Component-based Engineering for Embedded Systems

Code Generation and Simulation from Heterogeneous Models

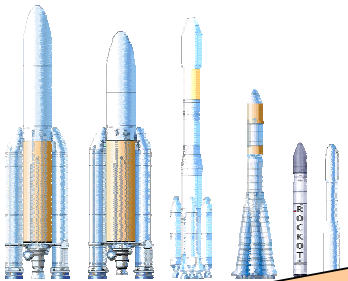
- State of the practice
- State of the art
- Challenges

David LESENS
EADS SPACE Transportation
david.lesens@space.eads.net

EADS SPACE Transportation: Major Products and Capabilities

EADS SPACE TRANSPORTATION

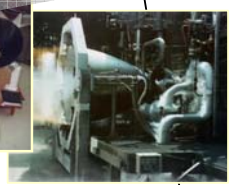
Launchers:
- Ariane
- Soyuz
- Rockot



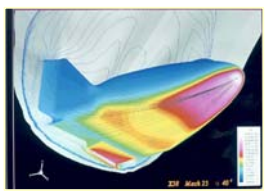
Orbital Systems:
- Columbus
- ATV
- Operations
- Atmospheric Reentry Syst

Real time critical embedded systems

Ballistic



System Design



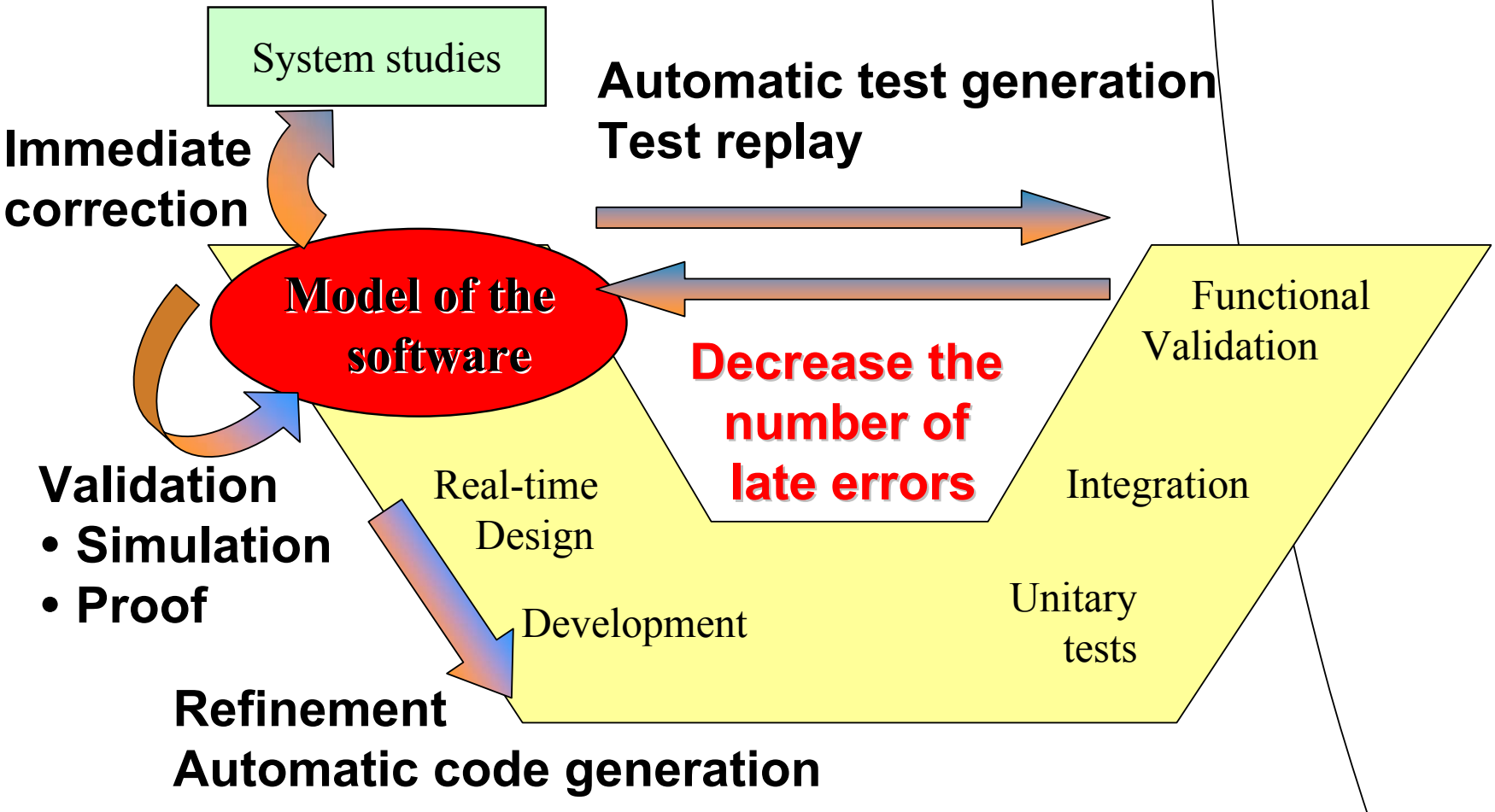
Future Launchers

System Integration & Production

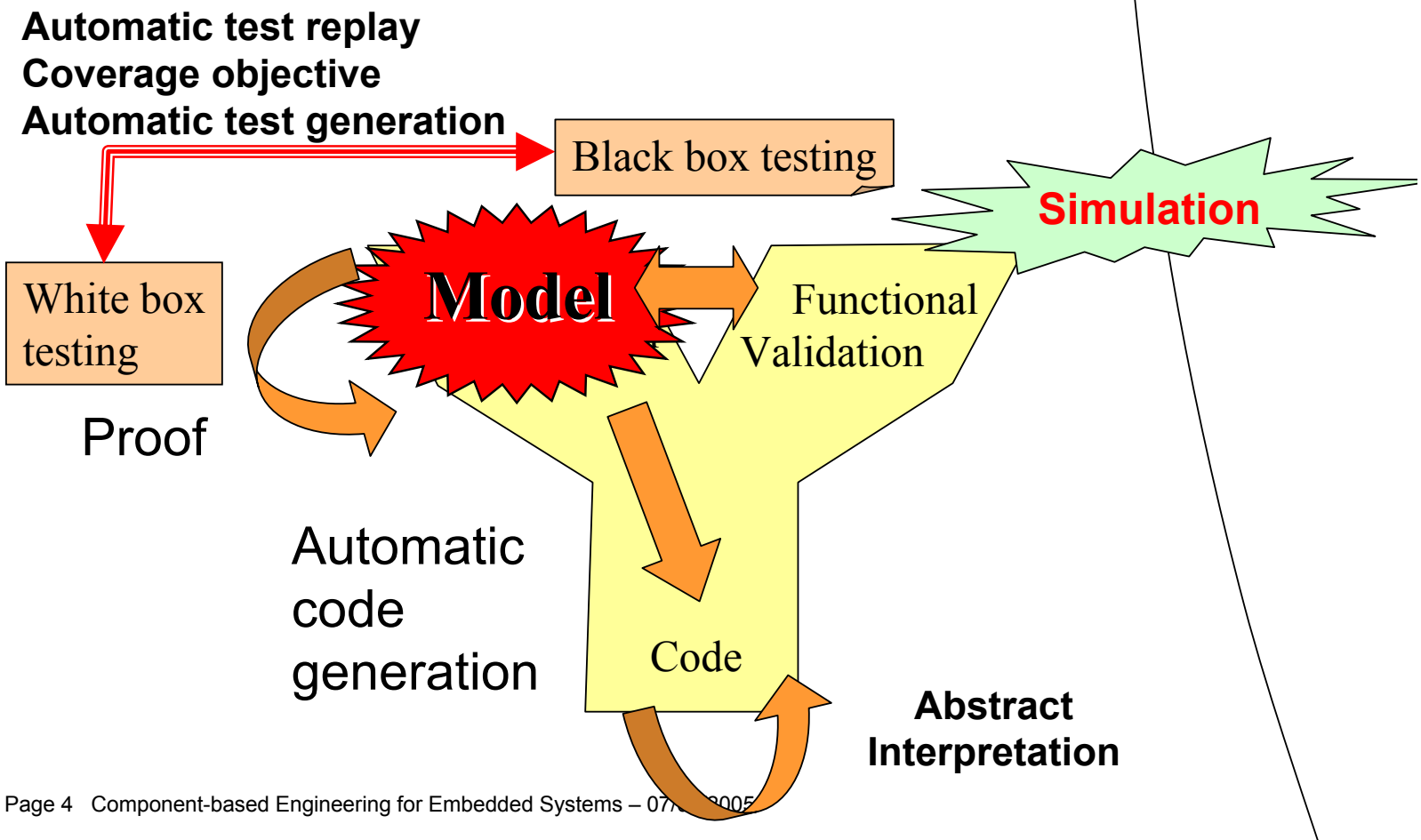
Overview

- Introduction: Why shall we model software components ?
- Why shall we consider heterogeneous systems ?
- How shall we model software components ?
- Scheduling policy
- Conclusion

Why shall we use a model ?



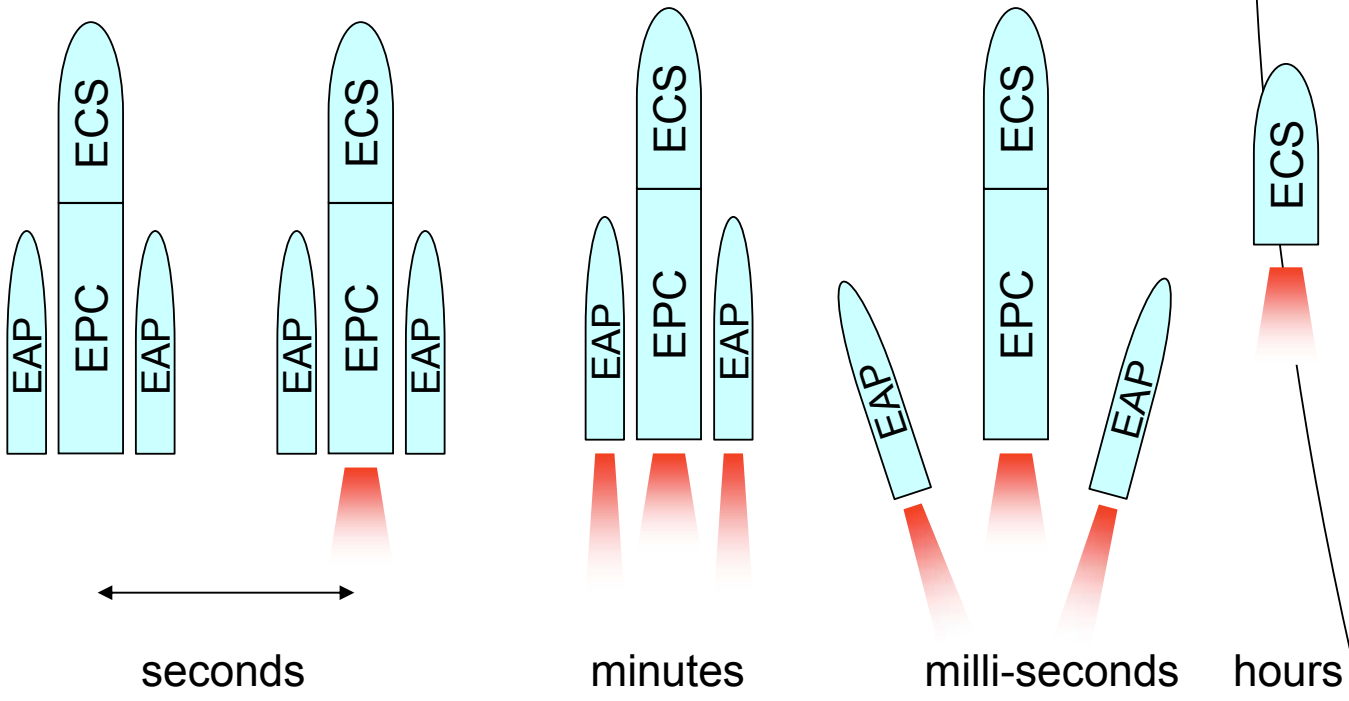
Software development with Automatic Code Generation



Overview

- Introduction: Why shall we model software components ?
- Why shall we consider heterogeneous systems ?
  - Event driven components
  - Control / command (algorithmic) components
  - Control / command (non algorithmic) components
- How shall we model software components ?
- Scheduling policy
- Conclusion

