Faithful multi-task implementations of synchronous programs

Paul Caspi

Verimag-CNRS (*emeritus*)

WSS 16 october2009

- historical overview
- multi-tasking

Some safety critical control systems

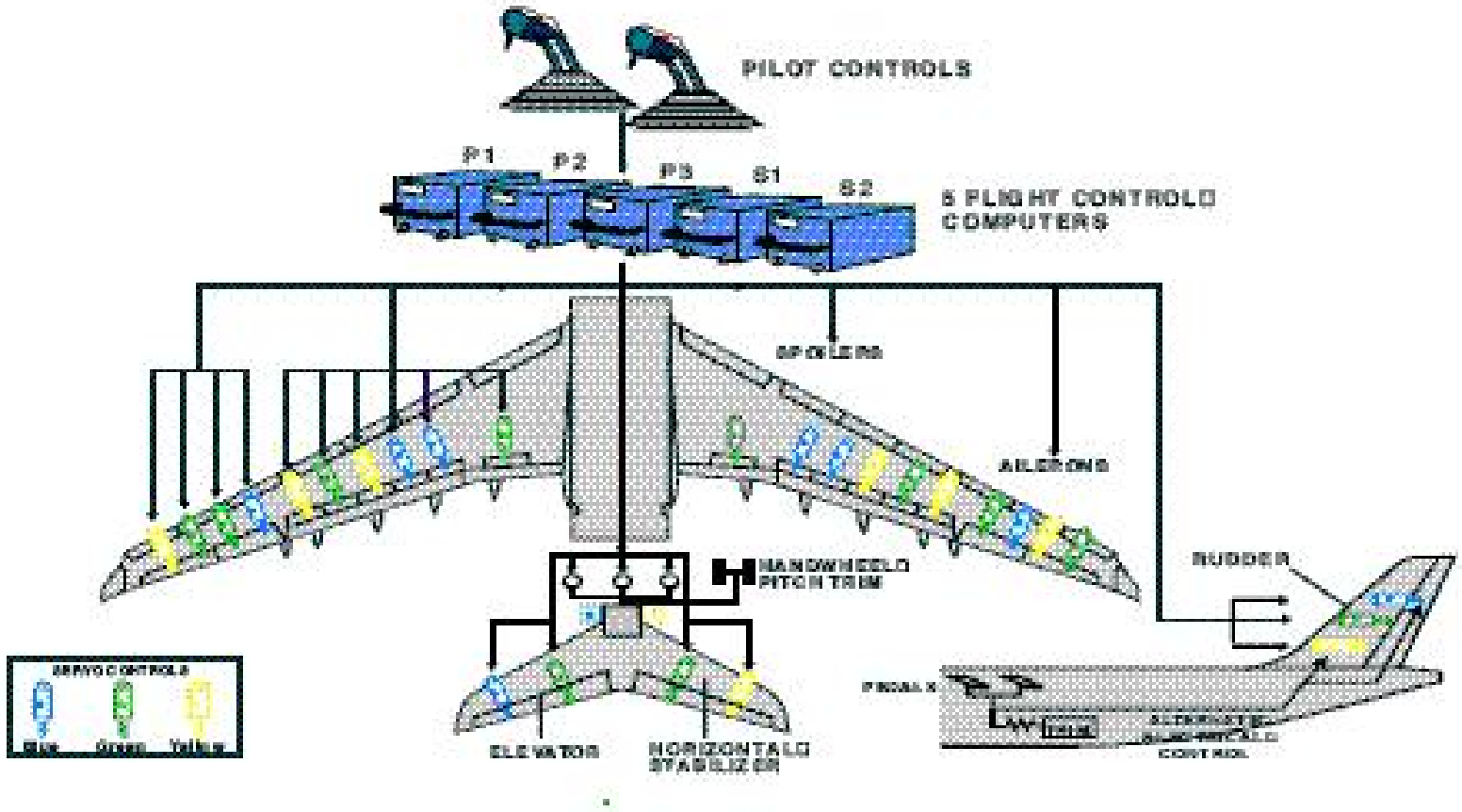
flight control

emergency shutdown

speed control, signalling

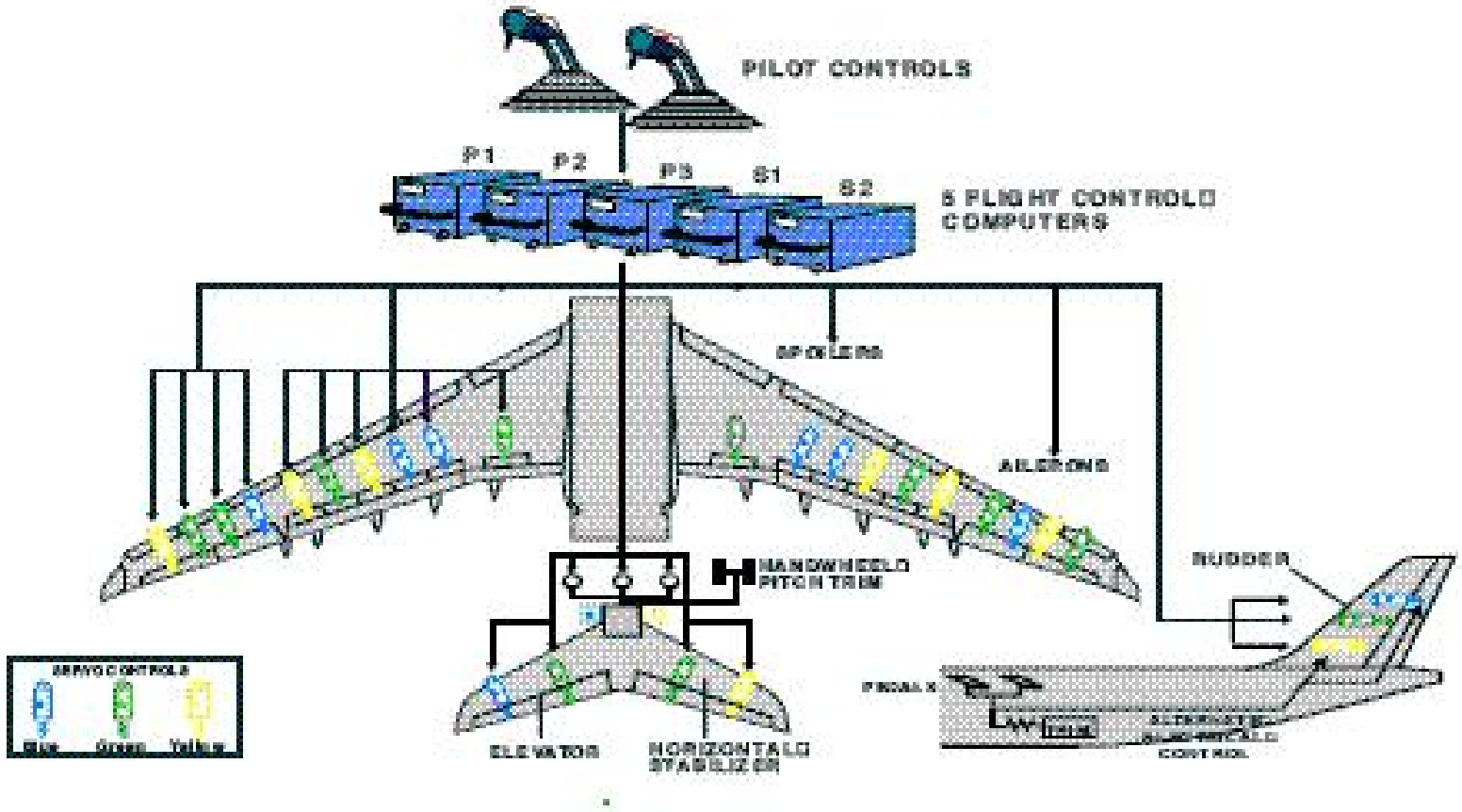
full automation

Looking inside



Fly-by-wire ? Drive-by-wire ? Electronic Control Units ?

Looking inside



Fly-by-wire ? Drive-by-wire ? Electronic Control Units ?

Fly-by-computers ! Fly-by-software !

Two Questions

Knowing the low reliability of computing technology

- thousands of car “recalled” for computing bugs
- Ariane V accident
- your personal computer . . .

Two Questions

Knowing the low reliability of computing technology

- thousands of car “recalled” for computing bugs
- Ariane V accident
- your personal computer . . .

1. *Is it wise to use this poor technology in safety critical systems?*

Two Questions

Knowing the low reliability of computing technology

- thousands of car “recalled” for computing bugs
- Ariane V accident
- your personal computer . . .

