A flexible component model to integrate smart devices in service-based applications for ambient assisted living

Advisor: Jean-Marc Jézéquel
Co-Advisor: Olivier Barais
Reviewer: Elisabetta Di Nitto
Reviewer: Didier Donsez
Member: Daniel Thouroude
Member: Romain Rouvoy
Aging of the European population

More needs in health care and assistance

Limited resources in Hospital and care centers
- Rooms and Places
- Manpower

Promote cares at home
- People prefer to stay at home
- Shorter stays
Introduction :: State-Of-The-Art :: Contribution :: Validation :: Conclusion

AAL - Home Automation - Requirements

How to secure and help people at home?

Home automation: automation of tasks, support of chores

Huge number of technologies

Media
- Bus
- Radio
- Infrared
- PLC

Gateways
- RS232
- RS484
- TCP/IP

Transport Protocols
- IOBL
- X2D
- BackNet
- X10
- 6lowPan
- KNX
- DALI
- OWN
- ZigBee
- Zwave
- io-Homecontrol
- LonWork

Device
- iPod
- WiiMote
- SunSpot
- Temper
- Android
- Nabztag
- 3G modem (SMS)

How to support carers in their work?

Services through Internet

New services created regularly

Skype(SMS)
Google Document
UPnP
Vigicrue
Google Talk
Flickr
REST
Google Agenda
RTFE
Weather.com
Mail
XMPP
Facebook
DPWS

lundi 5 décembre 11
1 person = 1 solution = customization

- Pathologie(s)
- Disabilities
- User preferences
- Technical specificities
Human in the loop

Accessibility, Acceptability

Confidence in the system
**Human in the loop**

- Accessibility, Acceptability
- Confidence in the system

**Space in the loop**

- Variability management tools (versions, products, etc.)
- Remote control and administration
**Introduction**

- **State-Of-The-Art**
- **Contribution**
- **Validation**
- **Conclusion**

AAL - Home Automation - **Requirements**

- **Human in the loop**
  - Accessibility, Acceptability
  - Confidence in the system

- **Space in the loop**
  - Variability management tools (versions, products, etc.)
  - Remote control and administration

- **Time in the loop**
  - Adaptation to evolution of pathology, of technology, of user preferences
Introduction :: State-Of-The-Art :: Contribution :: Validation :: Conclusion

AAL - Home Automation - Requirements

- **Human in the loop**
  - Accessibility, Acceptability
  - Confidence in the system

- **Space in the loop**
  - Variability management tools (versions, products, etc.)
  - Remote control and administration

- **Time in the loop**
  - Adaptation to evolution of pathology, of technology, of user preferences

- **Variability in the loop**
  - Interoperability, device-service integration
  - Openness, connection of heterogeneous systems
AAL - Home Automation - Requirements

- Hardware not generic enough and/or not customizable for each demand.
- Customization at the software level?

A software engineering approach to create on-demand customized solutions

- Very flexible for Interoperability and Openness
- Supports adaptation and evolution, even after deployment
- Eases the management of Variability
## Introduction :: **State-Of-The-Art** :: Contribution :: Validation :: Conclusion

### Survey - CBSE - SOA - CM4SOA - Synthesis

<table>
<thead>
<tr>
<th>Generic Approaches</th>
<th>Interoperability</th>
<th>Openness</th>
<th>Adaptation</th>
<th>Evolution</th>
<th>Variability Management</th>
<th>Safety &amp; Security</th>
</tr>
</thead>
<tbody>
<tr>
<td>Darwin [GMK02]</td>
<td>+</td>
<td>+</td>
<td></td>
<td>+</td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Koala [RvdLKM00]</td>
<td>+</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Fractal [BCL+06]</td>
<td></td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>OSGi [All11]</td>
<td>+</td>
<td></td>
<td>+</td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>ESB [Cha04]</td>
<td>+</td>
<td></td>
<td>+</td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>SCA [sca]</td>
<td>+</td>
<td></td>
<td></td>
<td>+</td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>FraSCAti [MRRS10]</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>iPojo [EHL07]</td>
<td>+</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Domain-Specific Approaches</th>
<th>Interoperability</th>
<th>Openness</th>
<th>Adaptation</th>
<th>Evolution</th>
<th>Variability Management</th>
<th>Safety &amp; Security</th>
</tr>
</thead>
<tbody>
<tr>
<td>uMiddle [NT07]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Soprano [WSO+10]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Gaïa [RHC+02]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Dia Suite [CBC10]</td>
<td>+</td>
<td></td>
<td>+</td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Habitation [JRS+09]</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>WADL [CDT08]</td>
<td>+</td>
<td></td>
<td>+</td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>PervML [MPC06]</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>AutoHome [BDLM11]</td>
<td></td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>WComp [FHL+11]</td>
<td></td>
<td>+</td>
<td></td>
<td></td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Niagara [Tri08]</td>
<td></td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
</tbody>
</table>
Component-Based Software Engineering

