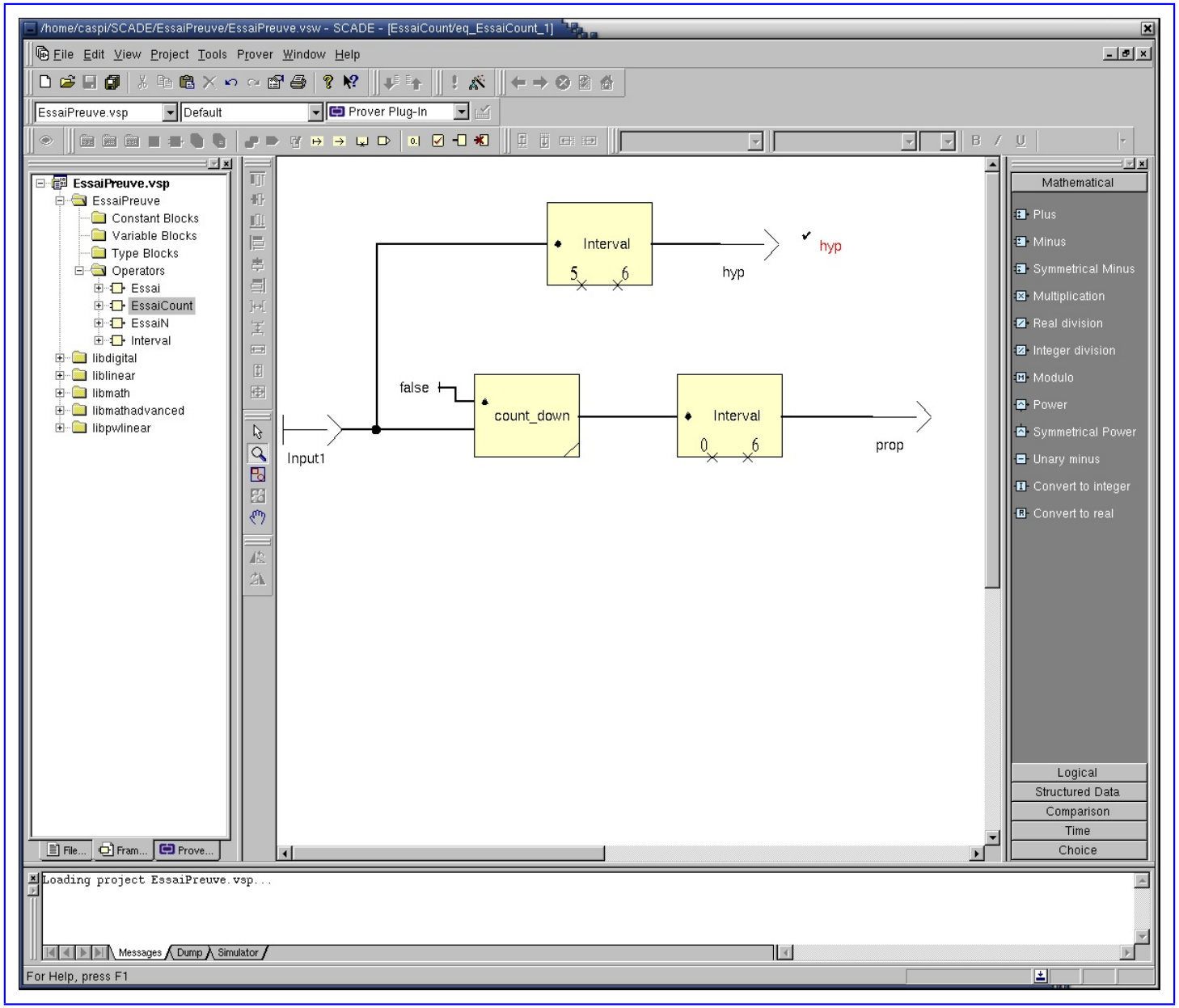
Lustre et les langages synchrones

Histoire de Lustre

- **Deux membres de l'équipe Lustre passent chez Merlin Gerin et l'utilisent pour créer l'atelier SAGA qui sert à concevoir le contrôle-commande du cœur des réacteurs 1450MW Framatome C'est un succès**
- **Devant ce succès, un troisième membre passe chez Verilog et rencontre les gens d'Airbus. La décision de créer Scade est prise.**
- **Verilog est absorbée par CS. Scade sert à programmer la signalisation du métro de Hong-Kong (Csee-Transport)**
- **Scade passe à Telelogic, puis à Esterel-Technologies, dont il devient un produit phare**

Succès industriel Succès commercial ?