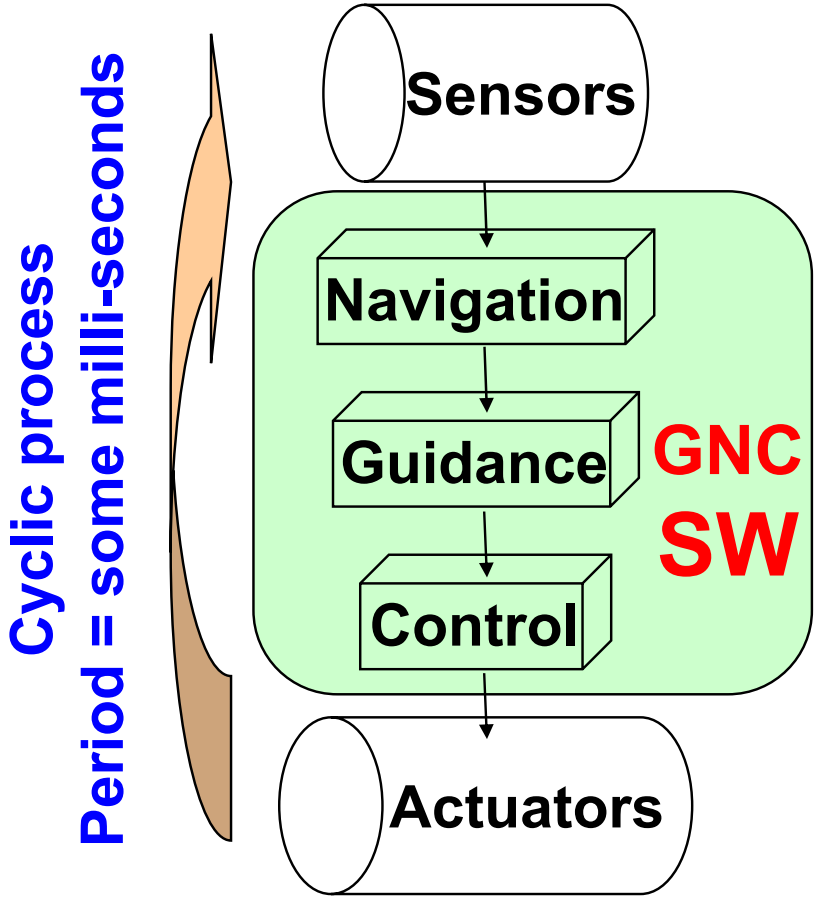
Spacecraft management / mission phases



Spacecraft management deals with timed sporadic events

⇨ Use of timed asynchronous semantics

Spacecraft control / command



- Acquisition of measurement
- Where am I ?
- Where shall I go ?
- Compute the commands
- Send commands to actuators

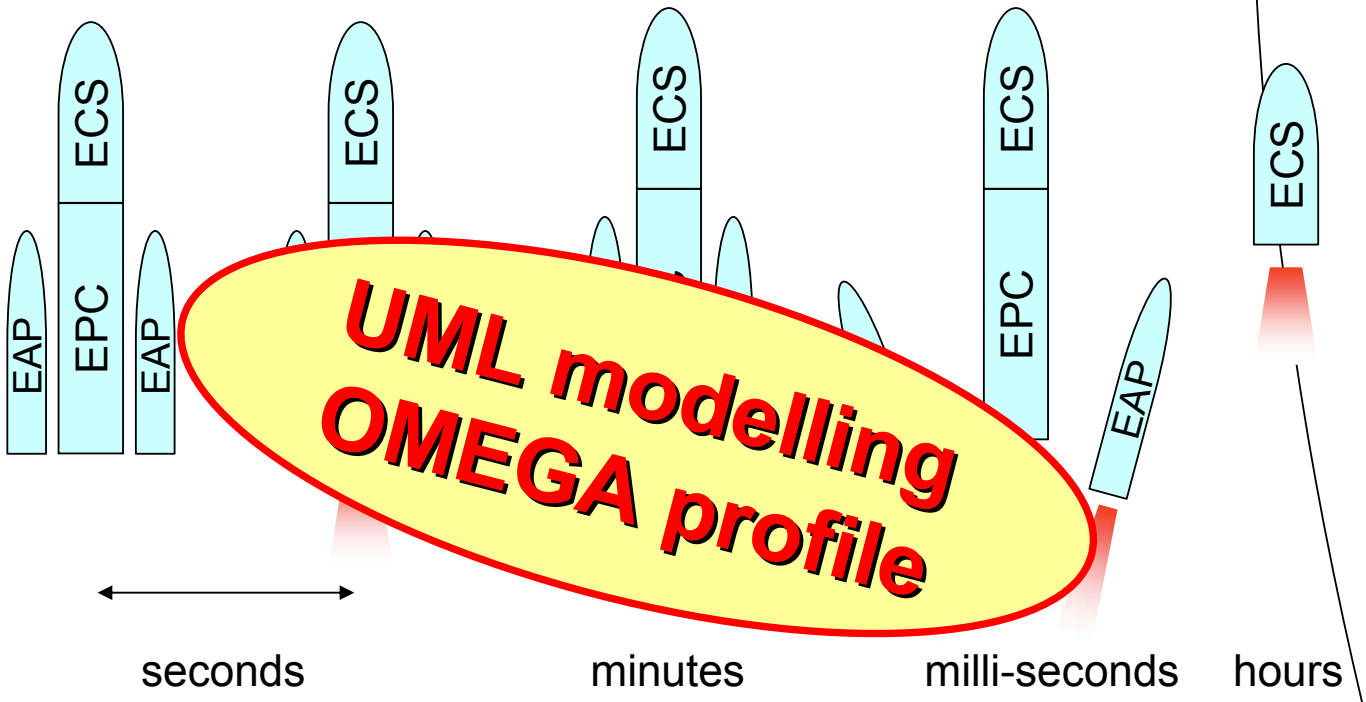
Spacecraft control command

Globally Asynchronous / Locally Synchronous (GALS)

Overview

- Introduction: Why shall we model software components ?
- Why shall we consider heterogeneous systems ?
- How shall we model software components ?
  - State of the practice
  - State of the art
    - **OMEGA** project **2002 - 2005**
  - Challenges
    - **ASSERT** project **2004 - 2007**
    - **ACG** (Automatic Code Generation) project **2003 - 2006**
- Scheduling policy
- Conclusion

Spacecraft management / mission phases



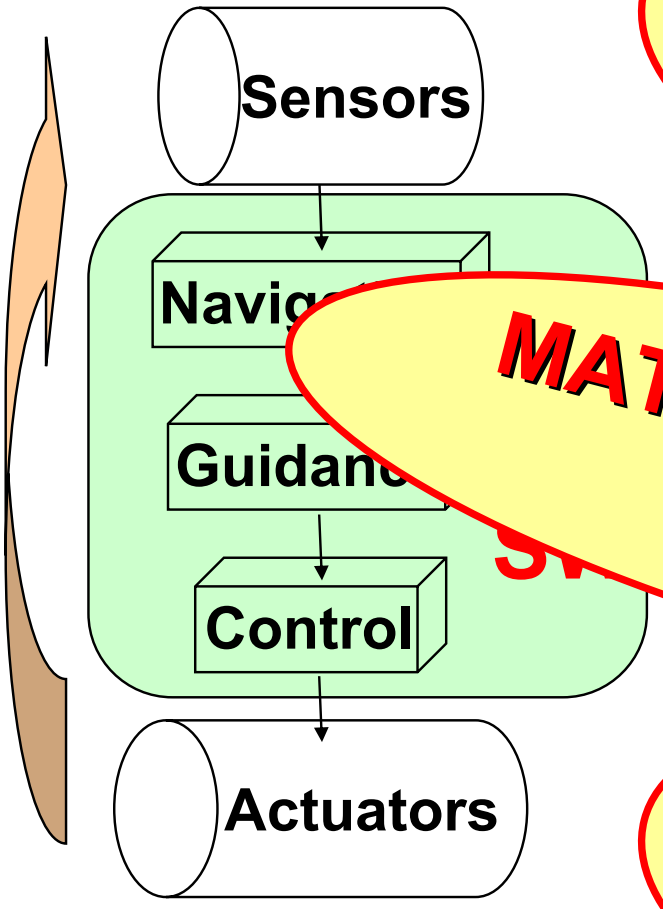
Spacecraft management deals with timed sporadic events

⇨ Use of timed asynchronous semantics

Spacecraft control / comman

Cyclic process

Period = some milli-seconds



SCADE modelling

surement

Where am I ?

