A Concept for Dynamic Wiring of Components

Specification and Verification of Component-Based Systems Workshop

© Dirk Niebuhr

Clausthal University of Technology
Institute of Computer Sciences – Software Systems Engineering
Chair of Prof. Dr. Andreas Rausch
• Following common approaches, component wiring (System Configuration) is defined at development time

• Upcoming trends of dynamic systems
  – Pervasive Computing
  – Ubiquitous Computing
  – Organic Computing

⚠ Wiring needs to be changeable at runtime

⚠ Decision, which components fit together (Matching) has to be made at runtime
Problems for Runtime Matching of Components

Motivation | Problem | Approach | Conclusions

A Concept for Dynamic Wiring of Components

Possible Contracts:

\[ C_{AB} = \text{ComponentB} \times t_B \times \text{ComponentA} \times t_A \] holds\((t_B, t_A) = \text{false} \)

\[ C_{AC} = \text{ComponentC} \times t_C \times \text{ComponentA} \times t_A \] holds\((t_C, t_A) = \text{true} \)

Which one is fulfilled?
Several options for semantical match:

- Prove holds(prov, req)
  - only possible when using a restricted specification language

- Bisimulation
  - Excessive simulation overhead
  - Correct behavior only proven for next execution step

- Runtime-testing (*our favorite approach*)
  - Test cases need to be good enough
  - Testbed needs to simulate the „real“ system
• Test cases:
  • $t_{c_1} : \text{square}(0) : 0$
  • $t_{c_2} : \text{square}(3) : 9$
  • $t_{c_3} : \text{square}(-3) : 9$
Conclusions

- Proving the correctness of a component wiring at runtime is not possible in general.
- Runtime-testing enables us to detect mismatches in interface **semantics**.
- Test cases need to be „good enough“!
- Testbed needs to be specified (Duplicate the Components vs. Generating Test-Components) in detail.
- Test case optimization may be worthwhile (Local Testing <-> Global Testing, test only new components).
- Cyclic Dependencies and hierarchical composition has not been considered yet.