Scade


/home/caspi/SCADE/EssaiPreuve/EssaiPreuve.vsw - SCADE - [EssaiCount/eq_EssaiCount_1]
File Edit View Project Tools Prover Window Help
EssaiPreuve.vsp
Default
Prover Plug-In
EssaiPreuve.vsp
EssaiPreuve
Constant Blocks
Variable Blocks
Type Blocks
Operators
Essai
EssaiCount
EssaiN
Interval
libdigital
liblinear
libmath
libmathadvanced
libpwlinear
Interval
5
6
hyp
hyp
false
count_down
Interval
0
6
prop
Input1
Mathematical
Plus
Minus
Symmetrical Minus
Multiplication
Real division
Integer division
Modulo
Power
Symmetrical Power
Unary minus
Convert to integer
Convert to real
Logical
Structured Data
Comparison
Time
Choice
Loading project EssaiPreuve.vsp...
Messages
Dump
Simulator
For Help, press F1

Scade

- interface graphique
- modularité
- vérifications statiques
  - typage, horloges, boucles de causalité, initialisation
- simulateurs
- générateur de code qualifié DO178B niveau A
- sémantique formelle
- model-checker Prover Technologies
- interface Esterel
- interface Simulink

Le meilleur de la technique ?

Jusqu'où iront Matlab/Simulink/Stateflow ?

Pendant ce temps, sur la côte Est …

La société Mathworks est créée en 1984 autour du produit Matlab

Dans les années 90, Simulink, éditeur et simulateur de schémas-blocs apparaît et s'enrichit de

- **nombreuses bibliothèques,**
- **Stateflow éditeur d'automates hiérarchiques et parallèles** *(empruntés aux StateCharts)***,**
- **générateurs de code**

Le succès est très grand et Mathworks est dans la position d'être le Microsoft des systèmes embarqués synchrones

Pourquoi ?

Simulink/Stateflow contient

- **Tout Lustre et tout Esterel**

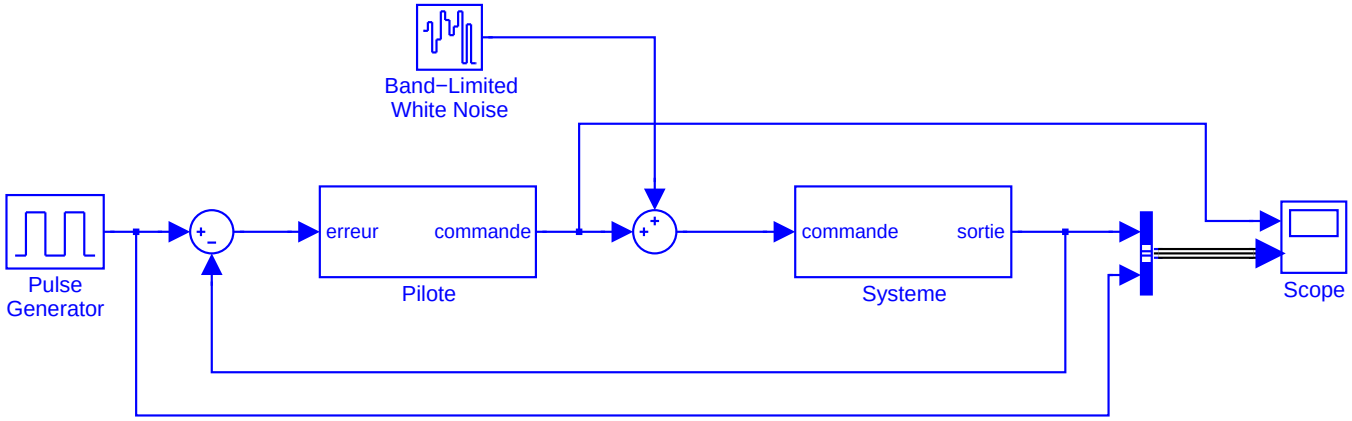
- **Les équations différentielles**
  permettant une modélisation des systèmes à contrôler, la synthèse des contrôleurs et maintenant la génération de code

