Language and Framework Requirements for Adaptation Models

6th International Workshop on Models@run.time
Wellington, New Zealand, October 17, 2011

Thomas Vogel and Holger Giese
System Analysis and Modeling Group
Hasso Plattner Institute
University of Potsdam, Germany
Models@run.time for Self-adaptive Software

MDE & Models at Runtime for
- Knowledge
- Feedback Loop activities

Feedback Loop [Kephart and Chess, 2003]
Motivation

Models@run.time for Self-adaptive Software

- Focus on causal connection
  (e.g., discussions at MRT’09 and ’10)
  ⇒ Monitor and Execute

- Reusing or applying existing techniques for decision-making
  (rule-based or search-based)
  ⇒ Analyze and Plan

Feedback Loop [Kephart and Chess, 2003]
Related Work

Example solutions:
- rule-based: ECA, policies
- search-based: Utility functions, goals

Characteristics (requirements):
- Performance
- Support for validation
- Scalability

Stitch [Cheng, 2008]
- Requirements!
- Policy-based language
- System administration tasks

**RULE R_M**
**EVENT**
A new node N is detected onto the Platform
**CONDITION**
N.profile == PDA
**ACTION**
knowledge.domain.addNode(N)

**Adaptation model**

- Adaptation Rules → Variants
- Context model → Dependencies

**rule BecomeDA : // Becomes a DA**
**condition** ElectedDA and not LowBatt and not DA
**effect** DA

**AdaptationPolicy** ReplaceFiring
- \( (\text{Description} \quad \text{“Replaces firing component”}) \)
- \( (\text{Observation} \quad \text{energyReport (energy < 60)}) \)
- \( (\text{Response} \quad \text{RemoveComponent ReactiveFire}) \)
- \( (\text{Response} \quad \text{handfree_util = Distance}) \)
- \( \text{if (context.handfree AND STapp.handfree) or (context.handfree AND !STapp.handfree_offered) then 1 else 0} \)
- \( \text{response_util = if (context.response >= STapp. response) then 1 else 1 - ((STapp.response - context.response) / STapp.response)} \)
- \( \text{utility = if STapp.mem > context.mem then 0 else weight_hf + handfree_util + weight_rsp + response_util} \)

[Dubus and Merle, 2006, Morin et al., 2008, Fleurey et al., 2009, Georgas et al., 2009, Floch et al., 2006]
Related Work

Example solutions:

- rule-based: ECA, policies
- search-based: Utility functions, goals

Characteristics (requirements):

- Performance
- Support for validation
- Scalability

Stitch [Cheng, 2008]

- Requirements!
- Policy-based language
- System administration tasks

No systematic investigation of requirements for analysis and planning activities in conjunction with models@run.time

**RULE R.M**

EVENT

A new node N is detected onto the Platform

CONDITION

N.profile == PDA

ACTION

knowledge.domain.addNode(N)

rule BecomeDA : // Becomes a DA

condition ElectedDA and not LowBatt and not DA

effect DA

AdaptionPolicy ReplaceFiring

(Description “Replaces firing component”)
(Observation energyReport (energy < 60))
(Response RemoveComponent ReactiveFire)
(Response )

if (context.handfree_util AND $STapp.handfree_util) or
(context.handfree_util AND !$STapp.handfree_util) then 1 else 0

response_util =

if (context.response > $STapp.response) then 1
else 1 - (($STapp.response - context.response) / $STapp.response)

utility =

if $STapp.mem > context.mem then 0
else weight_tf * handfree_util + weight_rsp * response_util

[Dubus and Merle, 2006, Morin et al., 2008, Fleurey et al., 2009, Georgas et al., 2009, Floch et al., 2006]
Adaptation Models

MDE and models@run.time perspective (MODELS'10 Workshops)

Requirements for adaptation models concerning:

- **Languages** (meta-models, constraints, model operations etc.)
- **Frameworks** (execution environment)

**Note:** Not claiming a *complete* enumeration or *finalized* definitions
# Language Requirements (LR)

## Functional LR

<table>
<thead>
<tr>
<th>LR-1</th>
<th>Functional Specification/Goals</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR-2</td>
<td>Quality Dimensions</td>
</tr>
<tr>
<td>LR-3</td>
<td>Preferences</td>
</tr>
<tr>
<td>LR-4</td>
<td>Access to Reflection Models</td>
</tr>
<tr>
<td>LR-5</td>
<td>Events</td>
</tr>
<tr>
<td>LR-6</td>
<td>Evaluation Conditions</td>
</tr>
<tr>
<td>LR-7</td>
<td>Evaluation Results</td>
</tr>
<tr>
<td>LR-8</td>
<td>Adaptation Options</td>
</tr>
<tr>
<td>LR-9</td>
<td>Adaptation Conditions</td>
</tr>
<tr>
<td>LR-10</td>
<td>Adaptation Costs/Benefits</td>
</tr>
<tr>
<td>LR-11</td>
<td>History of Decisions</td>
</tr>
</tbody>
</table>