1. *Is it wise to use this poor technology in safety critical systems?*

2. *Why, nevertheless, things are not as bad as could be expected?*

Two Questions

Knowing the low reliability of computing technology

- thousands of car “recalled” for computing bugs
- Ariane V accident
- your personal computer . . .

1. *Is it wise to use this poor technology in safety critical systems?*

2. *Why, nevertheless, things are not as bad as could be expected?*

A Tentative Answer

The safety-critical control industry has designed a very strong model-based development method

Some steps in model-based design of control systems

- Early eigthies: first automatic program generation from high level control models
  - SAO (block-diagrams) and the Airbus A320 fly-by-wire
  - Lustre, Signal, Esterel, the French synchronous school
  - ...
- integration into control modelling tool-boxes (early nineties)
  - Targetlink (Dspace), Ascet (Etas), RTW (Mathworks), Simulink Gateway (Esterel)
- Qualified code generation
  - SCADE (Esterel): Do178B Level A, EN 50128 SIL 4
- Connection with formal tools
  - Prover plugins: SCADE, RTW
  - Abstract interpretation
- Time triggered distributed implementations

Interest of automatic code generation

Twofold :

- Automatic code generation from high-level control models:

  $=>$ easier and earlier debugging

- Graphic languages close to the cultural background of application engineers, test pilots, suppliers, certification authorities, . . . :

  $=>$ allows easier communication within the entreprise

  $=>$ preserves the know-how and makes easier the technology transfer

Single Thread Code Generation

- Allows generating code for any discrete-time model that can be simulated
- Allows many optimisations
- The need for Real-Time Operating System is minimised
- Provides in general robust and efficient code
- But in some cases it is very inefficient and even not possible:

need for multi-thread code generation

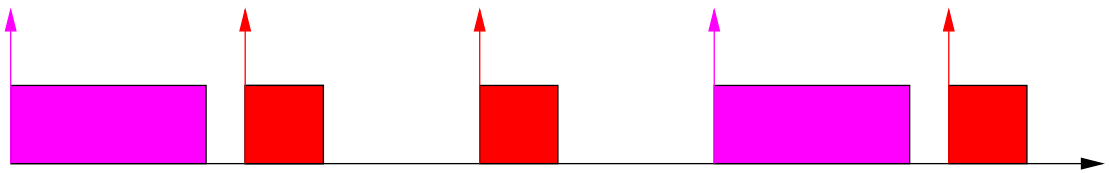
Multi-Periodic Systems

Models are based on null execution times
But implementations take time !!

Example:

- period (3,0)
- period(1,0)

single-thread code generation:
can yield



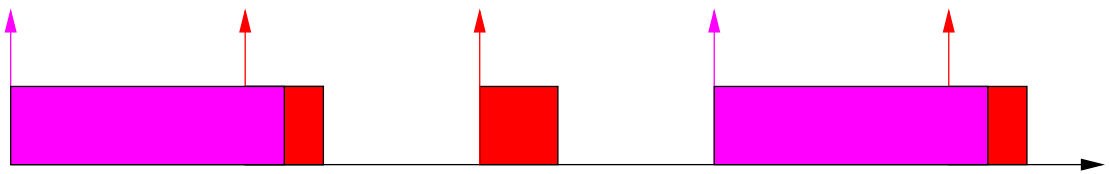
Multi-Periodic Systems

Models are based on null execution times
But implementations take time !!

Example:

- period (3,0)
- period(1,0)

single-thread code generation:
can yield even worse



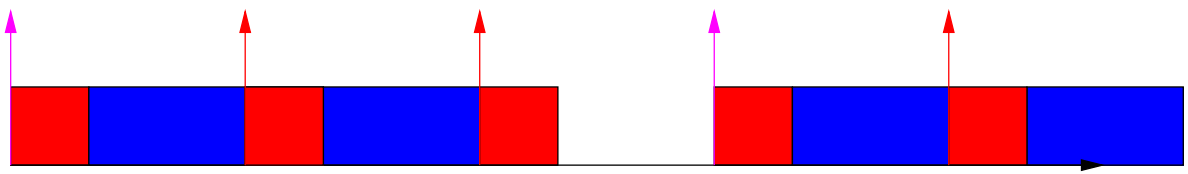
Multi-Periodic Systems

Models are based on null execution times
But implementations take time !!

