

Modeling Systems with ADORA

Requirements Engineering II

Michał Kołtonik Serge Hänni

29.11.2010

Outline

- 1 ADORA
- 2 ADORA tool
- 3 Scenario Simulations

Common approach to modeling

- different models for each aspect, loosely coupled collection of models
- flat models (all elements in one view)
- use cases loosely integrated with class and object models
- no consistency checking between different aspects

ADORA build upon 5 principles

- abstract objects
- hierarchical decomposition
- integrated model
- adaptable degree of formality
- contextual visualization

abstract objects, object sets, types

- object - abstract representation of a single instance in the system
- object set - set of object instances
- structure of object:
 - properties
 - attributes (different levels of formality)
 - directed relationships to other objects/object sets
 - operations
 - standardized properties (can be used to express non-functional requirements)
 - parts (optionally):
 - objects/object sets
 - states
 - scenarios
- types (for expressing commonalities of objects in different positions of decomposition hierarchy)

Relationships between objects

- all kind of information flow between objects
- directed binary associations
- abstract relationships (implied higher-level relationships)
- information flow (sending event or scenario interacts with object)

System's behaviour

- behaviour embedded into the object decomposition hierarchy
- behaviour easy to model on all levels of abstraction
- pure states (elementary states or refined pure statecharts)
- objects with state (additional properties and/or or other parts)
- events are broadcast only in object where it originates from
- state transitions can be modelled in diagrams or transition tables

Scenarios

- hierarchically structured scenarios integrated into hierarchical model
- instance scenarios
- type scenarios (use cases)

Contextual visualization

- Fisheye views
- display local detail in global context
- well-balanced

Demo

Purpose

- simulation based revalidation of requirements models
- → simulate incomplete, semi-formal models

Continuous Integration / Regression Testing

- revalidate/reverify frequently, but not manually
- automatically run previously recorded simulation runs

Automation vs. Informal Models

- use stored inputs for regression test, compare outputs with stored ones
- problems
 - required behavior/functionality missing
 - upon informally specified transitions: choose formal transition, or create new one
 - unhandled event: ignore or accept