SCCC 2007

November 8-9th, Iquique, Chile

Specifying Fractal and GCM Components With UML

Solange Ahumada, Ludovic Apvrille, Tomás Barros, Antonio Cansado, Eric Madelaine and Emil Salageanu

Introduction

- ***Strong emphasis on system specification methods and tools***
  - Component-Based Software Development
    - UML 2[1] → Component Diagrams
- ***Specification***
  - Informal: non-expert users ☺ ambiguity ☹
  - Formal: expert user, longer time ☹ precise → verification ☺
  - Textual or Graphical

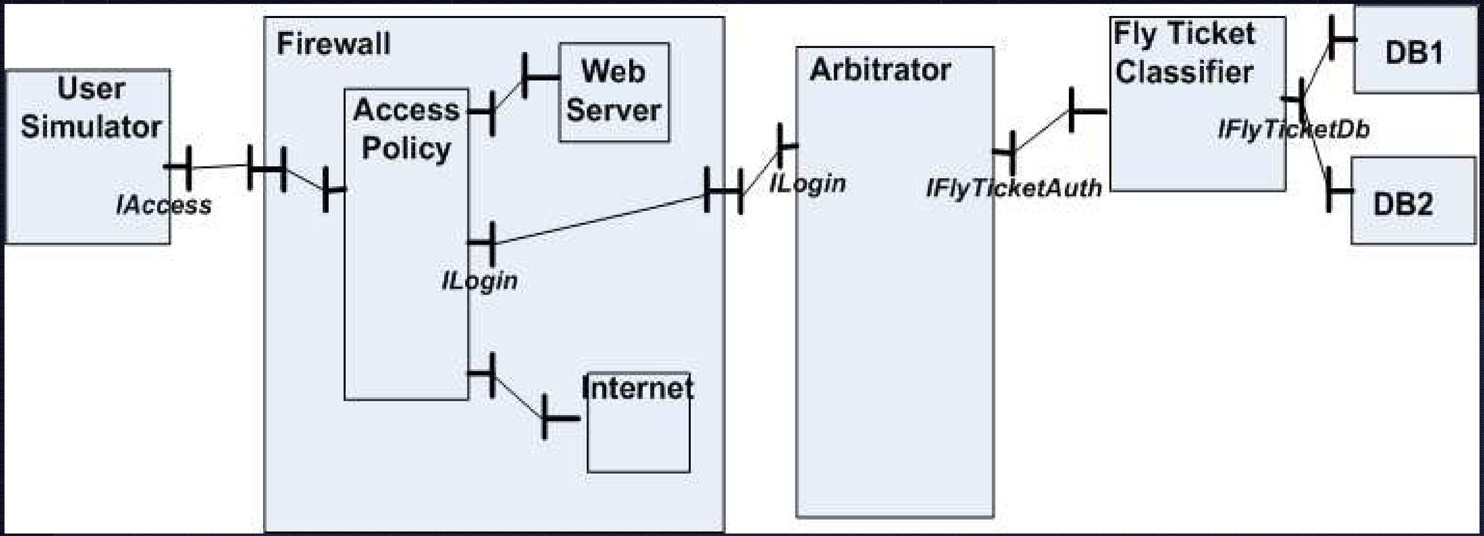
Our contribution:

- A UML-based framework and tool for specifying and model checking software components
- A novel UML profile proposal dedicated to distributed and asynchronous software components
  - Grid applications

State of the art of component models

- Introduction
- Our contribution
- ***State of the art of component models***
  - ***Fractal***
  - GCM
  - Turtle
  - UML 2 and Fractal
- CTTool
  - overview
  - CSD
  - SMD
  - tools
- GCM/ProActive components
- Language Extensions
- Conclusion
- References

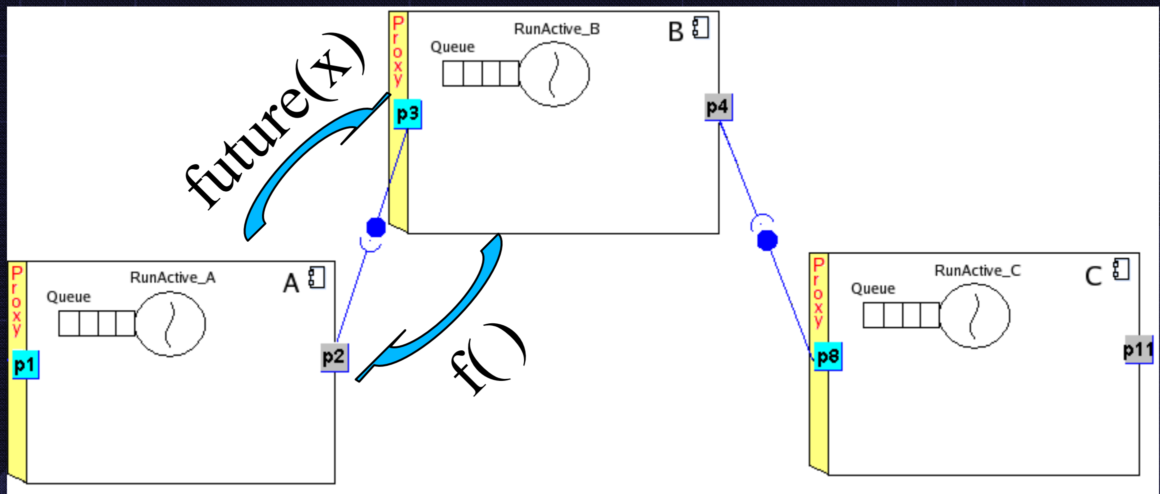
- Fractal[2]
  - Hierarchical component model
    - Component, controller, content, interface and binding.
  - Behavior protocol between components
  - Graphical editor, but no modelling tool.



