Component-Interaction Automata for Specification and Verification of Component Interactions

P. Vařeková and B. Zimmerova

Masaryk University in Brno
Czech Republic

IFM, 2005
Content

• Motivation  
  slide 3

• Specification languages  
  4 - 5

• Component-Interaction automata  
  6 - 8
  – Component-Interaction automata
  – Composition of Component-Interaction automata

• Conclusion and future work  
  9
Motivation

- Component-based software systems (CBSs)
- Component interactions in CBSs
- Verification of interaction properties

Specification languages
- Architecture description languages (ADLs)
- Automata-based languages
Specification Languages - ADLs

- **Wright** (R. J. Allen, 1997)
- **Darwin / Tracta** (J. Magee, J. Kramer, D. Giannakopulou, 1999)
- **SOFA** (F. Plasil, S. Visnovsky, 2002)

+ Hierarchical component architecture
+ Supported by tools
+ User friendly

- Verification of a small fixed set of properties
Specification Languages – Automata-based languages

- Interface automata (L. de Alfaro, T. A. Henzinger, 2001)
- Team automata (C. Ellis, 1997)

+ General and formal
+ Verification of temporal properties

- Specification of component interactions only
- Composition not parameterised by assembly
Component-Interaction Automata (CI automata)

- Automata based language
- Three types of actions
  - input, output, internal
- CCS like synchronization
  - one input and one output action which becomes internal later on
- Flexible composition
  - can be parameterised by characteristics of the system

- Close to architecture description languages
  - can be semi-automatically transformed into CI automata
- Close to Buchi automata
  - infinite traces
Component-Interaction Automaton

- Hierarchy
- States (initial states)
- Labels (structured - input, output, internal)
- Labelled transitions

Component: Database

Component: Client

Hierarchy: (1)

Hierarchy: (2)
Component-Interaction Automaton

- Hierarchy
- States (initial states)
- Labels (structured - input, output, internal)
- Labelled transitions

Component: Database

Component: Client
Component-Interaction Automaton

- Hierarchy
- States (initial states)
- Labels (structured - input, output, internal)
- Labelled transitions

Component: Database

Hierarchy: (1)

Labelled transitions:
- $(0, \text{Done}, -) \rightarrow (1, \text{Log}, 1)$
- $(2, \text{Ins}, -) \rightarrow (1, \text{Log}, 1)$

Component: Client

Hierarchy: (2)

Labelled transitions:
- $(0, \text{Done}, 2) \rightarrow (1, \text{Log}, 1)$
Component-Interaction Automaton

- Hierarchy
- States (initial states)
- Labels (structured - input, output, internal)
- Labelled transitions

Component: Database

Hierarchy: (1)

Component: Client

Hierarchy: (2)
Composition of CI automata

• every finite set of CI automata can be composed

• transition set of composed CI automaton $\subseteq$ complete transition space

In figures, states $ijk$ stands for $(i, j, k)$
Composition of CI automata

• every finite set of CI automata can be composed

• transition set of composed CI automaton \( \subseteq \) complete transition space
Composition of CI automata

• every finite set of CI automata can be composed

• transition set of composed CI automaton \( \subseteq \text{complete transition space} \)

In figures, states ijk stands for (i, j, k)
Conclusion and future work

• CI automata
  – language for specification and verification of component interactions

• Current work
  – equivalences and relations among CI automata
  – temporal logic

• Future work
  – automatic transformation from ADL to CI automata
  – automatic transformation from CI automata to input language of model checking tools
Properties verification

- Transition of component-interaction automaton to input language of verification tool

- Specification temporal properties
  \[ G((2, \text{ins, 1}) \Rightarrow F(1, \text{done, 2})) \]

- Temporal verification, model checking (Divine)
  true