⇒ Concepts contained or referenced by adaptation models
⇒ Expressiveness of the language

## Non-functional LR

<table>
<thead>
<tr>
<th>LR-12</th>
<th>Modularity, Abstractions, Scalability</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR-13</td>
<td>Side Effects</td>
</tr>
<tr>
<td>LR-14</td>
<td>Parameters</td>
</tr>
<tr>
<td>LR-15</td>
<td>Formality</td>
</tr>
<tr>
<td>LR-16</td>
<td>Reusability</td>
</tr>
<tr>
<td>LR-17</td>
<td>Ease of Use</td>
</tr>
</tbody>
</table>

⇒ Quality of the language and adaptation models
Functional Language Requirements (I)

To-be specification of the running system (reference values)

**LR-1 Functional Specification/Goals**
Desired behavior, what the system should do

**LR-2 Quality Dimensions**
Desired QoS, how the system should be

**LR-3 Preferences**
Balancing competing quality dimensions or goals
Functional Language Requirements (I)

To-be specification of the running system (reference values)

**LR-1 Functional Specification/Goals**
Desired behavior, what the system should do

**LR-2 Quality Dimensions**
Desired QoS, how the system should be

**LR-3 Preferences**
Balancing competing quality dimensions or goals

As-Is situation of the running system

**LR-4 Access to Reflection Models**
Monitor & Execute changes through causally connected models

**LR-5 Events**
Trigger for analysis and planning; locating runtime phenomena
Functional Language Requirements (II)

Analysis of the running system

**LR-6 Evaluation Conditions**
Relate as-is (LR-4, 5) and to-be (LR-1, 2, 3) situations.

**LR-7 Evaluation Results**
Identify adaptation need, annotate reflection models (LR-4)
Functional Language Requirements (II)

Analysis of the running system

**LR-6 Evaluation Conditions**
Relate as-is (LR-4, 5) and to-be (LR-1, 2, 3) situations.

**LR-7 Evaluation Results**
Identify adaptation need, annotate reflection models (LR-4)

Planning of adaptation

**LR-8 Adaptation Options**
Variability (config. space) and how to change reflection models

**LR-9 Adaptation Conditions**
Applicability of adaptation options (by LR-4, 5, 7, 8)

**LR-10 Adaptation Costs and Benefits**
Select options wrt goals, qualities and preferences (LR-1, 2, 3)

**LR-11 History of Decisions** wrt analysis and planning
Characteristics and qualities of a language and models

**LR-12 Modularity, Abstractions and Scalability**
Composition of sub-models and different abstraction levels to promote scalability

**LR-13 Side Effects**
Explicit meta-information about side effects on reflection models \( \implies \) consistency of the running system

**LR-14 Parameters**
Built-in mechanism to adjust adaptation models at runtime
**LR-15  Formality**
How formal the modeling language should be?
⇝ Online or offline V&V of adaptation models

**LR-16  Reusability**
Degree of dependency between languages for adaptation models and reflection models

**LR-17  Ease of Use**
Modeling paradigm, notations, tools
⇝ Support engineers in creating, validating and verifying adaptation models
Framework Requirements (FR)

- Framework: Execution environment of adaptation models
- Specific requirements for executing/applying adaptation models

<table>
<thead>
<tr>
<th>Framework Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FR-1</strong> Consistency</td>
</tr>
<tr>
<td><strong>FR-2</strong> Incrementality</td>
</tr>
<tr>
<td><strong>FR-3</strong> Reversibility</td>
</tr>
</tbody>
</table>

**Note:** Typical non-functional requirements (reliability, security, etc.) of software are relevant for such frameworks as well, but left here.
**FR-1 Consistency**
Preserve consistency of reflection models (running systems)
⇝ Conditions for performing adaptations (LR-9)
⇝ Transaction support for adaptation models

**FR-2 Incrementality**
For example,
  - Locate need for analysis in reflection models by events
  - Incremental planning
  - Incrementally apply adaptation options on reflection models
  - ... to avoid searching or copying potentially large models

**FR-3 Reversibility**
Reverse incremental operations (do and undo of operations)
FR-4 Priorities
Organizing modular adaptation models by priorities, e.g., to order and analyze evaluation conditions based on criticality

FR-5 Time Scales
From exactly pre-defined adaptations for mission-critical situations to dynamically synthesizing adaptation plans

FR-6 Flexibility
Adapting adaptation models at runtime
⇝ Learning effects
⇝ Unanticipated scenarios
⇝ Hierarchical control
Adaptation Models and Feedback Loops