MATLAB / SIMULINK modelling

• Co

nds to actuators

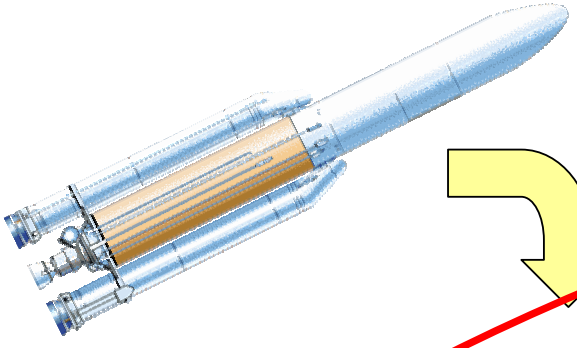
SCADE modelling

Spacecraft cont

Globally Asynchronous / Locally (GALS)

Modelling of control low

EADS SPACE TRANSPORTATION

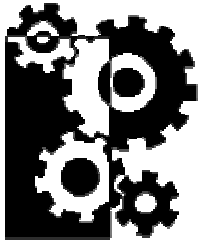


State of the practice

Algorithms feasibility

Simulation

GNC SIMULINK model



Environment SIMULINK model

Algorithm development

Rapid prototyping

2006 / 2007

Automatic code generation

Processor

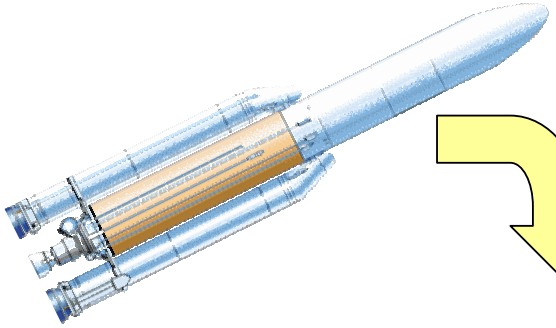
ACG

February 2006

Modelling of a synchronous/asynchronous SW Vehicle management

EADS SPACE TRANSPORTATION

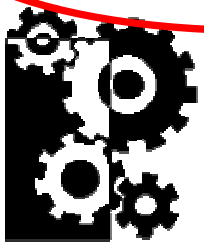
State of the practice



Software specification verification

Simulation & proof

Software SCADE / UML model Ω



Environment SCADE / UML model Ω

State of the art

Software specification

Processor

Automatic code generation

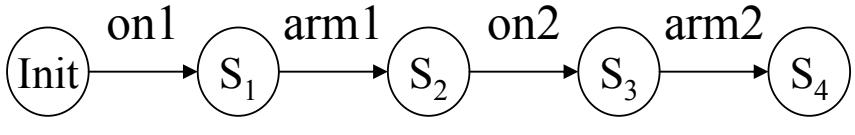
ACG

February 2006

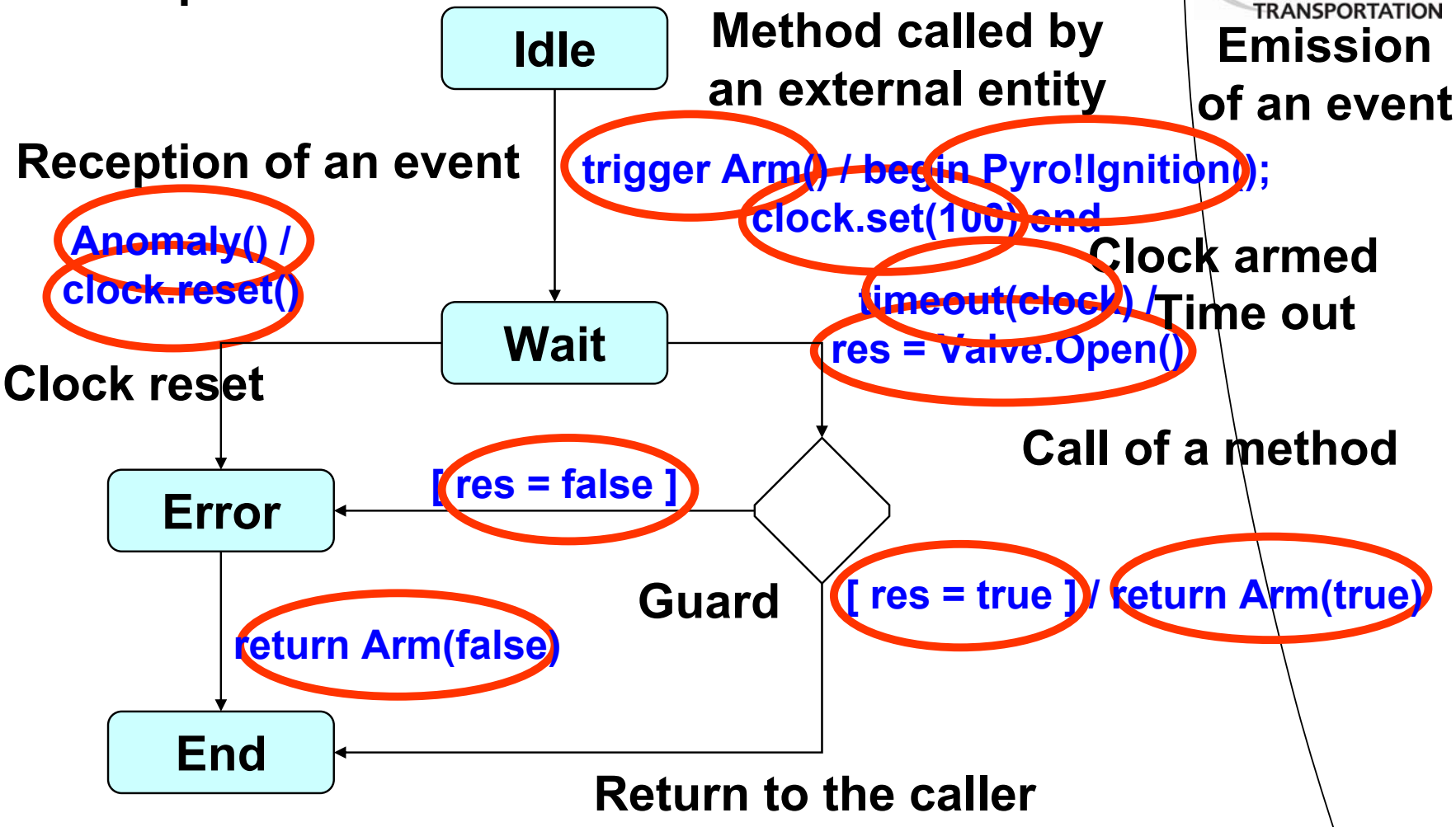
Example of proved properties with SCADE
⇨ Simulation + Proof by model checking

