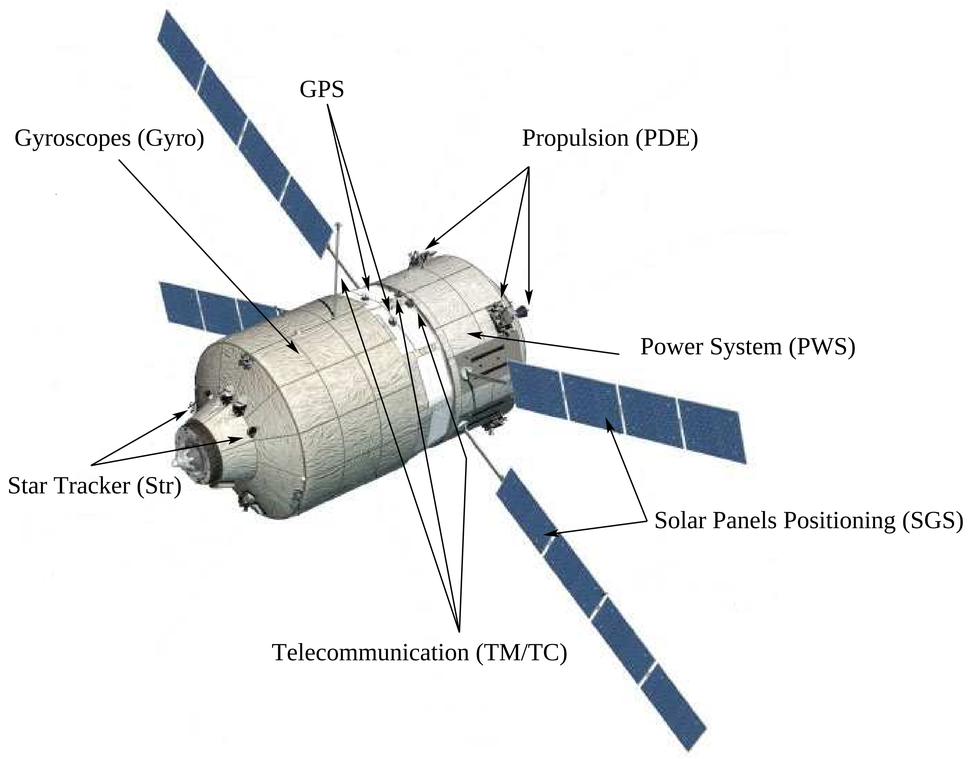
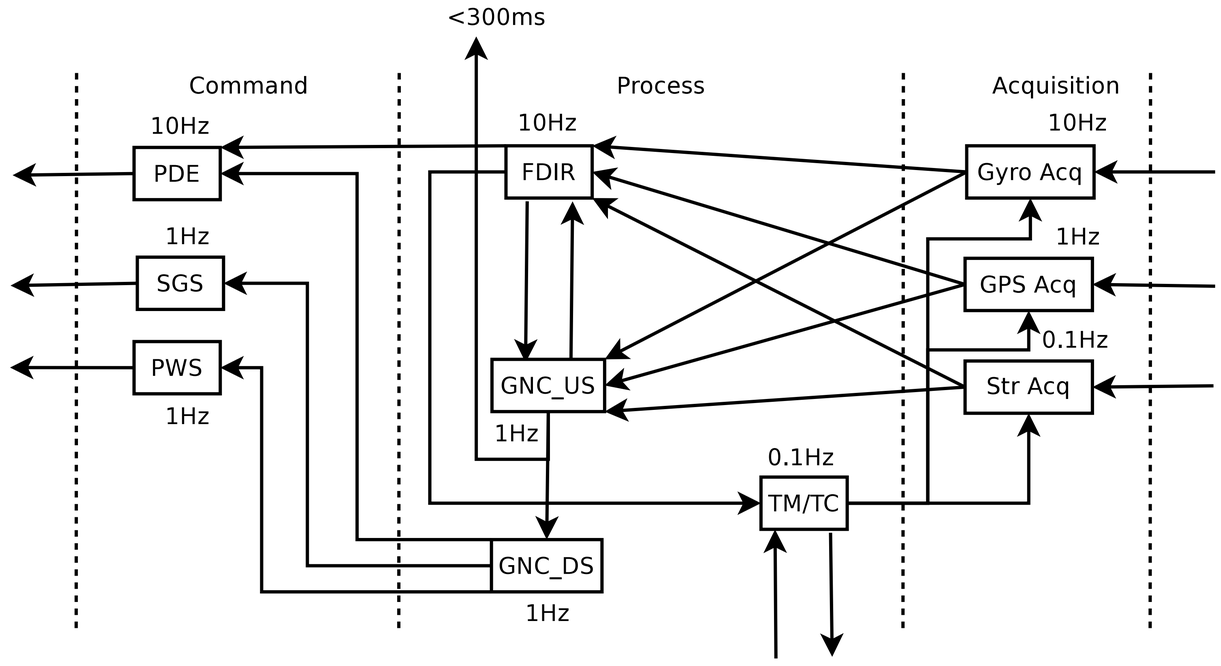
Prelude: Lustre with real-time constraints

Frédéric Boniol, Julien Forget, Claire Pagetti

Context: programming multi-periodic systems





Main characteristics:

- **Multi-rate**: different equipments ⇒ different control rates;
- **Hard real-time constraints**: periods, deadlines;
- Operations of different rates communicate ⇒ multi-periodic communications.