State of the art of component models

Grid Component Model (GCM)[3]

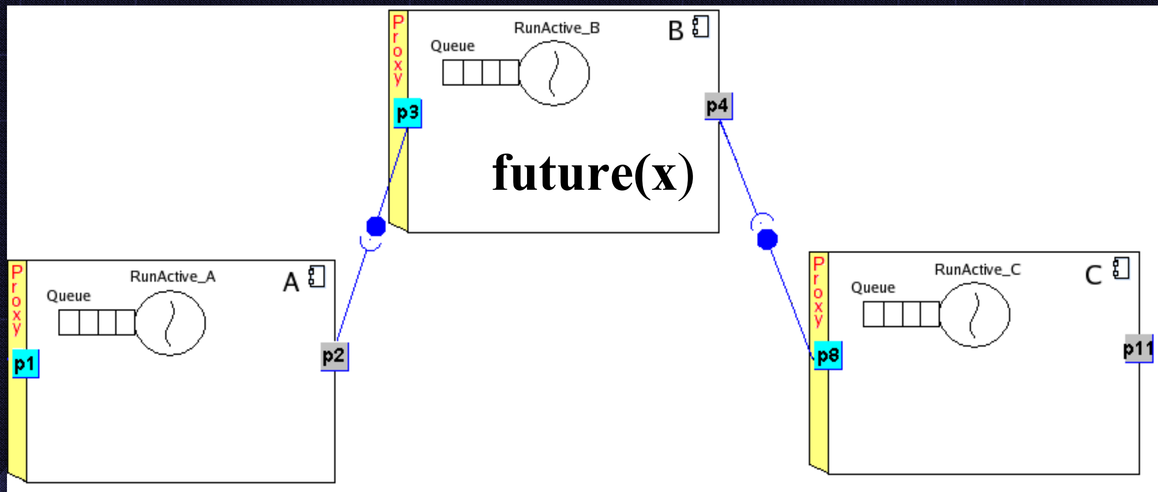
- Extension of Fractal to distributed applications
- Asynchronous method calls
- Implementation: ProActive[6]
  - Active object
  - Future value: rendez-vous



State of the art of component models

Grid Component Model (GCM)[3]

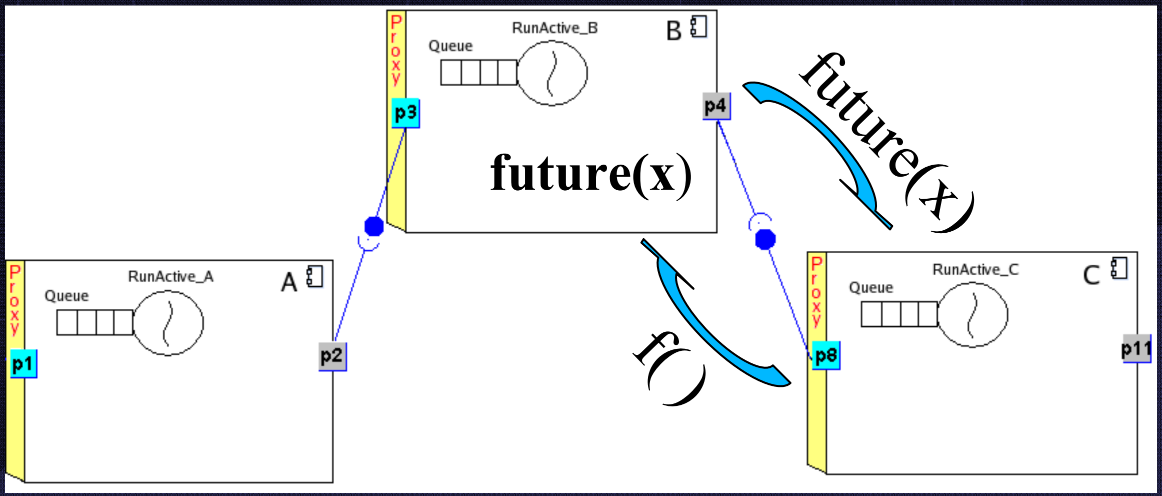
- Extension of Fractal to distributed applications
- Asynchronous method calls
- Implementation: ProActive[6]
  - Active object
  - Future value: rendez-vous



State of the art of component models

Grid Component Model (GCM)[3]

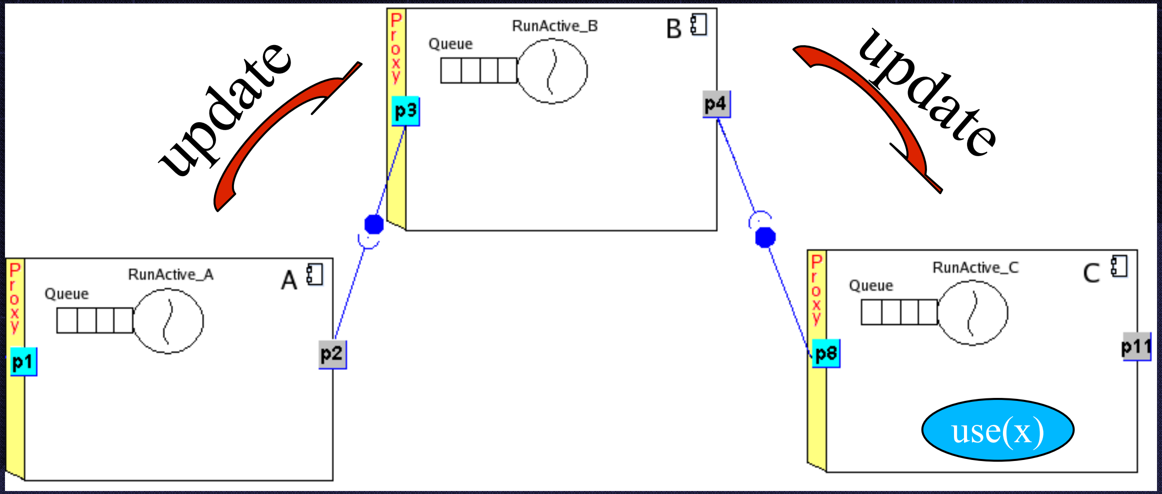
- Extension of Fractal to distributed applications
- Asynchronous method calls
- Implementation: ProActive[6]
  - Active object
  - Future value: rendez-vous



State of the art of component models

Grid Component Model (GCM)[3]

