Goal-Oriented Specification of Adaptation Semantics in Adaptive Systems

Greg Brown, Betty H.C. Cheng, Heather Goldsby, Ji Zhang

Presented by Ji Zhang

This work was supported in part by the U.S. Department of the Navy, Office of Naval Research under Grant No. N00014-01-1-0744, and in part by National Science Foundation grants CCR-9901017, EIA-0000433, EIA-0130724, CCF-0541131, and CNS-0551622, and ITR-0313142, and by Siemens Cooperate Research, and a Michigan State University Quality Fund Concept Grant.
Need for Dynamic Adaptation

• **Pervasive Computing.**
  – Promises anywhere, anytime access to data and computing.

• **Autonomic Computing.**
  – Promises self-managed and long-running systems that require only limited human guidance.
Tasks in Dynamic Adaptation

- enabling
- assurance
- monitoring
- decision-making
Proposed Approach

• Goal-oriented specifications can aid in correct design of adaptive systems
  – Representation of adaptation semantics [JSS06]
  – Graphical wrapper for formal definition

• Existing goal-oriented models
  – KAOS
    • Feather, van Lamsweerde, Fickas [Feather98]
  – Tropos/i* mixture
    • Lapouchnian, Mylopoulos [Yu05]
  – Either could have been used for graphically representing these adaptation semantics
Agenda

• KAOS Goal-Oriented Notation
• Adaptation Semantics Models
• Summary and Future Work
KAOS – Goal Models

- Use graphical objects to model system requirements [Dardenne93]
  - Goals
  - OR-refinement
  - AND-refinement
  - Sequential annotation [Yu05]
Operationalization

- Requirements are modeled in terms of system goals
  - Requirements
  - Agents
  - Operations
  - Events

![Diagram showing goals, requirements, agents, operations, and events]
A-LTL: Adapt Operator Extended LTL

- Extends the Linear Temporal Logic with an adapt operator [JSS06].

\[ \phi \xrightarrow{\Omega} \psi \]

- Used to specify adaptation semantics
One-Point Adaptation

- Initially behaves as **source**.
- At one point after “adapt request”, starts to behave as **target**.
One-Point Adaptation Model

monitor agent

request target

performance

output

adapt to target

monitoring

change agent

performance

adapt to target

adapt from source to target

in source

; in target
**Goal**: Achieve [adapt_from_sourc_to_target]
**Concerns** $S_{SPEC}$, $T_{SPEC}$, $A_{REQ}$
**RefinedTo**: in_source, in_target
**InformalDef**: The program initially satisfies $S_{SPEC}$. When a safe state is reached, the program starts to satisfy $T_{SPEC}$
**FormalDef**: $\left( S_{SPEC} \land \Diamond A_{REQ} \right)^{\Omega} T_{SPEC}$

---

**Goal**: Achieve [adapt_from_sourc_to_target]
**Concerns** $S_{SPEC}$, $T_{SPEC}$, $A_{REQ}$
**Refines**: adapt_from_sourc_to_target
**InformalDef**: The program satisfies $S_{SPEC}$
**FormalDef**: $S_{SPEC}$

---

**Goal**: Achieve [in_target]
**Concerns** $T_{SPEC}$
**Refines**: adapt_from_sourc_to_target
**InformalDef**: The program satisfies $T_{SPEC}$
**FormalDef**: $T_{SPEC}$
Guided Adaptation

- Initially behaves as source.
- A condition restrict the program to reach a safe state.
- Finally, the program behaves as target.
Guided Adaptation Model

- adapt from source to target
- adapt from source to restricted
- in target
- in source
- in restricted
- adapt to restricted
- adapt to target
- request restricted
- change agent
- monitor
- request target
- output
- request restricted
- monitor agent
- performance
- request target
- performance
- output
Overlap Adaptation

- The source and target behavior may overlap.
- A condition guides the program to reach a safe state.
Overlap adaptation model
Summary

• Goal-oriented specification of common adaptation semantics in KAOS notation
• Graphical wrapper to formal A-LTL adaptation semantics
• Benefits of formal specification as well as those of graphical representation
  – Ease of understanding
  – Tool support
  – Analyses
Future Work

• Use of adaptation semantics in different application domains.
• Goal-oriented decision-making
• Goal-oriented modeling in the context of model-driven development of adaptive systems [ICSE06]
Acknowledgements

• Members in the Software Engineering and Network Systems Laboratory at Michigan State University

• SEAMS reviewers

• Grants: This work was supported in part by the U.S. Department of the Navy, Office of Naval Research under Grant No. N00014-01-1-0744, and in part by National Science Foundation grants CCR-9901017, EIA-0000433, EIA-0130724, CCF-0541131, and CNS-0551622, and ITR-0313142, and by Siemens Cooperate Research, and a Michigan State University Quality Fund Concept Grant.