Language: specifying real-time constraints

```
imported node F(i, j: int) returns (o, p: int) wcet 2;
imported node S(i: int) returns (o: int) wcet 10;

node sampling(i: rate (10, 0)) returns (o: due 8)
  var vf, vs;
let
  (o, vf)=F(i, (0 fby vs)*^3);
  vs=S(vf/^3);
tel
```

Data-flow semantics:

date	0	10	20	30	40	50	60	70	80	...
vf	vf_0	vf_1	vf_2	vf_3	vf_4	vf_5	vf_6	vf_7	vf_8	...
vf/^3	vf_0			vf_3			vf_6			...
vs	vs_0			vs_1			vs_2			...
0 fby vs	0			vs_0			vs_1			...
(0 fby vs)*^3	0	0	0	vs_0	vs_0	vs_0	vs_1	vs_1	vs_1	...

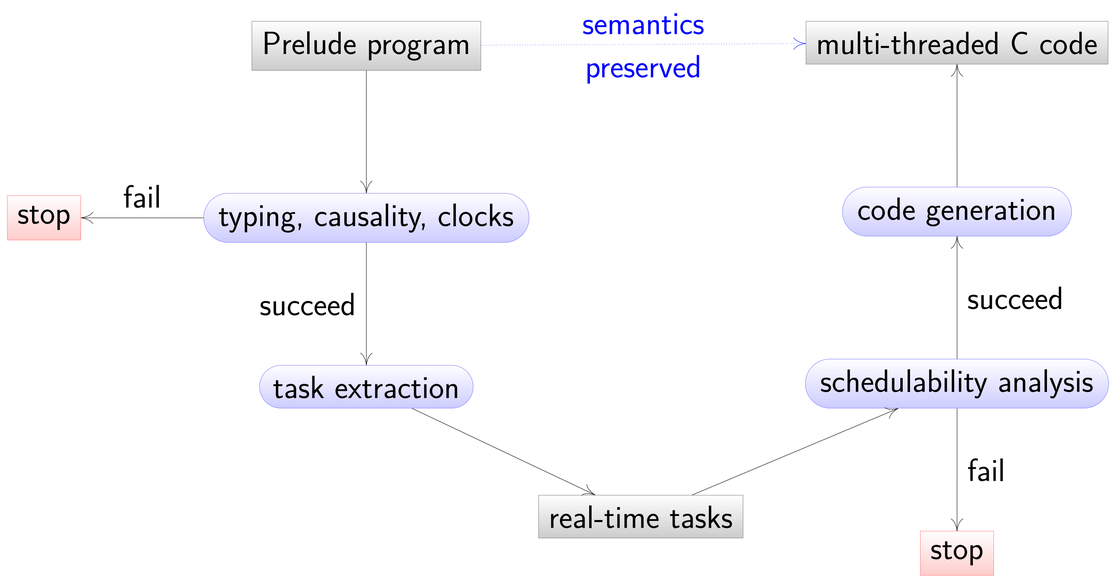
Language overview:

- Functional and temporal Architecture Description Language;
  ⇒ Assemble mono-periodic imported nodes in a multi-periodic program;
- The **clock** (rate) of a flow defines its period and phase;
- **Rate transition** operators (/^, *^) handle multi-rate communications;
- **Relaxed synchronous hypothesis**: complete before next activation of *that* flow (no global instant).

Strictly periodic clocks:

- **Integer clocks**;
- **Arithmetic clock transformations** (rate transitions);
- Easy clock unification;
- Naturally translates to periodicity constraints ⇒ efficient scheduling.

Compiler: generating multi-task C code



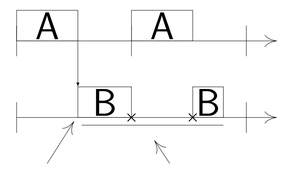
Multi-tasking:

- Each imported node is mapped to a task (≈ thread);
- Generated code is **OS-independent**;
- Available wrappers and schedulers: Linux, SchedMCore, mono/multi-core;
- Easy to extend to another OS (a few days work).

Multi-rate data-dependencies:

1. **Data can only be consumed after being produced**
   ⇒ precedence constraints for the scheduler;
2. **Data must not be overwritten before being consumed**
   ⇒ buffering-based communication protocol.

$A \overset{/\hat{}2}{\hookrightarrow} B$:



(1): B_0 after A_0 (2) keep A_0 available

CRIStAL Centre de Recherche en Informatique, Signal et Automatique de Lille
ONERA THE FRENCH AEROSPACE LAB