- Extension of Fractal to distributed applications
- Asynchronous method calls
- Implementation: ProActive[6]
  - Active object
  - Future value: rendez-vous

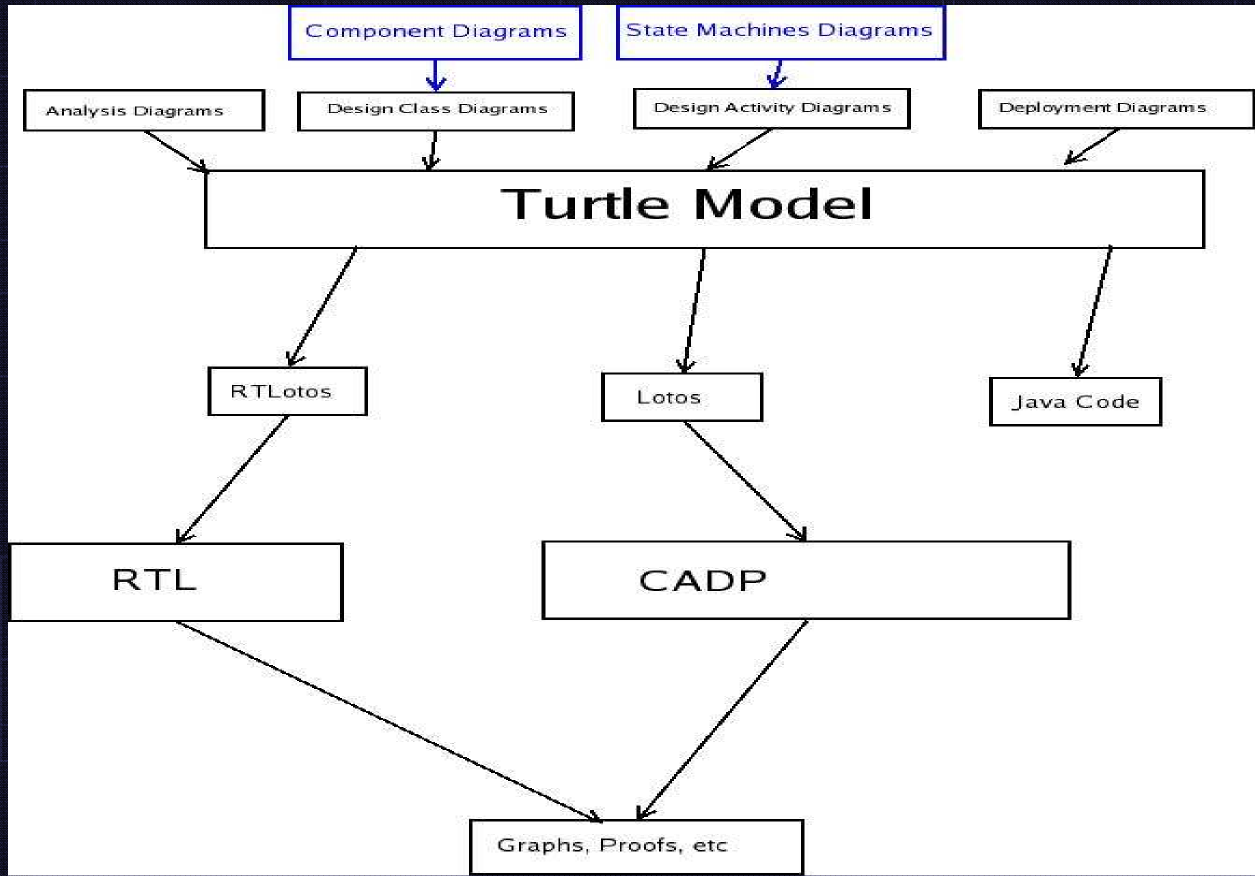


State of the art of component models

- Turtle[4] Model
  - Ludovic Apvrille, ENST, LabSoC Laboratory
  - UML(1.4) profile dedicated to the modelling and formal validation of real-time systems
  - Formal semantics for UML
  - Set of diagrams
- Implemented by TTool
- Analyze of possible system errors

State of the art of component models

- Introduction
- Our contribution
- State of the art of component models
    - Fractal
    - GCM
    - ***Turtle***
    - UML 2 and Fractal
- **CTTool**
    - **overview**
    - **CSD**
    - **SMD**
    - **tools**
- **GCM/ProActive components**
- **Language Extensions**
- **Conclusion**
- **References**



State of the art of component models

- UML 2 components and Fractal
  - Vladimir Mencl and Matej Polak, Charles University, Prague Distributed System Research Group
  - Mapping from Fractal to UML 2 (no behavior)
  - Component:
    - hierarchy / nested components
    - provided and required interfaces
    - attributes
  - Port
    - has provided and required Interfaces
    - has multiplicity (=> collection interfaces)

State of the art of component models

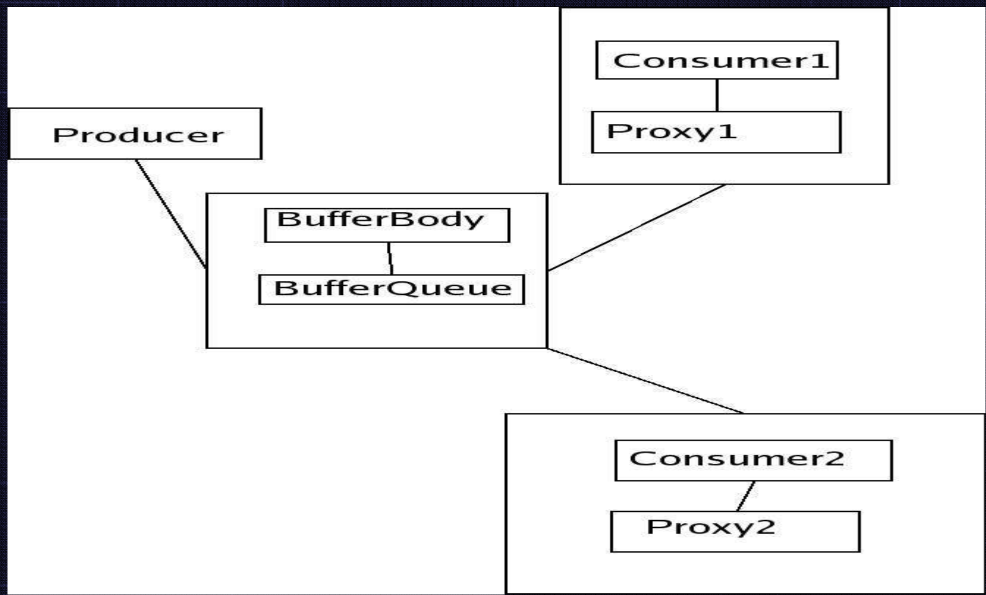
- Introduction
- Our contribution
- State of the art of component models
  - Fractal
  - GCM
  - Turtle
  - ***UML 2 and Fractal***
- **CTTool**
  - **overview**
  - **CSD**
  - **SMD**
  - **tools**
- **GCM/ProActive components**
- **Language Extensions**
- **Conclusion**
- **References**