- A CAM test can only be triggered by a “red button” signal
  - true_after_false(CAM_TEST_TRIG) ⇒ RED_BUTTON
  - No assertion is required from the environment to satisfy this property

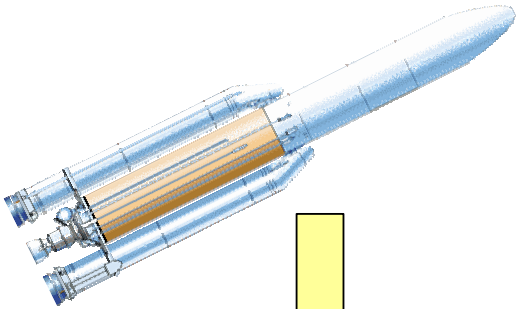
- When the initialisation of the two MSU chains is correct, they can not triggered both a CAM at the same time
  - #(MSU1_CAM_TRIG , MSU2_CAM_TRIG)
  - It is satisfied only when the initialisation of the 2 MSU is correct
  - cam_arm(SWITCH_ON_MSU1, ARM_MSU1, SWITCH_ON_MSU2, ARM_MSU2, RED_BUTTON)



Example of timed automaton in OMEGA



Complete validation



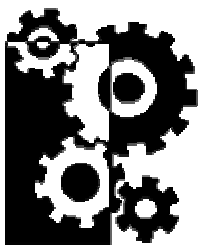
Interconnection between models / platforms

