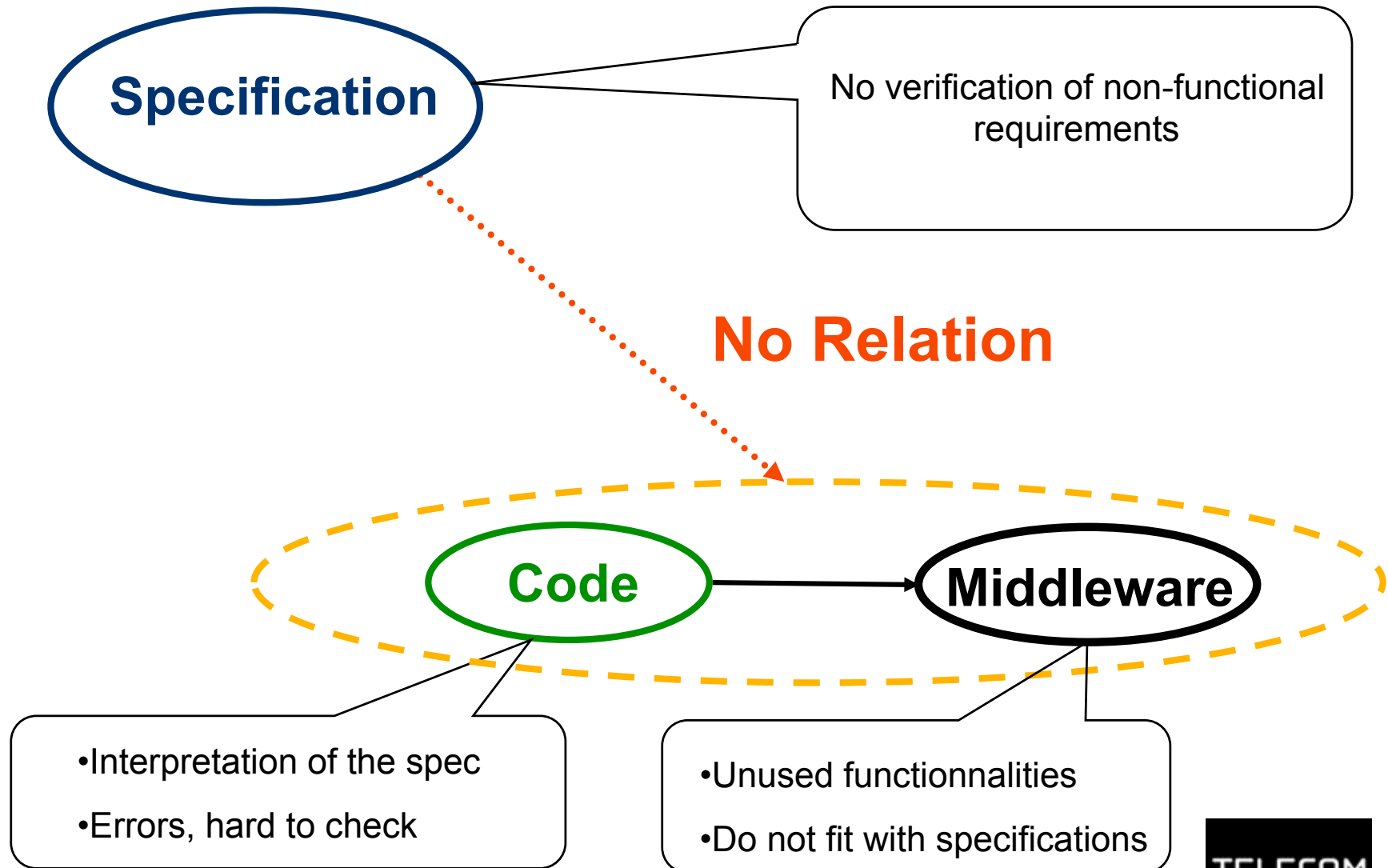


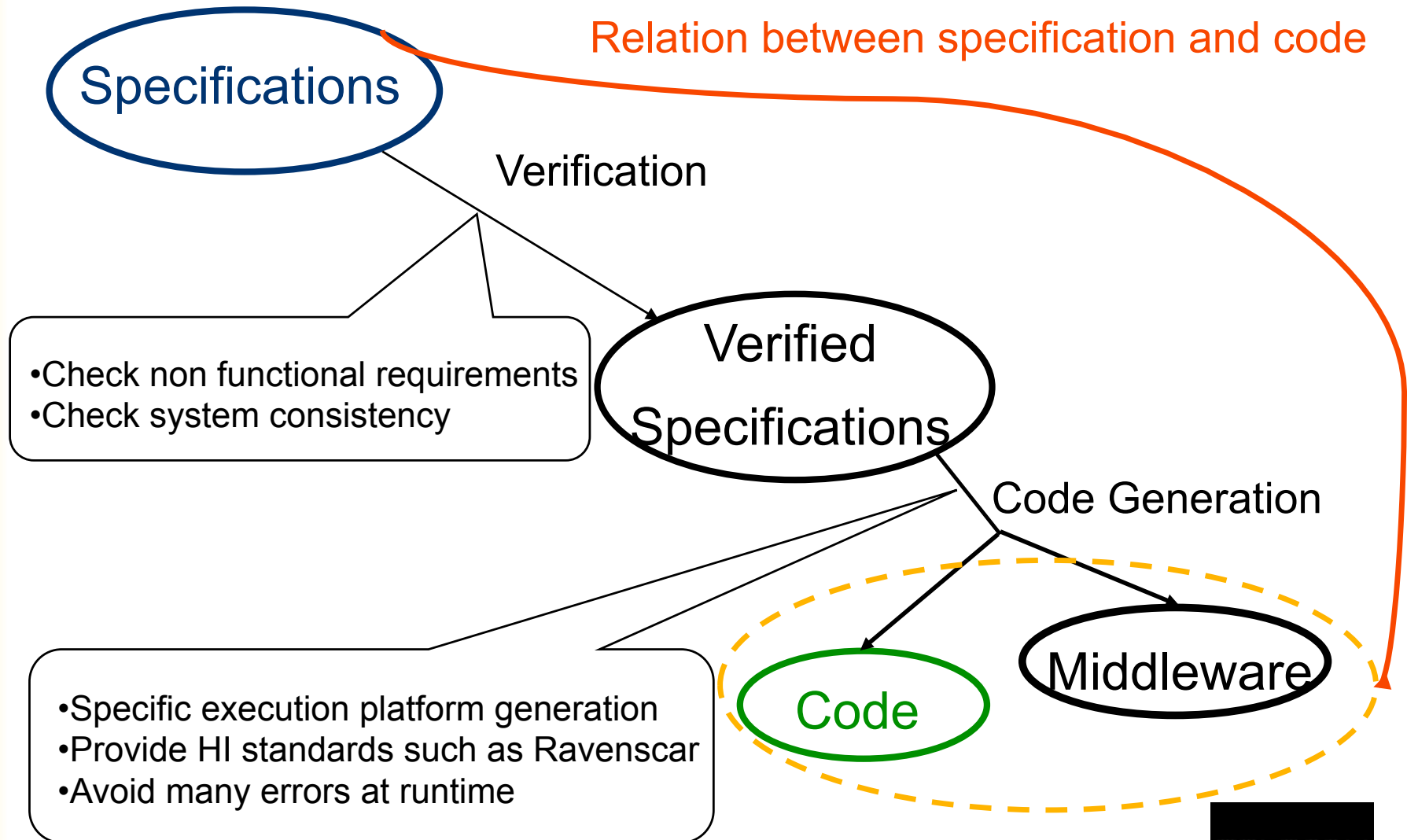
Background

- Distributed Real-Time Embedded (DRE) systems + High Integrity
 - ⇒ High Integrity : life/mission critical
- DRE brings specific constraints
 - ⇒ Deployment and configuration problems
 - ⇒ Small memory footprint, no dead-code
- HI brings dedicated design approaches
 - ⇒ Relation between specifications and the implementation
 - ⇒ Follow standards (DO178B, Ravenscar, etc)

