ARINC653, AADL annex

Laurent Pautet, Télécom ParisTech
Laurent.Pautet@telecom-paristech.fr

Julien Delange, Télécom ParisTech
Julien.Delange@telecom-paristech.fr
ARINC653
- Avionics standard
- Standardized API (called APEX – APplication Executive)
- Central part of the IMA philosophy
- Time & space partitioning

Rationale of ARINC653 annex for AADLv2
- Standardized modeling patterns
- Better modeling & analysis support
- Code generation from AADL to ARINC653 O/S
ARINC653 standard overview

- **Partitioning support**
  - Software isolated in partitions
  - Partitions run as if they were on a single processor

- **Time isolation**
  - Execution during a fixed & predefined time slice
  - Tasks scheduled with a dedicated scheduling policy

- **Space isolation**
  - Code & data stored in a separated address space

- **Fault containment**
  - Faults are propagated from processor to partitions
  - Partition-dependent recovery strategy
ARINC653 services

- **Time and space isolation**
  - Time slices allocation
  - Address spaces allocation

- **Tasking (process) services**
  - Similar to the thread concept

- **Communication services**
  - Intra-partition
  - Inter-partitions (module enforced)

- **Health Monitoring**
  - Recover faults at module, partition or process levels
Map ARINC653 services to AADL models

- **Partitioning support**
  - Partition execution context: virtual processor
  - Partition content: process

- **Partitions control (with time & space specification)**
  - Support for partitions execution: processor

- **Tasking/process service**
  - Thread component

- **Communication services**
  - Rely on ports connections

- **Health Monitoring**
  - Dedicated properties (ARINC653 property set)
Map ARINC653 services to AADL models

- **Partitioning support**
  - Partition execution context: virtual processor
  - Partition content: process

- **Partitions control (with time & space specification)**
  - Support for partitions execution: processor

- **Tasking/process service**
  - Thread component

- **Communication services**
  - Rely on ports connections

- **Health Monitoring**
  - Dedicated properties (ARINC653 property set)
From ARINC 653 architecture to AADL models

Partition 1 (virtual processor) -> Module (processor) -> Process 1 (thread) -> Segment 1 (memory)
Current standardization state

- Joint effort from academic and industrial partners
  - Show the relevance of the annex for the industry
  - Support from academic tools (e.g. Ocarina with ARINC653 code generation)

- Presented for ballot at the next meeting (May 2010)

- Publication as a standardized annex for end-2010
  - White paper to illustrate modeling patterns usage
  - Examples and case studies coming with the annex
Conclusion

- **Standardized modeling patterns**
  - Mapping ARINC653 services
  - Enforce AADL semantics

- **Ease design, analysis and implementation**
  - Benefits from AADL validation tools
  - Code generators already available

- **Publication in late 2010**
  - Ready for the ballot process
  - Standardization as an annex document

- **First implementation in Ocarina (compiler) + POK (run-time)**
  - [http://pok.gunnm.org](http://pok.gunnm.org)