- Connectors
  - Cannot be linked to interfaces (only to ports)
- Interfaces via Ports:
  - Only one interface per port.
  - Position of interface client/server.
  - Boolean attribute: mandatory or optional.

CTTool Overview

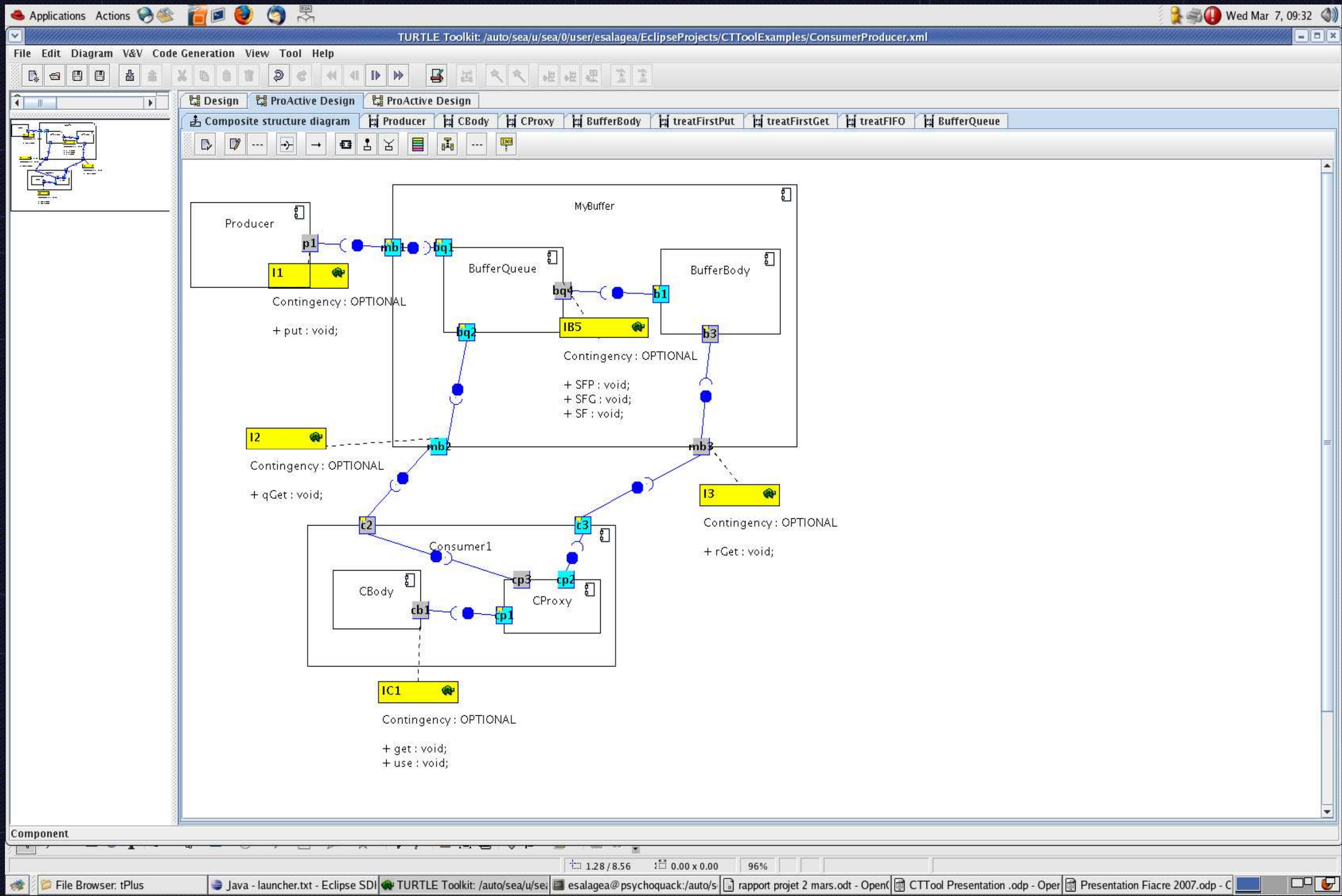
- Based on UML 2
- Fractal component model
- Editor + verification environment using TTool code base
  - generation of Lotos code
  - bridges to CADP toolset