Traditional design of DRE systems



Objectives



Development process

- Relation between specs and code

- Specification verification

- ⇒ System consistency, schedulability analysis

- Automatic code generation

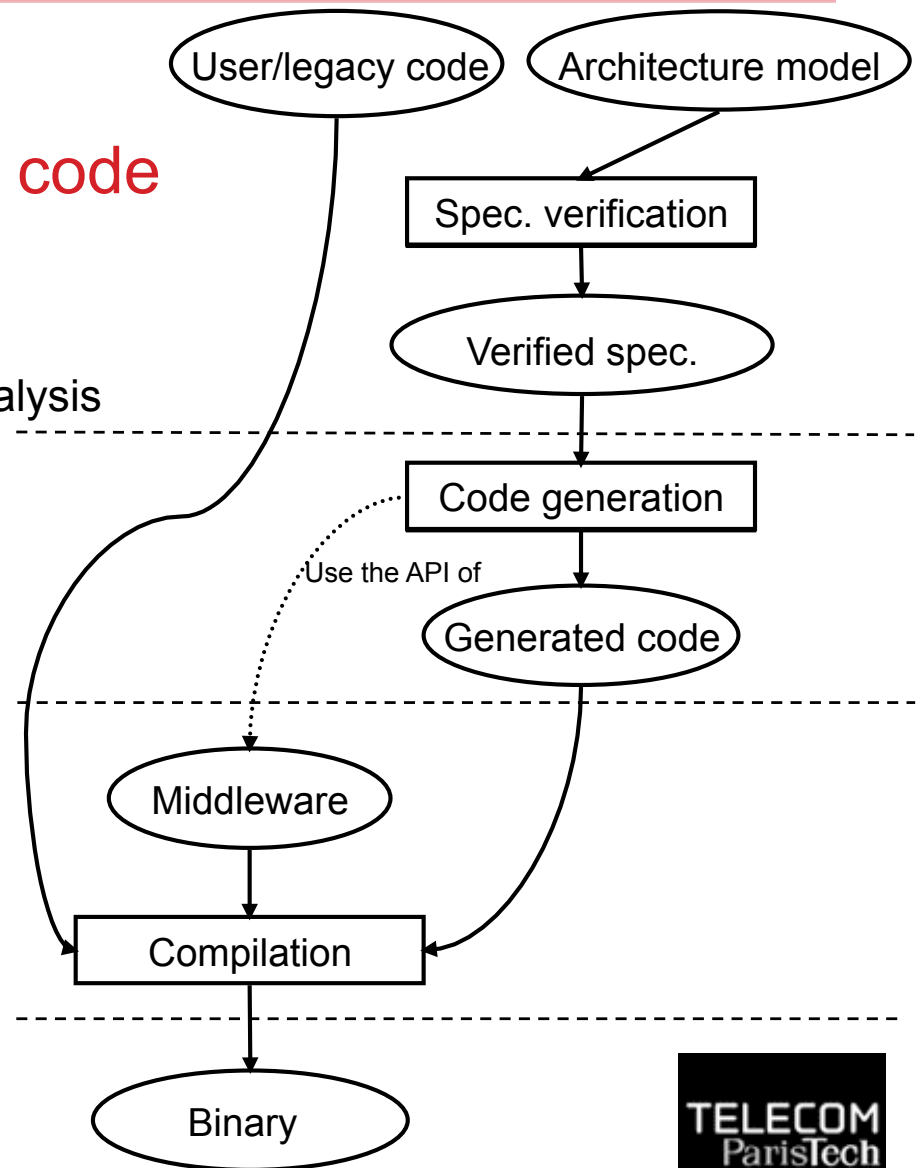
- ⇒ Deterministic process

- ⇒ Ravenscar profile

- Minimal middleware

- ⇒ Generic static services

- ⇒ Model-specific functionalities



Main choices

● Architecture Model

- ⇒ AADL, verifiable, extensible
- ⇒ Use Ravenscar Concurrency Model (RCM)