Example:

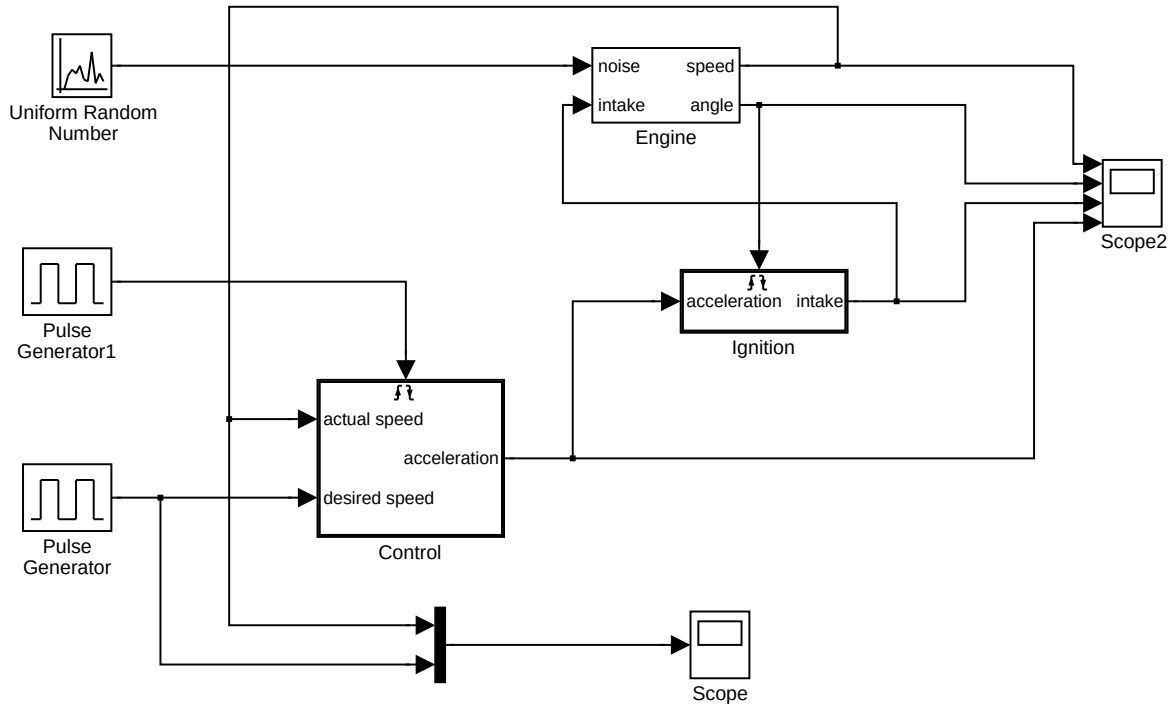
- period (3,0)
- period(1,0)