Producer-Consumer Case-Study



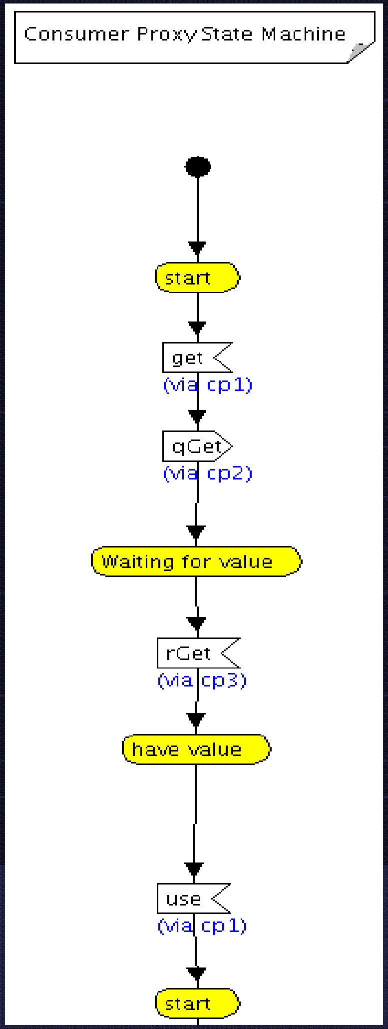
CTTool: Composite Structure Diagrams

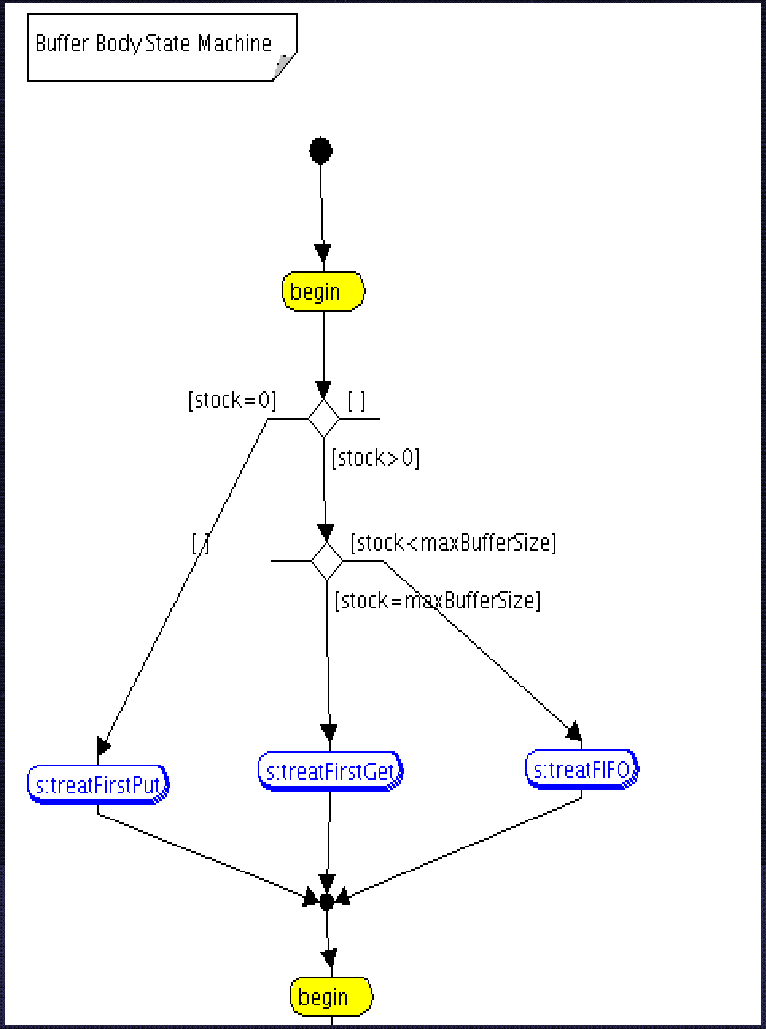
- Introduction
- Our contribution
- State of the art of component models
  - Fractal
  - GCM
  - Turtle
  - UML 2 and Fractal
- ***CTTool***
  - overview
  - ***CSD***
  - SMD
  - tools
- GCM/ProActive components
- Language Extensions
- Conclusion
- References



CTTool: State Machine Diagrams

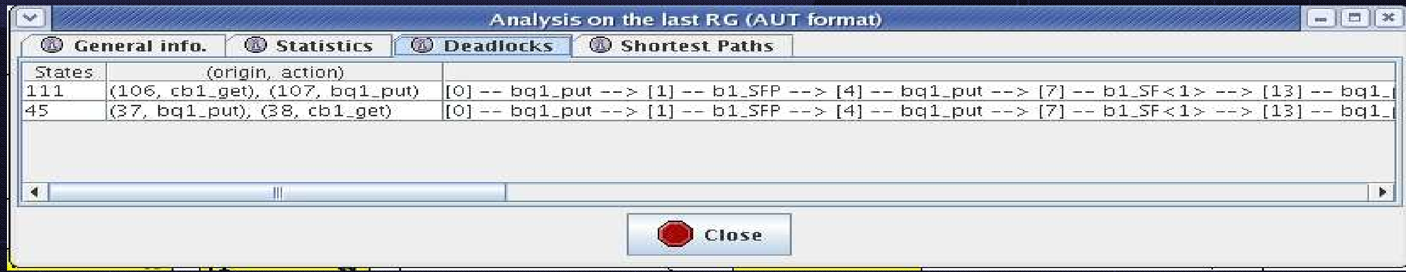
- Introduction
- Our contribution
- State of the art of component models
  - Fractal
  - GCM
  - Turtle
  - UML 2 and Fractal
- ***CTTool***
  - overview
  - CSD
  - ***SMD***
  - tools
- **GCM/ProActive components**
- **Language Extensions**
- **Conclusion**
- **References**