ASSERT

2007

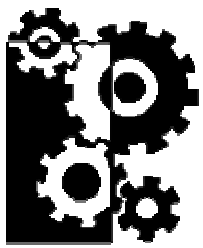
Simulation

Software SCADE / UML model



GNC SIMULINK model

Simulation



Environment SCADE / UML / SIMULINK model

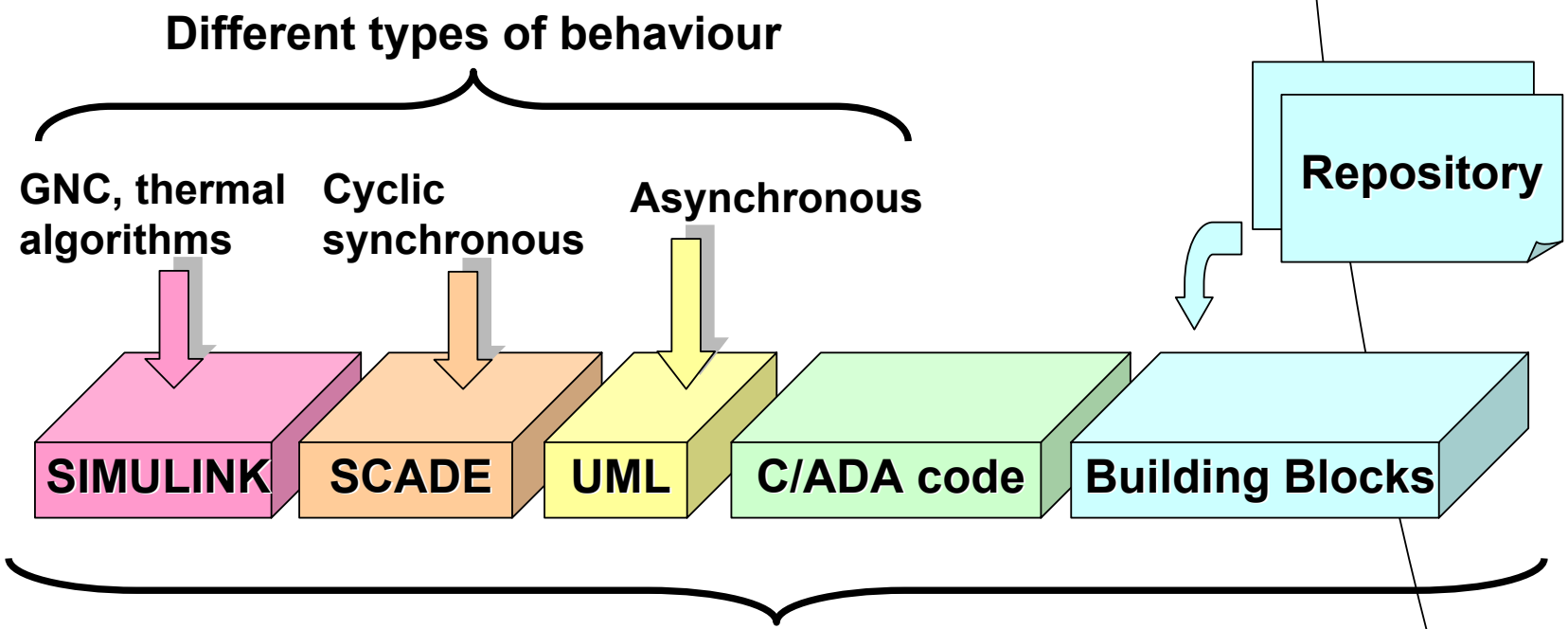
Automatic code generation

Flight code

Rapid prototyping

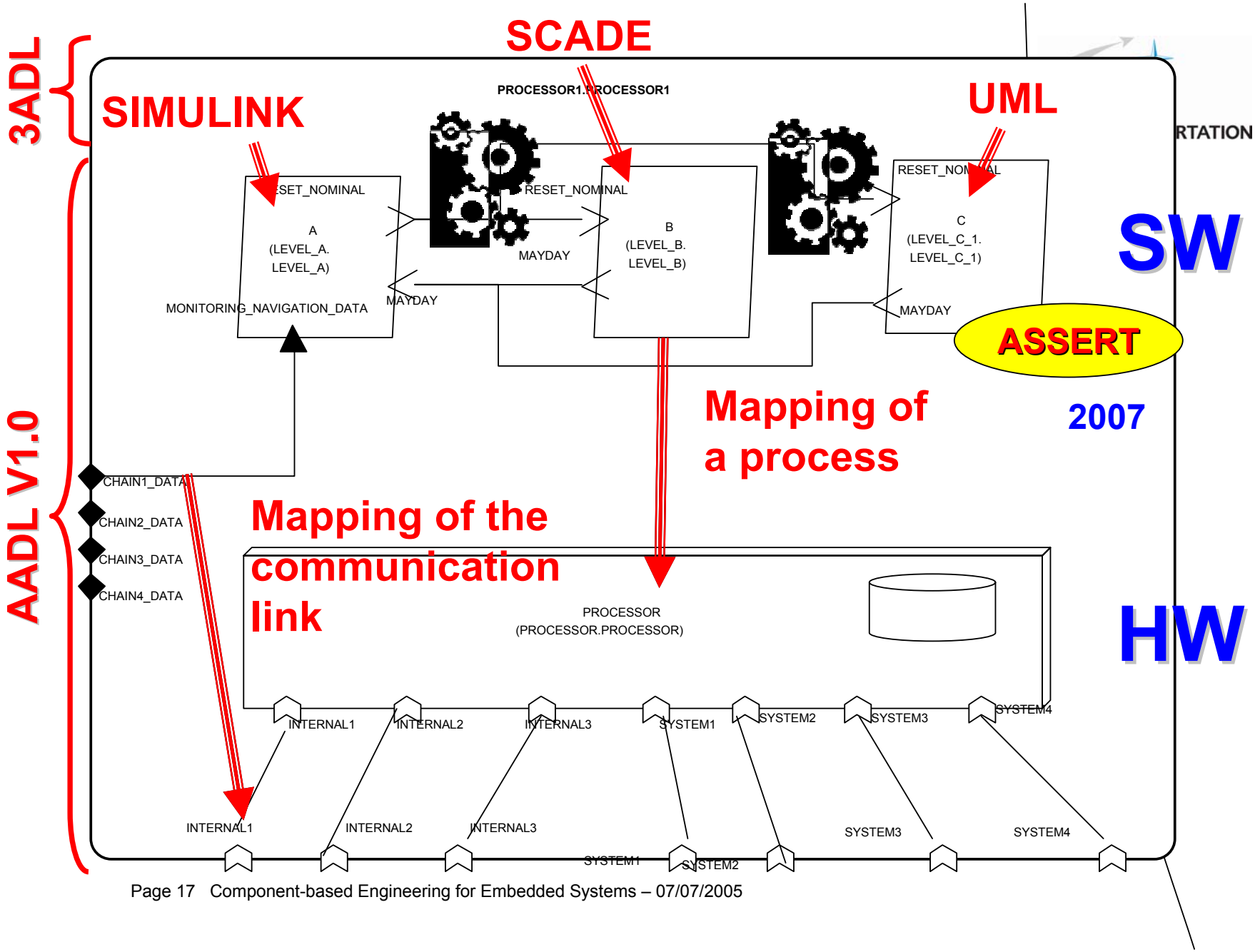
Feasibility

ASSERT project: 2004 - 2007



How to make the glue (static and dynamic) ?

⇨ **Use of an AADL modelling + proof**


3ADL
SCADE
SIMULINK
UML
PROCESSOR1.PROCESSOR1
RESET_NOMINAL
A
(LEVEL_A.
LEVEL_A)
MONITORING_NAVIGATION_DATA
MAYDAY
RESET_NOMINAL
B
(LEVEL_B.
LEVEL_B)
MAYDAY
RESET_NOMINAL
C
(LEVEL_C_1.
LEVEL_C_1)
MAYDAY
SW
ASSERT
2007
Mapping of
a process
AADL V1.0
CHAIN1_DATA
CHAIN2_DATA
CHAIN3_DATA
CHAIN4_DATA
Mapping of the