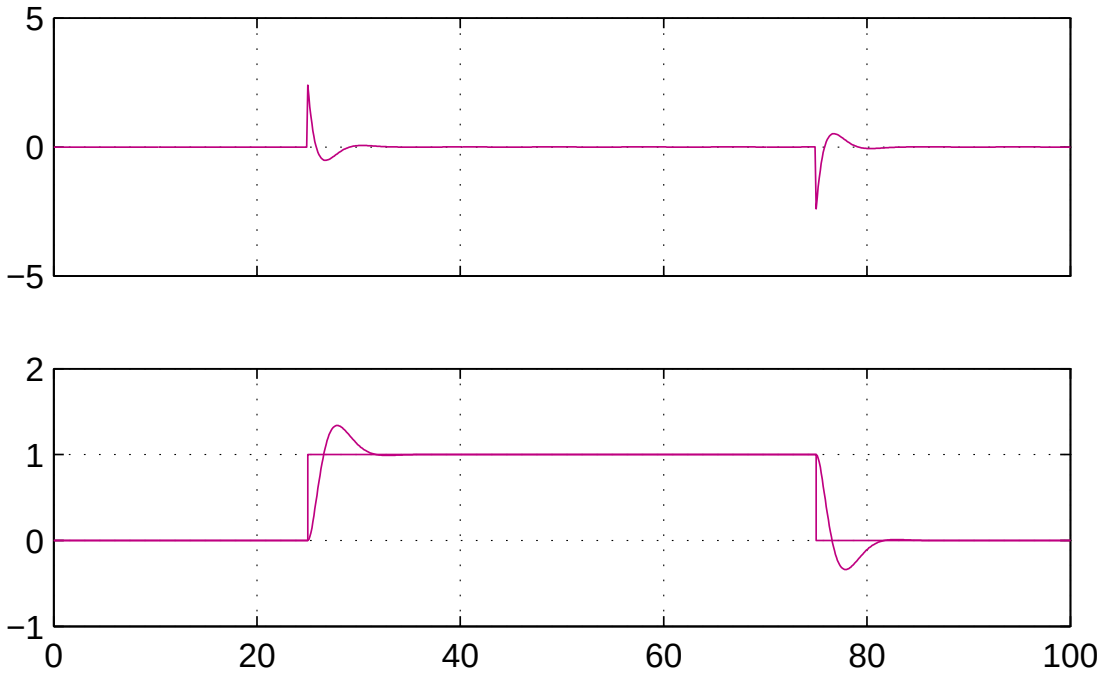
- **De très belles interfaces graphiques**

- **De grandes facilités de simulation et d'observation**

- **…**

Simulink





Time offset: 0

Et pourtant . . .

Simulink/Stateflow est une pure création de numériciens et d'automaticiens

et n'a aucune des qualités informatiques requises pour générer du code pour des systèmes embarqués critiques :

- **Pas de typage** *(du moins initialement)*
- **Pas de sémantique**
  si ce n'est celle du simulateur dépendant de nombreux réglages
- **Des possibilités de non-terminaison**
  (boucles infinies, débordement de piles, . . .)
- **Faibles possibilités de vérification formelle**

Les progrès sont rapides mais les tares initiales nombreuses

Conclusion

Une histoire en trois temps :

- **SAO : les praticiens créent leurs propres concepts**
- **Scade : une synthèse harmonieuse s'opère**
- **Simulink/Stateflow ne va-t-il pas tout avaler ?**

L'embarqué synchrone a été, avec les circuits, le domaine où la conception par modèles est allée le plus loin.

Les succès sont intéressants

La problématique n'est pas encore complètement élucidée

Quelles leçons les autres domaines peuvent-ils en tirer ?