multi-thread code generation:
and preemptive scheduling can yield



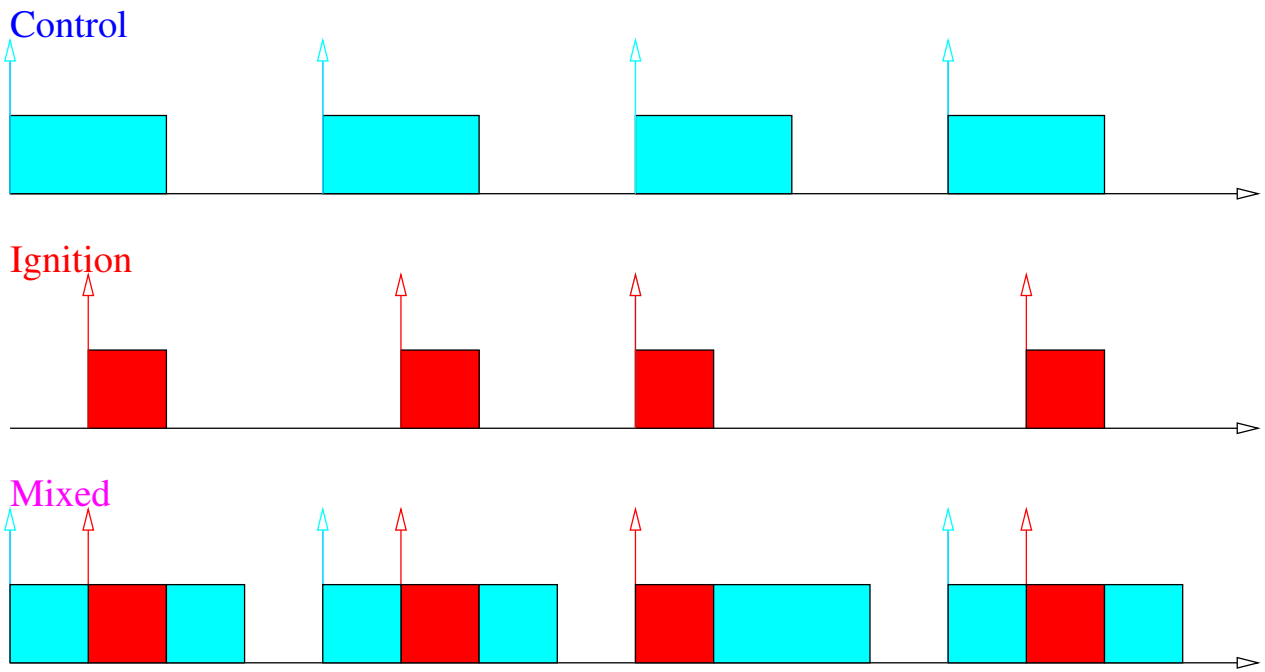
Event and time-triggered systems

An engine control example:



Preemptive scheduling

If the deadline associated with event-triggered computations is smaller than the execution time of time-triggered tasks, preemptive scheduling is mandatory:



Scheduling

Schedulability test: formula of response times

$$R_j = \sum_{i=1,j-1} \left\lceil \frac{R_j}{T_i} \right\rceil C_i + C_j$$

- thread priorities in decreasing order
- T_i minimum inter-arrival time of thread i
- C_i: worst case execution times of thread i
- $\left\lceil \frac{R_j}{T_i} \right\rceil$:number of times j can be preempted by i while executing
- $\left\lceil \frac{R_j}{T_i} \right\rceil C_i$: maximum time during which j can be preempted by i while executing
- The sum is taken on every thread with higher priority

Inter-task communication

Communication integrity, several approaches:

- Blocking approaches based on semaphores
  Priority inversion (pathfinder !!)
  priority inheritance, priority ceiling protocols
- Lock-free methods
- Loop-free, wait-free methods
  Burns et Chen (triple buffer)
  provide easier schedulability analysis ?

What about semantics?

. . . and model-based development?

What about semantics?

. . . and model-based development?

Preemption alters the ordering of computations

- In many cases it does not matter (robustness, continuity, faithfulness. . .)
- In some cases it can (discontinuities, critical races, . . .)

Can we propose executions that be functionally equivalent to the model?

A general solution (Scaife & Caspi 2004)

Ensures communication integrity and provides executions that are functionally equivalent to the model:

Applies to both periodic and sporadic activities

Based on:

1. Syntactic checks: communications from low to high priority tasks should go through a unit delay on the low task trigger

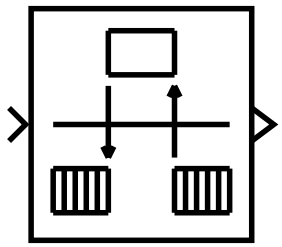
2. Double buffer protocols where distinction is made between the occurrence of triggering events and the task executions