● Runtime/middleware

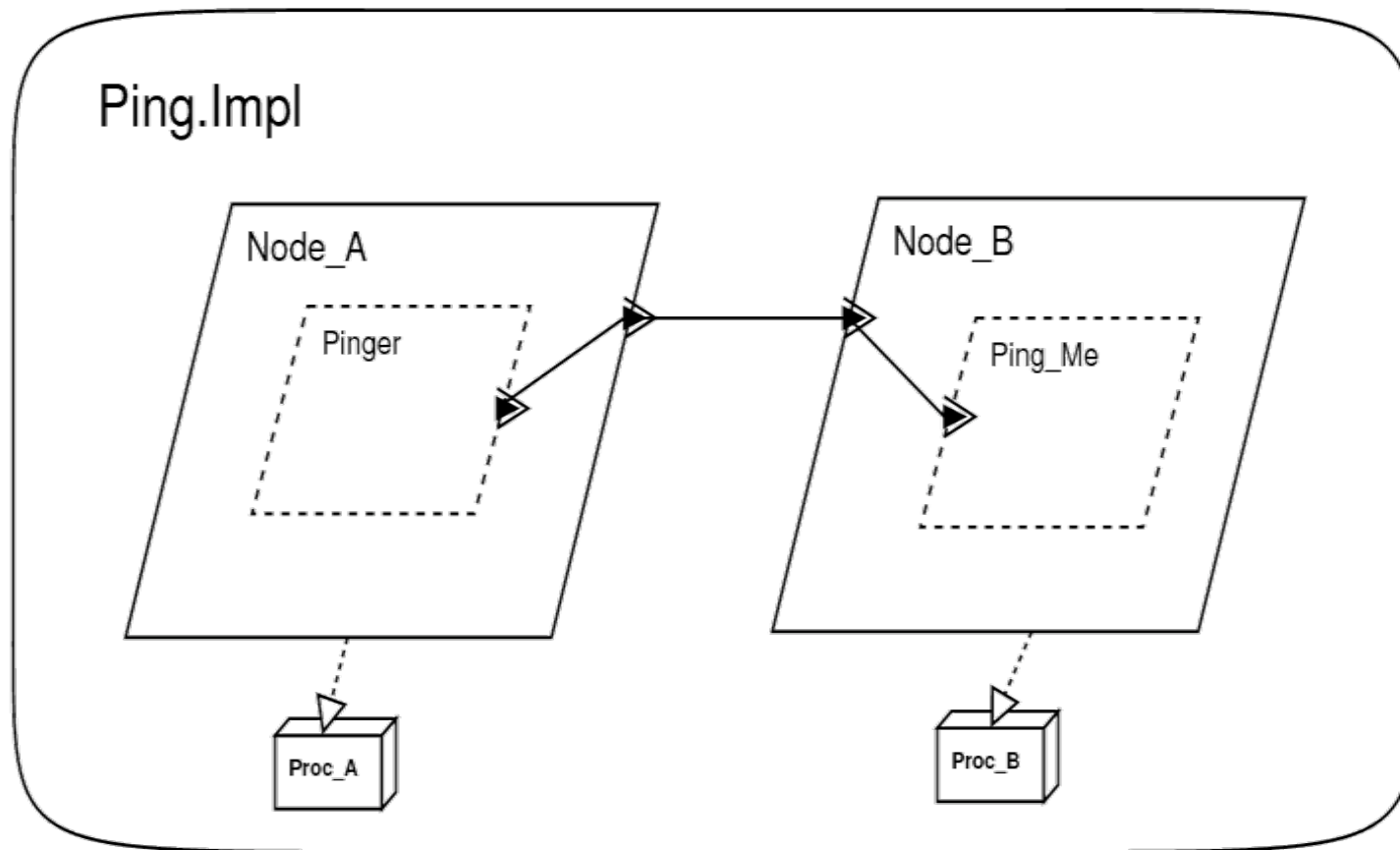
- ⇒ PolyORB-HI designed for Ada and its runtime
- ⇒ Design PolyORB-HI-C, runtime specific to the model

● Language independence

- ⇒ Provide tasks and resources-handling facilities
- ⇒ RCM in Ada : specific Ada code checked against Ravenscar profile by Ada compiler
- ⇒ RCM in C : specific C patterns already compatible with Ravenscar restrictions