communication
link
PROCESSOR
(PROCESSOR.PROCESSOR)
HW
INTERNAL1
INTERNAL2
INTERNAL3
SYSTEM1
SYSTEM2
SYSTEM3
SYSTEM4
INTERNAL1
INTERNAL2
INTERNAL3
SYSTEM1
SYSTEM2
SYSTEM3
SYSTEM4

Overview

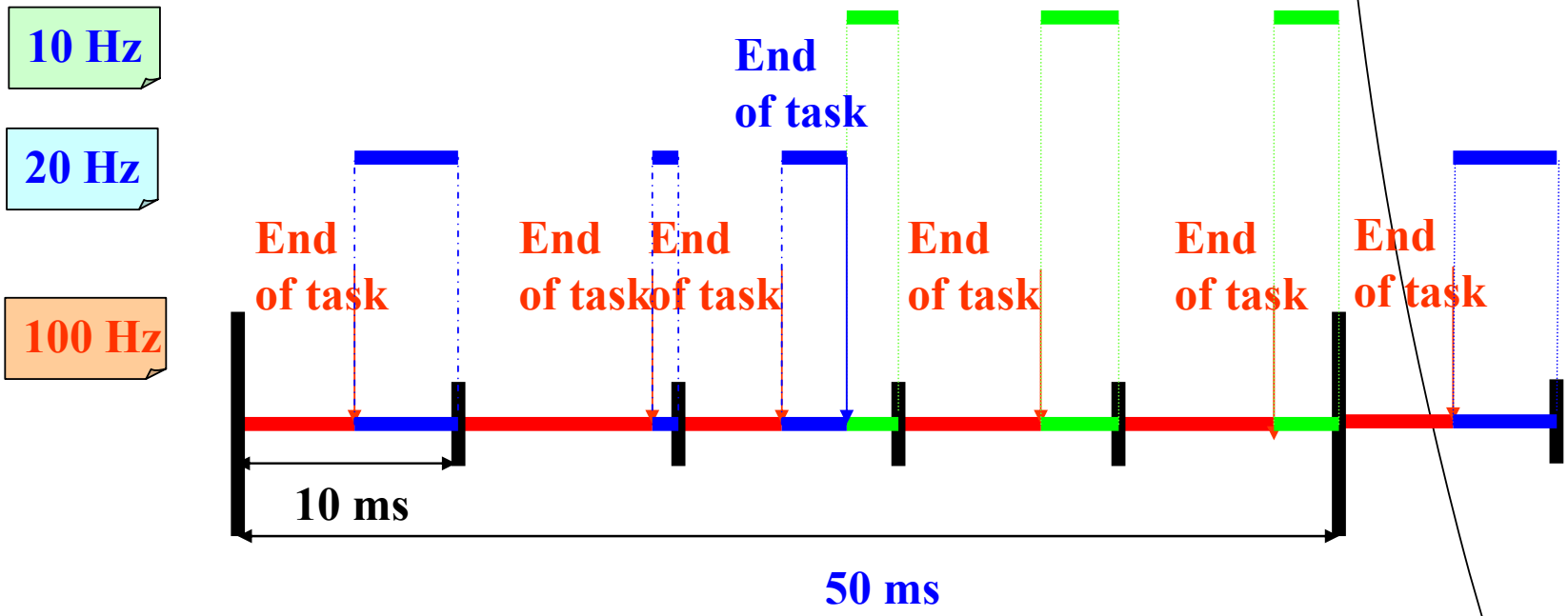
- Introduction: Why shall we model software components ?
- Why shall we consider heterogeneous systems ?
- How shall we model software components ?
- Scheduling policy
  - Mixing synchronous / asynchronous behaviour
  - Maintainability
- Conclusion

Multitasking for Critical Software

(Current state of the practice)

- Multitasking preemptive with fixed priority
- Rate Monotonic Analysis (RMA)

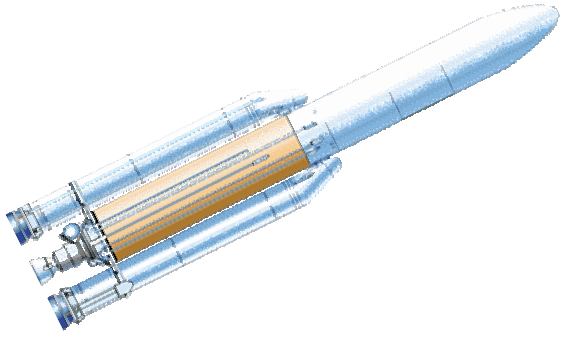
Deterministic



Well adapted for <u>Control / Command</u> (as a launcher)