Language Requirements

<table>
<thead>
<tr>
<th>Functional LR</th>
<th>Framework Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR-1 Functional Specification/Goals</td>
<td>FR-1 Consistency</td>
</tr>
<tr>
<td>LR-2 Quality Dimensions</td>
<td>FR-2 Incrementality</td>
</tr>
<tr>
<td>LR-3 Preferences</td>
<td>FR-3 Reversibility</td>
</tr>
<tr>
<td>LR-4 Access to Reflection Models</td>
<td>FR-4 Priorities</td>
</tr>
<tr>
<td>LR-5 Events</td>
<td>FR-5 Time Scales</td>
</tr>
<tr>
<td>LR-6 Evaluation Conditions</td>
<td>FR-6 Flexibility</td>
</tr>
<tr>
<td>LR-7 Evaluation Results</td>
<td></td>
</tr>
<tr>
<td>LR-8 Adaptation Options</td>
<td></td>
</tr>
<tr>
<td>LR-9 Adaptation Conditions</td>
<td></td>
</tr>
<tr>
<td>LR-10 Adaptation Costs/Benefits</td>
<td></td>
</tr>
<tr>
<td>LR-11 History of Decisions</td>
<td></td>
</tr>
</tbody>
</table>

Non-functional LR

| LR-12 Modularity, Abstractions, Scalability | LR-15 Formality |
| LR-13 Side Effects                       | LR-16 Reusability |
| LR-14 Parameters                         | LR-17 Ease of Use |

Relationships between requirements and loops? \(\sim\) loop “patterns”

Thomas Vogel and Holger Giese | Adaptation Models | MRT11 | Oct 17, 2011
Decoupled Analysis and Planning

- Highlights LR where the corresponding concepts are relevant
- Explicitly covers all functional LR
- Rather sophisticated analysis and planning steps
- Rather longer time scales

⇝ Search-based approaches
Coupled Analysis and Planning

- Highlights LR where the corresponding concepts are relevant
- LR written in brackets are only implicitly covered
- Precise specification of adaptation (like ECA \(\approx\) LR-5, 6, 8)
- Rather short time scales

\(\Rightarrow\) Rule-based approaches

Adaptation Models

\(LR-(1,2,3),6,(7),8,(9,10,11)\)

Analyze/Plan

LR-4

Reflection Models

LR-5  LR-8

Monitor

LR-5

Execute

LR-8
• Highlights LR where the corresponding concepts are relevant
• LR written in brackets are only implicitly covered
• Precise specification of adaptation (like ECA \approx LR-5, 6, 8)
• Rather short time scales

\implies \textbf{Rule-based approaches}

\textbf{Extreme poles spanning a range of “patterns”}.
Conclusion and Future Work

Conclusion

- Adaptation models for self-adaptive software using MRT
- Language and framework requirements for adaptation models
- Adaptation models and feedback loops

Future Work

- Analyze existing approaches with respect to the requirements
- Engineer a language and framework for our approach (ICAC’09, MODELS’09 Workshops, SEAMS’10)
- Integration of multiple languages in a framework
References I

Summary of the 5th International Workshop on Models@run.time.
In Dingel, J. and Solberg, A., editors, MODELS’10 Workshops, volume 6627 of LNCS, pages 204–208. Springer.

4th International Workshop on Models@run.time.
In Ghosh, S., editor, MODELS’09 Workshops, volume 6002 of LNCS, pages 119–123. Springer.

In M-ADAPT’07, pages 61–68.

Rainbow: Cost-Effective Software Architecture-Based Self-Adaptation.
PhD thesis, Carnegie Mellon University, Pittsburgh, USA.

Applying OMG D&C Specification and ECA Rules for Autonomous Distributed Component-based Systems.
In Models@run.time’06.

Modeling and Validating Dynamic Adaptation.
In Chaudron, M., editor, MODELS’08 Workshops, volume 5421 of LNCS, pages 97–108. Springer.

Using Architecture Models for Runtime Adaptability.
Software, 23(2):62–70.

Rainbow: Architecture-Based Self-Adaptation with Reusable Infrastructure.

Using Architectural Models to Manage and Visualize Runtime Adaptation.
Computer, 42(10):52–60.

The Vision of Autonomic Computing.

An Aspect-Oriented and Model-Driven Approach for Managing Dynamic Variability.

Evolving Models at Run Time to Address Functional and Non-Functional Adaptation Requirements.
In Models@run.time’09, volume 509 of CEUR-WS.org, pages 31–40.
Generating Synchronization Engines between Running Systems and Their Model-Based Views.
In Ghosh, S., editor, MODELS'09 Workshops, volume 6002 of LNCS, pages 140–154. Springer.

Adaptation and Abstract Runtime Models.
In SEAMS’10, pages 39–48. ACM.

Model-Driven Architectural Monitoring and Adaptation for Autonomic Systems.
In ICAC’09, pages 67–68. ACM.

Incremental Model Synchronization for Efficient Run-Time Monitoring.
In Ghosh, S., editor, MODELS’09 Workshops, volume 6002 of LNCS, pages 124–139. Springer.

The Role of Models and Megamodels at Runtime.
In Dingel, J. and Solberg, A., editors, MODELS’10 Workshops, volume 6627 of LNCS, pages 224–238. Springer.