Overview of AADL

- Hierarchical components
- Components properties
- Extension
- Verification



Generic process : from AADL models to binary code

- **Analyze and expand model**

 - ⇒ System consistency

- **Generate application code**

 - ⇒ Marshallers, data protection

- **Generate middleware**

 - ⇒ Deployment facilities

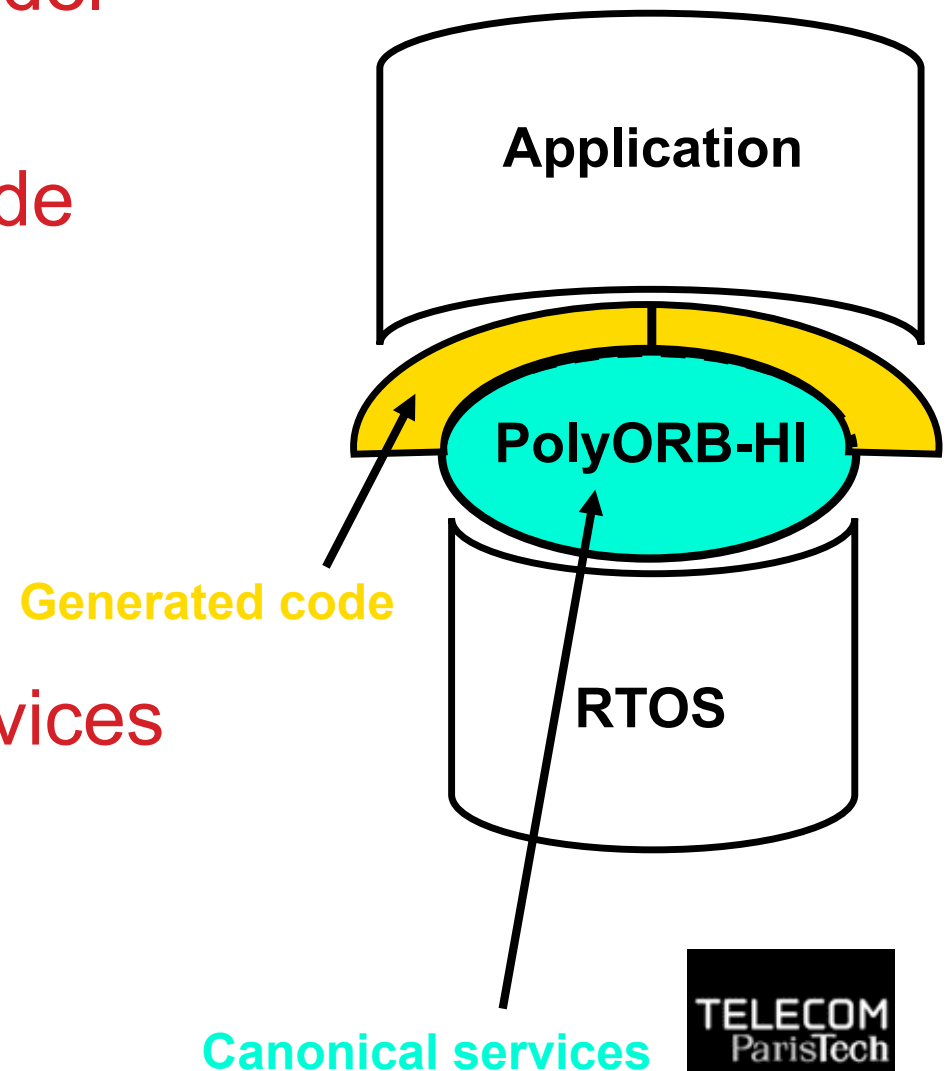
 - ⇒ Resources allocation

- **Assemble canonical services**

 - ⇒ with generated application code

 - ⇒ with generated middleware code

 - ⇒ with application code



Code generation patterns

Process

- Create a directory to store the generated files of this component
- Initialize the system, generate the entry point of the process

Thread

- Instantiate the thread and its connections
- Create a worker function for the thread execution

Subprogram

- Bound to user-defined or generated subprograms

Data

- Declare a new type using the data properties
- Declare new resources to lock/unlock data in case of protected ones

Connection

- Instantiate resources (buffers, listener thread, etc)
- Add function calls to send/receive data

Generating C from AADL Ravenscar model

● Ada code generation

- ⇒ Generate Ada code from AADL models using Ada concurrency constructs
- ⇒ Compile the whole Ada application against Ravenscar restrictions
- ⇒ Expand the Ada concurrency constructs into calls of the Ada Ravenscar runtime
- ⇒ Assemble Ada Ravenscar runtime, kernel and minimal middleware

● C code generation

- ⇒ Generate C code from AADL models using C patterns (Ravenscar compatible)
- ⇒ Expand the AADL concurrency constructs into kernel constructs
- ⇒ Compile the whole C application (but user code may not be Ravenscar compatible)
- ⇒ Assemble kernel and minimal middleware

Minimal middleware for the generated code

● PolyORB

- ⇒ First middleware formally verified in 2005
- ⇒ Open source project, support by Adacore, used in industry