But what about a <u>more complex system</u> (as a spacecraft)

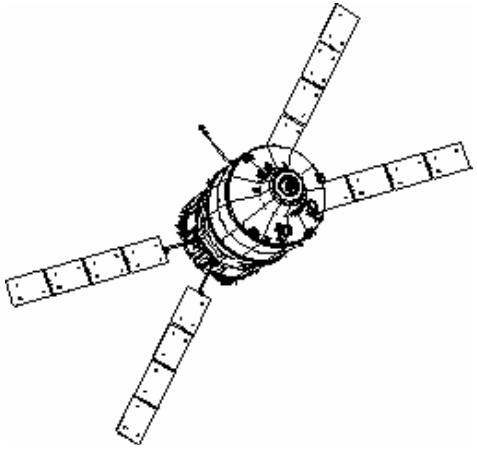
Critical Real Time Embedded Software



~ 20 states
Flight duration ~ 1 hour

Dynamic creation/deletion of
- objects
- processes

forbidden



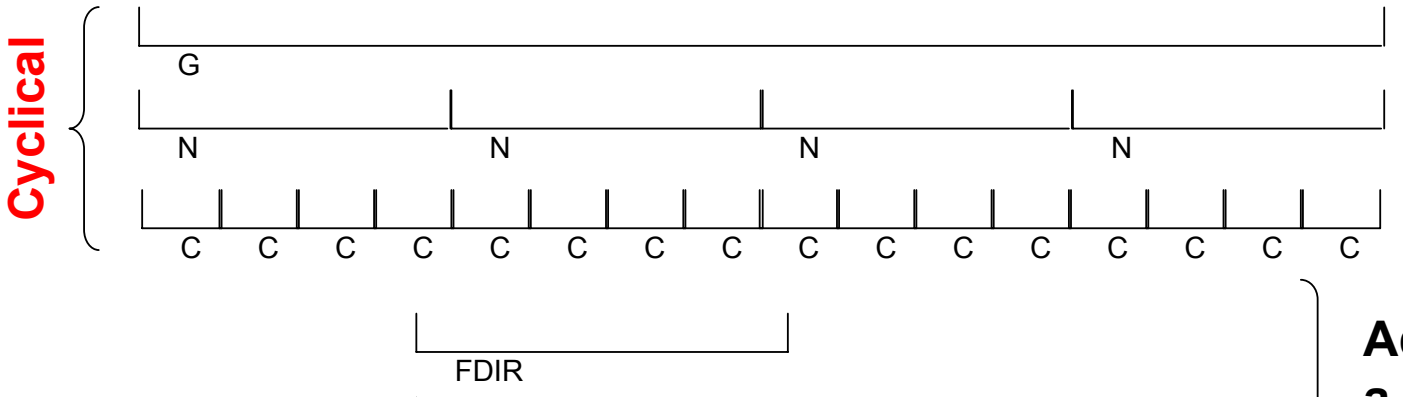
> 10000 states
Flight duration > 3 months

✓ Complex asynchronous system
✓ Need of "during the flight" maintenance

RMA to EDF

EADS SPACE TRANSPORTATION

Guidance, Navigation Control



Addition of a <u>new</u> component

Acyclic & asynchronous events

Rate Monotonic Analysis (RMA) ⟶ Early Deadline First (EDF)

But what happens in case of missing deadline?

EDF + Criticality Level

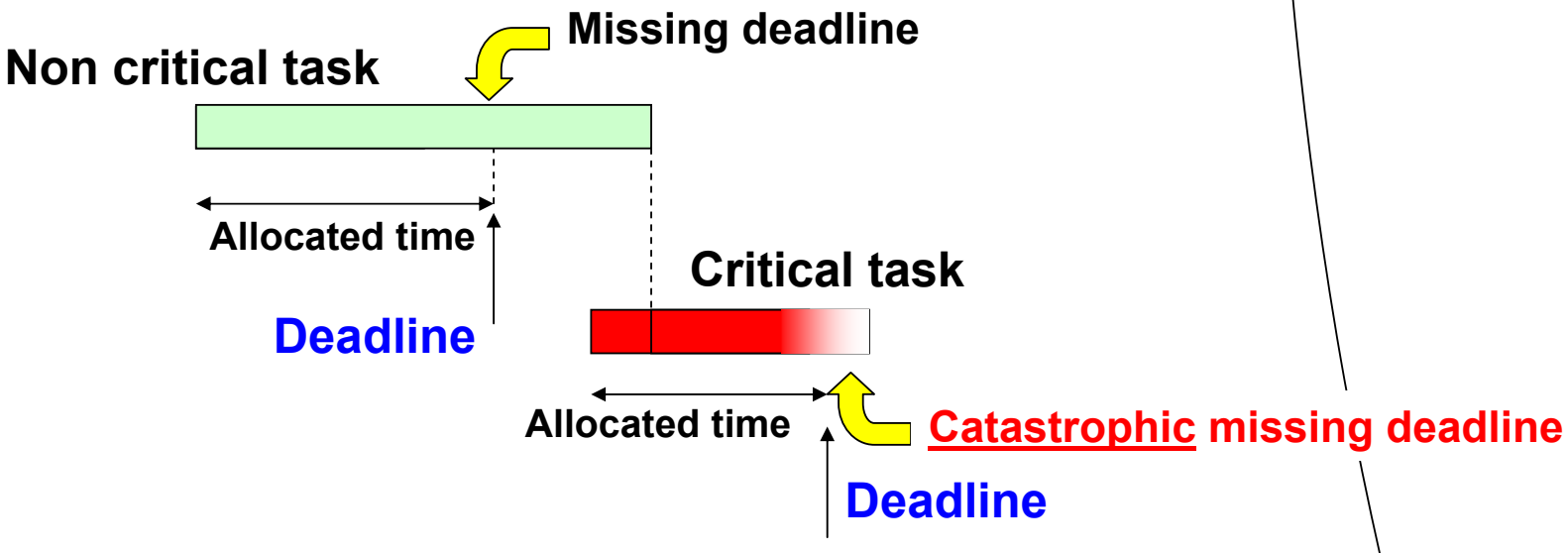
EADS SPACE TRANSPORTATION

"Criticality" level

- Cyclic missing deadline
  - For thermal control ⇨ No impact on the mission → **4**
  - For Guidance ⇨ Lost of precision → **3**
  - For Control ⇨ Destruction of the spacecraft → **2**
- Acyclic missing deadline
  - Telecommand ⇨ Lost of precision → **3**
  - Post contact boost ⇨ Destruction of the station → **1**