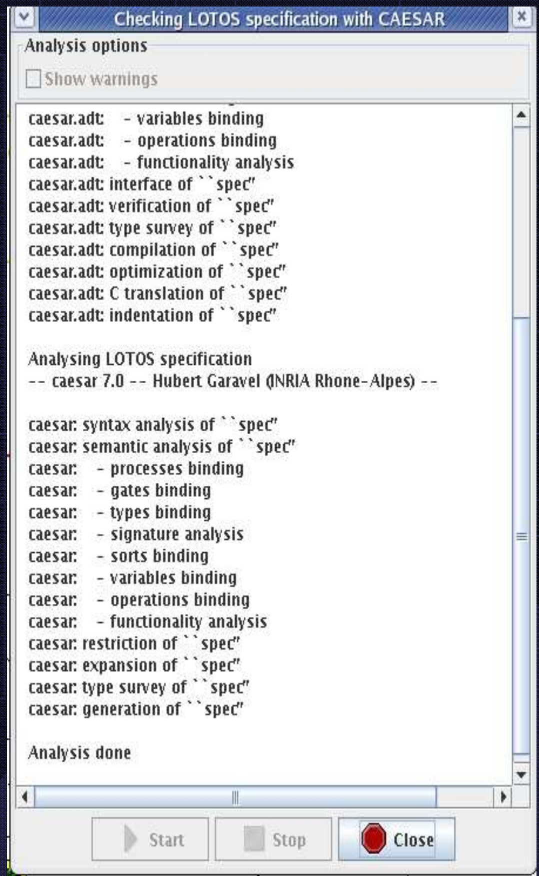


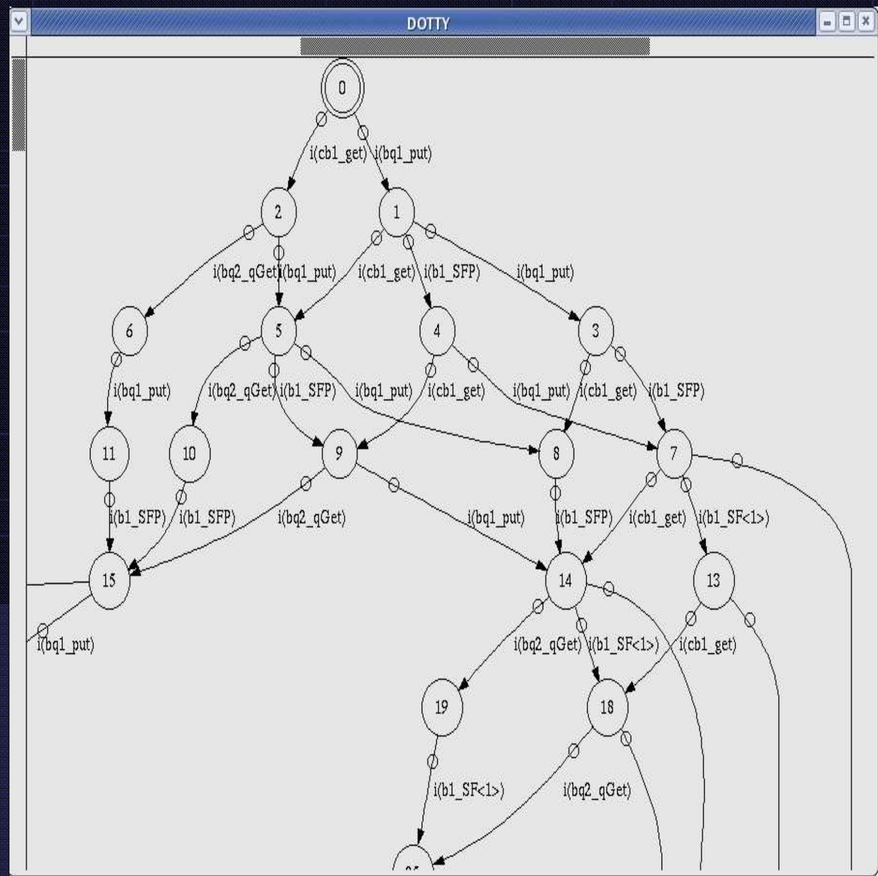


CTTool: use of CADP toolbox

- Introduction
- Our contribution
- State of the art of component models
  - Fractal
  - GCM
  - Turtle
  - UML 2 and Fractal
- ***CTTool***
  - overview
  - CSD
  - SMD
  - ***tools***
- GCM/ProActive components
- Language Extensions
- Conclusion
- References