● PolyORB-HI

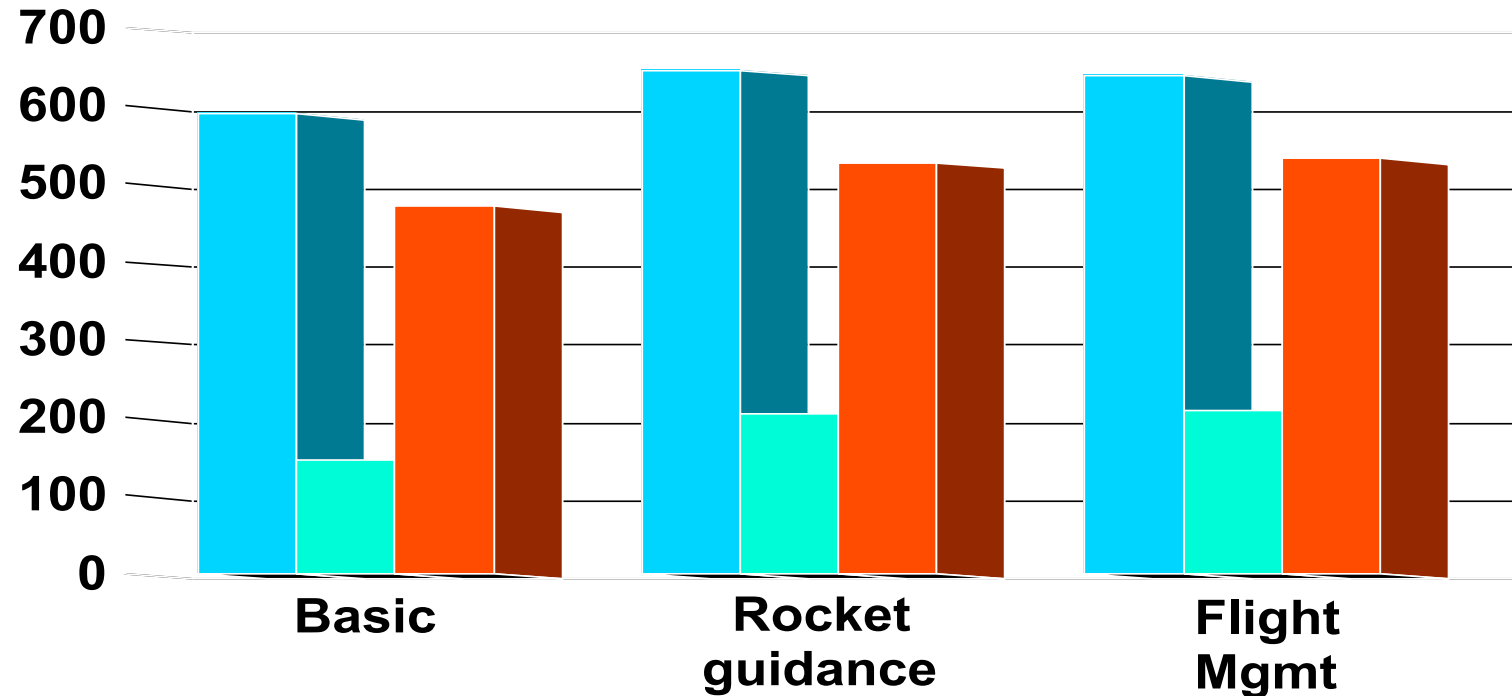
- ⇒ Designed in the ASSERT European Project
- ⇒ Services inherited from PolyORB, ignore irrelevant services in the HI context
- ⇒ Previously designed with Ada, now available for C

● Differences between Ada and C version

- ⇒ Ada version uses the Ada runtime to create tasks, resources and protect data
- ⇒ C version define model-specific runtime
- ⇒ Rely on code generation patterns including Ravenscar and concurrency one

Tests and benchmarks

Memory footprint (in KB) of produced binaries



- Low memory footprint
- Many platforms

■ Nokia N770
■ Nintendo DS
■ Native



Conclusions and Perspectives

- Modeling help us to design HI systems
- Interesting facts
 - ⇒ Code generation compliant with HI requirements
 - ⇒ Ravenscar approach applied at AADL level and implemented using C patterns generation
- Perspectives
 - ⇒ Minimal RTOS
 - ⇒ Safety/security requirements
 - ⇒ Code generation for other languages

Web resources

- <http://aadl.enst.fr>
 - ⇒ AADL portal at TELECOM ParisTech

- <http://aadl.enst.fr/ocarina/>
 - ⇒ Ocarina AADL toolkit website

- <http://aadl.enst.fr/polyorb-hi/>
 - ⇒ PolyORB-HI website



Questions ?

Thanks for listening