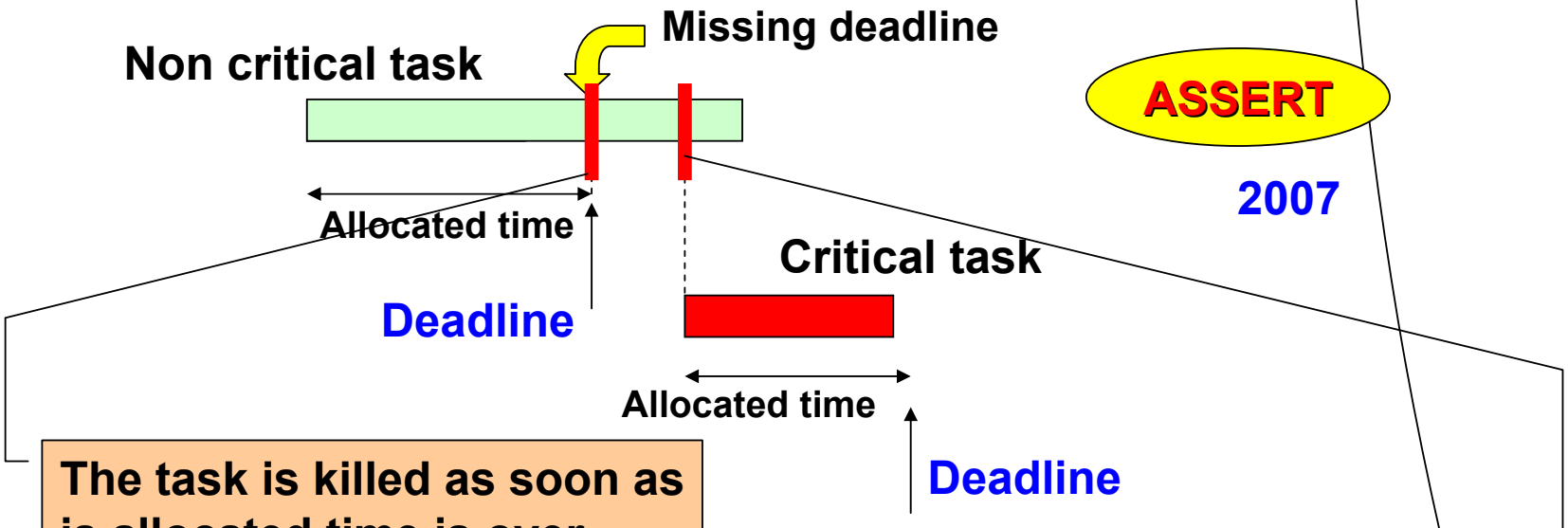
⇨ **Take into account the deadline <u>and</u> the criticality**

What can we do when a task does not meet its deadline ?



⇨ **The faulty non critical task has to be killed**
(or at least suspended)

OASIS approach versus ASSERT approach



ASSERT

2007

The task is killed as soon as is allocated time is over

OASIS approach

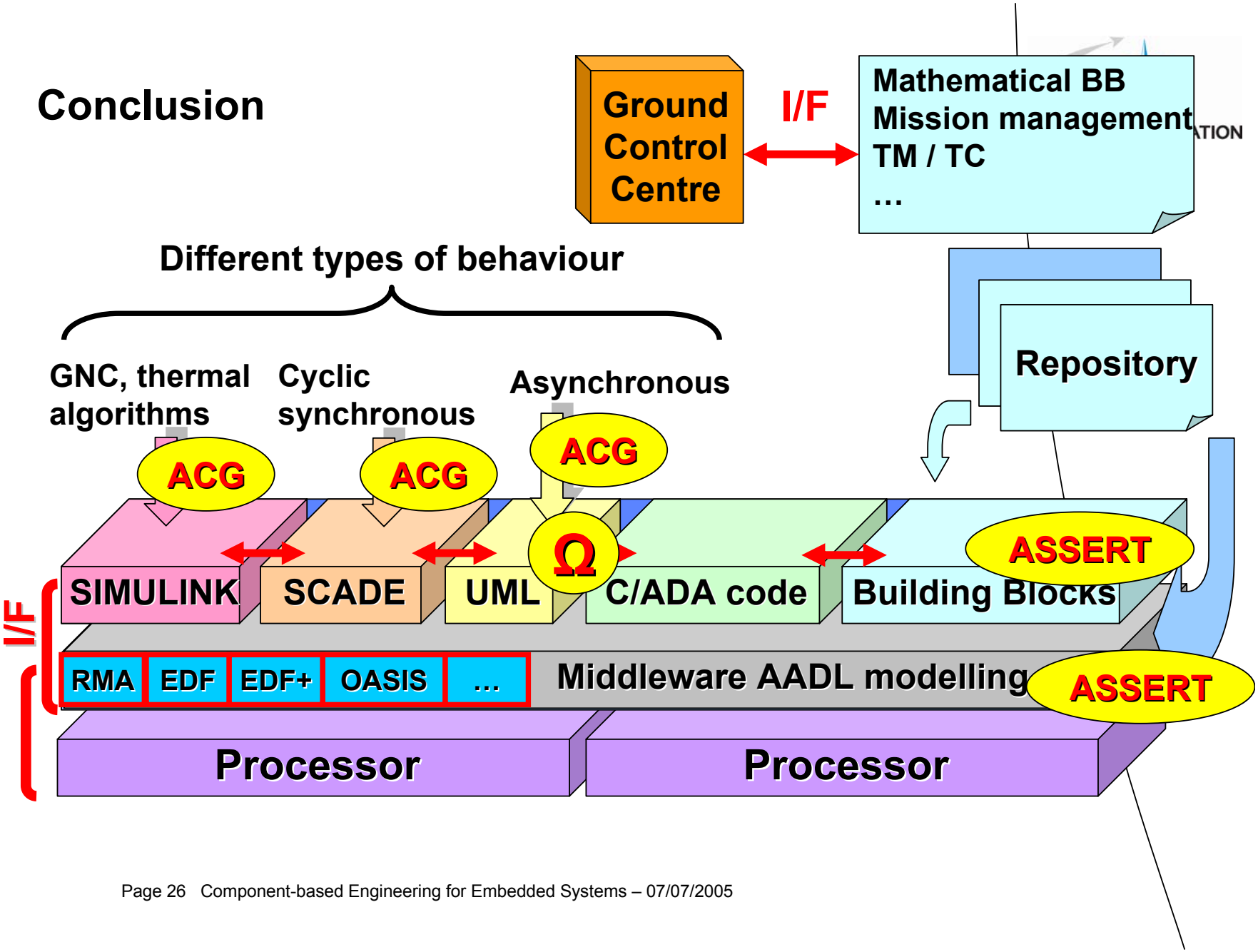
The task is killed as soon as it can avoid another task to meet its deadline

ASSERT approach

Overview

- Introduction: Why shall we model software components ?
- Why shall we consider heterogeneous systems ?
- How shall we model software components ?
- Scheduling policy
- Conclusion

Conclusion


Ground Control Centre
I/F
Mathematical BB
Mission management
TM / TC
…
Repository
Different types of behaviour
GNC, thermal algorithms
Cyclic synchronous
Asynchronous
ACG
ACG
ACG
Ω
ASSERT
SIMULINK
SCADE
UML
C/ADA code
Building Blocks
I/F
RMA
EDF
EDF+
OASIS
…
Middleware AADL modelling
ASSERT
Processor
Processor