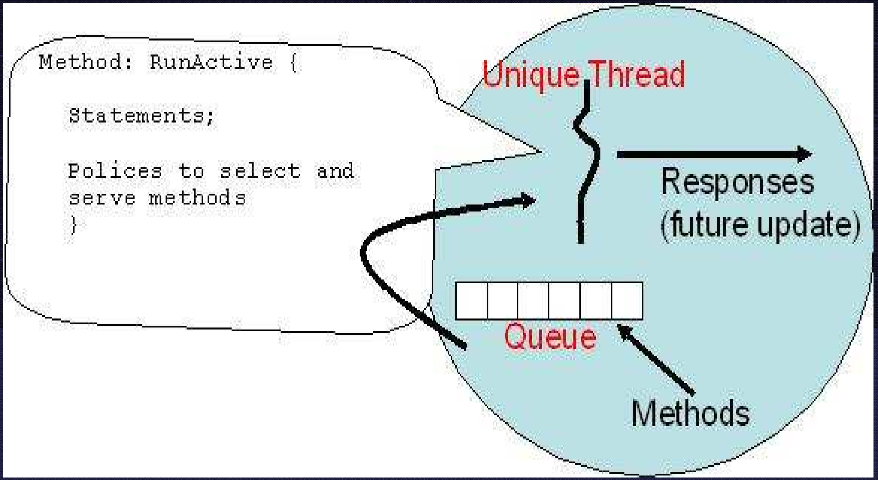






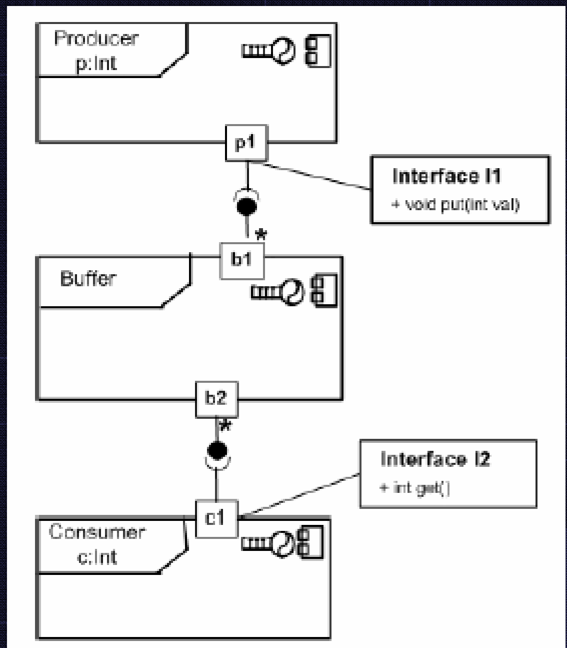
Specifying GCM/ProActive Components

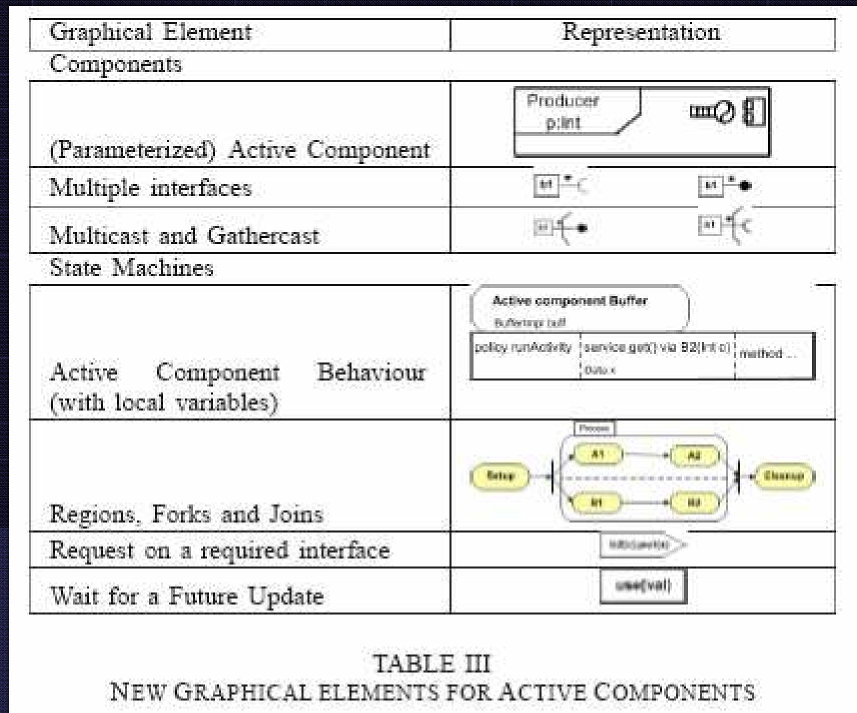
- ***Limitations***
  - Asynchronous method calls: queue, proxy.
  - Serving Policy
  - Multiplicity
  - Multicast / Gathercast interfaces



Language Extensions

- A GCM/ProActive component provides:
  - Request queue
  - Service thread





Graphical Element	Representation
Components	
(Parameterized) Active Component	Producer p:Int
Multiple interfaces	
Multicast and Gathercast	
State Machines	
Active Component Behaviour (with local variables)	Active component Buffer; policy runActivity; service get() via B2(int c); method ...
Regions, Forks and Joins	Setup; A1; A2; B1; B2; Cleanup
Request on a required interface	
Wait for a Future Update	use(val)

TABLE III
NEW GRAPHICAL ELEMENTS FOR ACTIVE COMPONENTS

Language Extensions

- New graphical construct for modelling the behaviour of an active component
  - Region diagrams
  - Sub-regions contains state machines diagrams
    - Service policy of the component
      - FIFO by default
      - States of the lifecycle
        - InitActivity
        - RunActivity
        - EndActivity
    - Service methods offered by the component
    - Sub-machines