Double buffer protocol

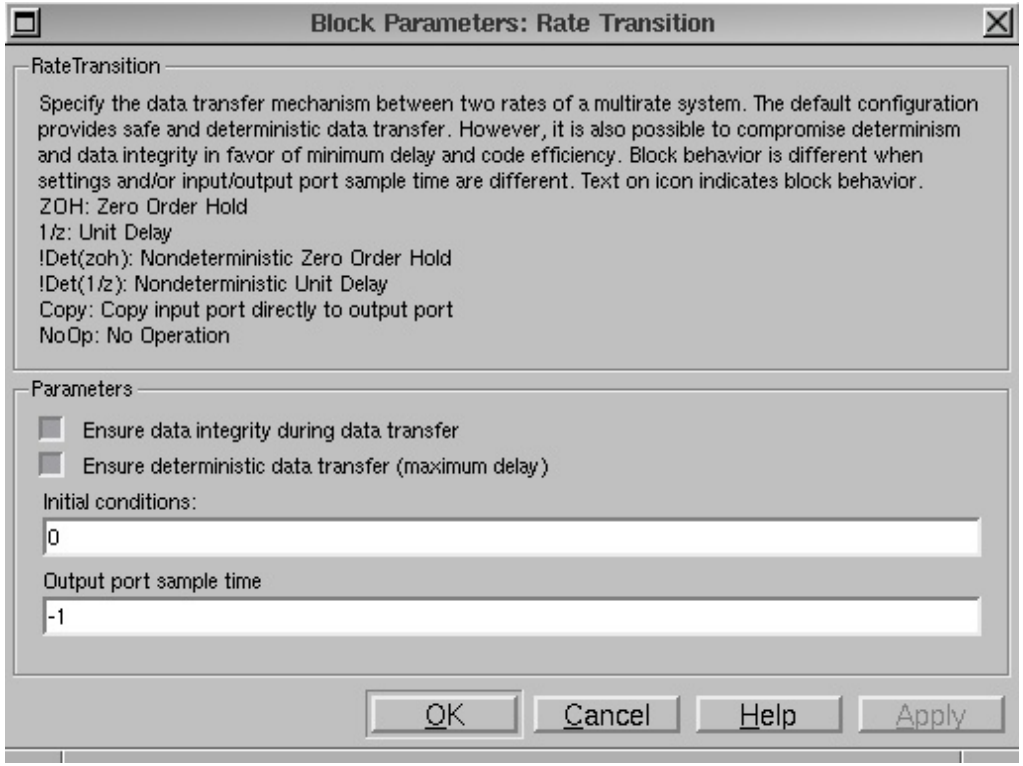
- From low to high
  - two buffers (“current” et “previous”) managed by P_l, toggled when e_l takes place
  - when e_h occurs, P_h stores the address of “previous”
  - P_l writes to “current” et P_h reads into “previous”
- From high to low
  - double buffer (”current” et ”next”) managed by P_l
  - on e_l ”current” is set to ”next”
  - on e_h ”next” is toggled if ”current” equals ”next”
  - P_h writes to ”next” and P_l reads into ”current”
- Bit toggling is assumed to take no time

Another (partial) solution

Simulink RTW...



Rate Transition



Only works for harmonic multiperiodic systems

Other Results

- Proof by Model-Checking
- Generalisation to EDF
  Works the same.
- Several Optimisations
  - Tripakis & al.
  - DiNatale, ASGV & al.

Industrial Perspectives

There seems to be a clear industrial interest :

- RTW
- SCADE

- Partially implemented in the ASSERT European IP

- Parades (Roma) is currently exploring the same ways

A Key Issue: Faithfulness

What you { *model* / *simulate* / *prove* } is what you { *implement* / *execute* }

(Gérard Berry 1984)

From Handicraft to Industry

In twenty years, the industry of critical control moved from

- handicraft :
  - paper design, human coding, validation on hardware
- to industry:
  - functional and architectural design and validation based on formal models that can be simulated and checked,
  - automatic code generation ensuring faithfulness between models and implementations

This is a notable advance that has to be pursued, strengthened and extended

Some references

- N. Scaife and P. Caspi: Integrating model-based design and preemptive scheduling in mixed time- and event-triggered systems, Euromicro Conference Real-Time Systems, ECRTS04, Catania, June 2004
- P.Caspi, N. Scaife, Ch. Sofronis, S. Tripakis: Semantics-preserving multitask implementation of synchronous programs, ACM Transactions on Embedded Computing Systems, 7(2), February 2008
- L. Mangeruca, M. Baleani, A. Ferrari and A. Sangiovanni-Vincentelli: Semantics Preserving Design of Embedded Control Software from Synchronous Models, IEEE Transactions on Software Engineering, Vol. 33, N. 8, August 2007.
- J.-L. CAMUS, P. VINCENT, O. GRAFF: A verifiable architecture for multi-task, multi-rate synchronous software, 4th European Congress ERTS EMBEDDED REAL TIME SOFTWARE, Toulouse 2008 http://www.esterel-technologies.com/technology/WhitePapers/
- http://www.mathworks.com/access/helpdesk/help/toolbox/simulink/
- G.Wang, M. Di Natale and A. Sangiovanni-Vincentelli: Optimizing the implementation of communication in synchronous reactive models with time constraints, to appear in IEEE Trans. INDUSTRIAL INFORMATICS, DECEMBER 2009