Thales case study
Hi-Lite — 13-July-2010
Thomas Vergnaud
MyCCM stands for “Make Your Component Container Model”

- Based on lwCCM standard with specific extensions
  - Open container to host non functional services
  - Connectors to support several interaction semantics
- Fully based on separation of concern realized by “Component / Container / Connector” approach
- Can be integrated into UML modelers with lwCCM + extensions profiles
- Customization for domain specific needs
  - Code generation framework
  - Adaptation of code generation to execution platforms
    - Languages (C, C++, Java, Ada)
    - Operating Systems (Linux, VxWorks, OSE-ck, OSEK)
    - Middleware (ORBs discrepancies, specific communication middleware)
- Interaction connectors definition and implementation
- Technical services definition and implementation
Structure of the MyCCM framework

- **several layers**
  - isolate the application code from the runtime
  - manage communications and execution
Structure of the MyCCM layers

- **full control on the MyCCM code**
  - envelope and deployment code is generated
  - runtime adaptation layer is static code
- **no control on application code**
- **no control on runtime code**
- **two generators**

  - data types
  - component interfaces
  - threads and processes
  - component allocation

  - envelope generator
    - high level primitives
    - runtime adaptation layer
  - deployment generator
    - envelope code
    - deployment code
    - envelope code
    - deployment code
formal approaches and MyCCM

- compute formal specifications of interfaces from component connections
  - define partial specifications
  - compute proofs by processing the connections
  - behind this…
    - how to create code annotations from model annotation?
    - how to prove that specifications and code match?

- prove component connections
  - prove the generated deployment code from architectural models
  - need for a notion of axioms, as the runtime itself cannot be proven (out of the scope of myccm)
  - In a general case, prove the framework code

- needs for specification of functional code
  - to be defined later