Language Extensions

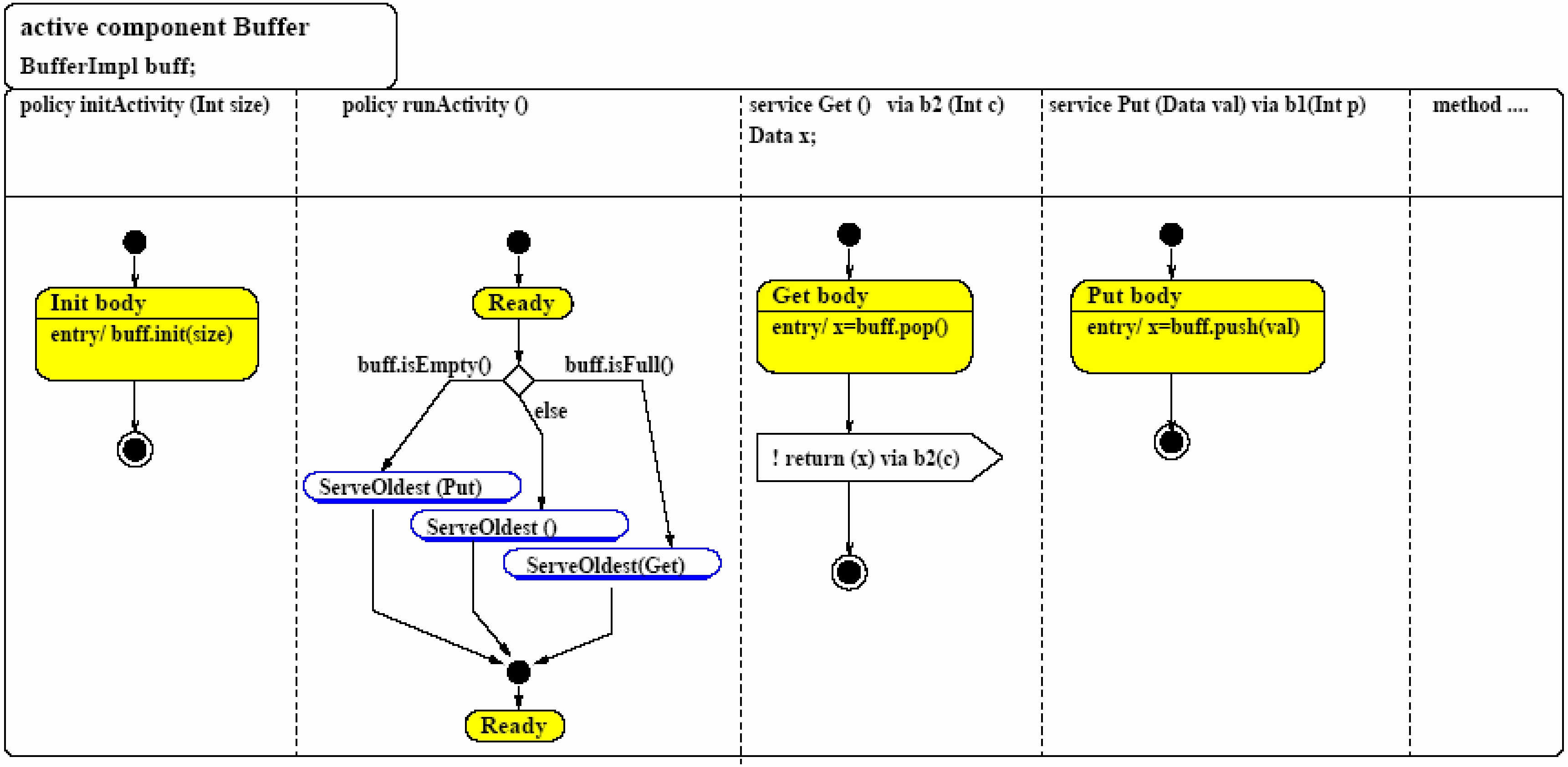
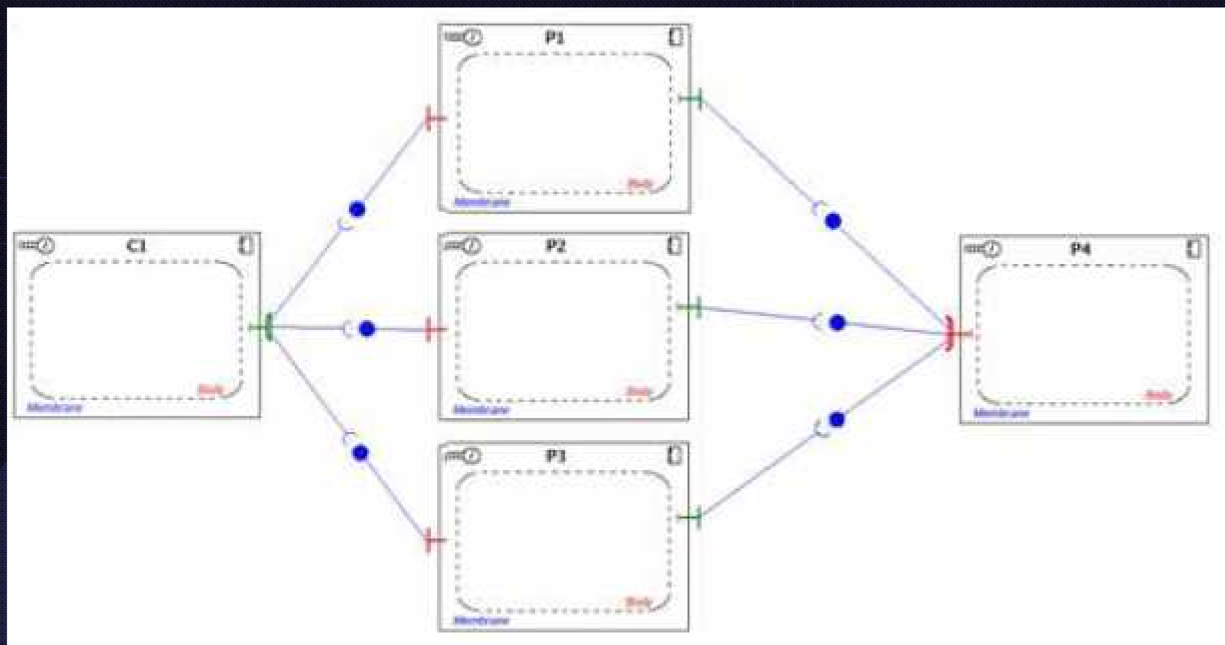


Fig. 4. State machines for the Buffer component

Language Extensions

- Introduction
- Our contribution
- State of the art of component models
  - Fractal
  - GCM
  - Turtle
  - UML 2 and Fractal
- CTTool
  - overview
  - CSD
  - SMD
  - tools
- GCM/ProActive components
- ***Language Extensions***
- Conclusion
- References

- Multicast client interface
  - A client interface connected to N server interfaces.
- Gathercast server interface
  - N client interfaces connected to a single server interface.



Conclusion

- Fractal and GCM components can be specified using UML 2 diagrams for specifications of architecture and behaviour.
- The graphical specification language is formal enough to be model-checked
- CTTool tested in a large scale case-study
  - Common Component Modelling Example (CoCoME)[5]
    - 16 components, 5 of them being composites
    - 5 layers of hierarchy

Conclusion

- Common Component Modelling Example (CoCoME)[5]
  - Generation of LOTOS model for model-cheking in CADP:
    - 81 distinc transition labels
    - Before reduction: 1.25 million states / 3 million transitions
    - After reduction: 9800 states / 33000 transitions
- Basis for addressing distributed components specification
  - To create a new UML profile for dealing with distributed active components.

References

[1] UML 2.0 Superstructure Specification, http://www.omg.org/cgibin/doc?ptc/2004-10-02, omg, Oct. 2004.

[2] E. Bruneton, T. Coupaye, M. Leclercp, V. Quema, and J. Stefani, “An open component model and its support in java.” in 7th Int. Symp. On Component-Based Software Engineering (CBSE-7), ser. LNCS 3054, may 2004.

[3] OASIS team and other partners in the CoreGRID Programming Model Virtual Institute, “Basic features of the grid component model (assessed),” 2006, deliverable D.PM.04, CoreGRID, Programming Model Institute.

[4] L. Apvrille, J.-P. Courtiat, C. Lohr, and P. de Saqui-Sannes, “TURTLE: A Real-Time UML Profile Supported by a Formal Validation Toolkit,” IEEE transactions on software Engineering, vol. 30, no. 7, jul 2004.

[5] “Common component modelling example (cocome).” [Online]. Available: http://agrausch.informatik.uni-kl.de/CoCoME

[6] D. Caromel, C. Delb´e, A. di Costanzo, and M. Leyton, “Proactive: an integrated platform for programming and running applications on grids and p2p systems,” Computational Methods in Science and Technology, vol. 12, no. 1, pp. 69–77, 2006.