Evolving Adaptable Systems: Potential and Challenges

Klaus Schmid, Holger Eichelberger
Institute of Computer Science, Software Systems Engineering
University of Hildesheim

Ursula Goltz, Malte Lochau
Institute for Programming and Reactive Systems
TU Braunschweig

Contents

1. Motivation

2. (D)SPLs and Adaptable Systems

3. Evolving Adaptable Systems

4. Solution Approach and Case Study
Evolving Systems

Long-Living (Software) Systems:

- continuous adaptation throughout whole life cycle [Parnas]
- changes enforced by a wide range of different factors

→ continuous evolution needed against aging of the system [Schmid, Lochau et al.]
Evolving Adaptable Systems: Potential and Challenges

Motivation

Long-Living Systems (LLS): future challenges
1. preventing aging and obsolescence
2. evolution and maintenance

Approaches:
- Software Product Lines: variability by construction
- Dynamic Software Product Lines: variability at runtime
- (Closed) Adaptable Systems: integration of evolution capabilities in a predefined scope

→ combinable to Evolvable Adaptive Systems?
- adaption space extension during evolution
- evolution of adaption space (models, components, …)
Evolving Adaptable Systems: Potential and Challenges

**Case Study: PROSA-X**

Generic Architecture for Control Module Integration of Parallel Robots [Steiner et al.]

Requirements:
- real time
- precision
- reliability
- safety
- flexible adaption of the component architecture

→ Self*-Properties: **Adaptivity** at runtime that preserves real time behavior
(D)SPL

Software Product Lines (SPL)
- set of product features composable to products (variants)
- product instantiation by parameterizable sets of artifacts (assets)
- integrates variability modeling [v.d. Linden et al., Schmid et al.]

Enhancement: Dynamic Software Product Lines (DSPL) [Hallsteinsen et al.]:
- variability at runtime
- but: predefined variability space
Evolving Adaptable Systems: Potential and Challenges

Adaptable Systems

Autonomous, strategic self-adjustments of the system to changing conditions, requirements, etc. at runtime (see e.g. [Cheng et al.])

- Closed Adaptivity: predefined adaption space with „discrete“ adaption modi
- Open Adaptivity: new adaptivity scenarios from environment injecting unpredictable adaptions, extensible adaption space

→ **Open Adaptivity** essential for LLS

Model-based adaption:
- models for adaption planning, analyzing, execution
- variability capabilities as part of modeling artifacts
- e.g.: generic, adaptable architecture models with configurable components
Case Study: Self-Management of PROSA-X

- Adaption modifies parameters, interconnections, etc. of implementation components
Evolving Adaptable Systems: Potential and Challenges

Case Study: PROSA-X Adaptivity

Self-Manager: adaption via runtime reconfiguration

Example: process migration
- node malfunction compensation
- re-scheduling in case of load fluctuations

Goal: preserving real time properties
Evolving Adaptable Systems

- combining (D)SPL runtime variability and Model-based (Open) Adaptivity
Adaptable Systems vs. DSPL

DSPL define a class of (closed) adaptive systems

<table>
<thead>
<tr>
<th>DSPL Variability Models</th>
<th>Adaptivity Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>capture evolution variants</td>
<td>complex criteria tradeoffs</td>
</tr>
<tr>
<td>logic based</td>
<td>metrics based</td>
</tr>
</tbody>
</table>

→ Both limit potential system evolutions by predefined adaption space

Extensions for Open Adaptivity support for LLS:

- extending space of potential adaptions
- „manual“ addition of functionality and further adaptivity capabilities
Evolving Adaptable Systems

Modifications on the structure of adaptive systems:

**Adaptivity Models** must include
- potential adaptations
- their interdependencies
- monitored events

**Implementation Components** must be augmented
- modification of existing components
- addition of new functionality

**Adaptivity Manager, Monitor, Executor**
- must be of high conceptual generality/genericity

→ Adaptable Adaptivity Manager
Solution Approach

- Evolution by "variable" Adaptivity Modules
Solution Approach

Derive adaptivity bundles similar to feature bundles [Schmid et al.] with:

- added capabilities

- declarative adaptivity model fragment

→ Coping with dependencies among model fragments using SPL variability management techniques
Case Study: Open Adaption

- PROSA-X: Closed Adaption capabilities

- Support for Open Adaptivity scenarios:
  - adding new skill primitives
  - adding new adaptronics
    - standard components
    - complex components, e.g. visual serving

Goal: Self-Manager → Evolvable Adaptivity Manager
- Adaptable Software Architectures for open adaptibility capabilities, strategies, etc.
- Adaptivity Modules/Models open for evolutionary changes
Case Study: Open Adaptivity Scenario

Evolvable Adaptivity-Manager

Adaptivity Model
Analyzing
Planning
Monitor
Adaptivity Executor

Open Adaptivity Scenario
adding visual serving component

Capability Implementation

Implementation Components
Implementation Components
Implementation Components

Bus

Adaptivity Manager Evolution
Layered Architecture Evolution
Challenges

Adaptivity Manager:
- How to characterize the ultimate power of an adaptivity manager?

Adaptivity Modeling Language (AML):
- Existing AMLs: Support for Evolution?
- Expressing Modularization in AML?
- Criteria for sufficient powerfulness of AMLs?

Implementation Components:
- Modularization of implementation components?
- Separation of individual components for flexible introductions?
- How to deal with interactions?
References