Great level of abstraction for Home Automation devices

Explicit architecture

Reconfiguration

Verifications on architecture consistency

Interfaces of components (services, ports, etc.) are strongly linked to components’ implementations.

Independence between model / runtime

Openness for third party connexion

Dynamic adaptations with new component types
Introduction :: State-Of-The-Art :: Contribution :: Validation :: Conclusion

Service-Oriented Architecture
OSGi [All11], ESB[Cha04]

- Adaptation and evolutions while running
- Openness through the paradigm of service
- Distribution

- Lack of architectural description / reflexion
- Services are defined by fixed interfaces. No flexibility.
Component Model for Service-Oriented Architecture

SCA [SCA], FraSCAti [MRRS10], iPojo [EHL07]

- Adaptation and evolution while running
- Architecture description
- Openness
- Distribution

- Independence of the model (reflection)
- Ports/Services defined by fixed interfaces. No flexibility.
Component Models for SOA

How to improve the flexibility?

How to let designers choose between strict and relaxed checkers?

Electronic components

- Can always be connected, regardless of their technical constraints
- Assemblies are modeled and simulated before welding, to prevent predictable failures
Decouple the typing of a component and move it from the implementation to the model

- Mapping between model and implementation

A set of tools to ease the component management, and check assemblies conformance

- Adaptable checkers, flexible type verifications

A runtime to execute this new component model

- Independent model: Model@Runtime
Software components

**Component**: Elementary piece of software that provides functionalities to other components, and can use other components’ functionalities.

**Component Model**: defines how a component must be described, and how components can be assembled (may include constraints).

**Model of a Component**: abstractly describes a type of component; conforms to the component model.

**Model of a System**: describes the component instances involved in the system and their connections/interactions.
Introduction :: State-Of-The-Art :: Contribution :: Validation :: Conclusion

Overview - Component Model - Software tools - Outcomes

Electronics

PIN CONNECTIONS

Inputs

GND 1

VCC 8

2

Output

3

4

Balance/Strobe

5

VEE 4

6

Balance

7

LM211

Component Model

Software

Component Implementation

FakeSimpleLight.class

Copyright © ON Semiconductor 2011
Classical structure
Channels: explicit and manages the communications between components

- More flexibility (can be changed)
- Media-independent
- Synchronous/Asynchronous
- Can perform some parameters alignments
Decoupling

```java
@Provides(
    @ProvidedPort(name = "on", type = PortType.MESSAGE),
    @ProvidedPort(name = "off", type = PortType.MESSAGE),
    @ProvidedPort(name = "toggle", className = OnOffService.class)
)

@Requires(
    @RequiredPort(name = "state", type=PortType.MESSAGE, optional = true)
)

@DictionaryType{
    @DictionaryAttribute(name = "color"),
    @DictionaryAttribute(name = "initialState", defaultValue = "off", optional = true)
}

@ComponentType
public class FakeSimpleLight extends AbstractFakeStuffComponent {
```
Mapping

```java
@Ports({
  @Port(name = "off", method = "process"),
  @Port(name = "toggle", method = "off")
})
public void lightOff(Message o) {
  o.put(getName() + ":state", "off");
  MessagePort log = getPortByName("log", MessagePort.class);
  if (log != null) {
    log.process(o);
  }
  MessagePort statePort = getPortByName("state", MessagePort.class);
  if (statePort != null) {
    statePort.process(o);
  }
}
```
Good practice: prefer using Message Ports

Dictionaries of <key,value> easier to align than fixed interfaces of services

Benefits [NFM+10]

Model of the component different from the actual implementation => flexibility

A MessagePort and a method of a service can be mapped on the same implementation method => reuse, capitalization
Model @ Runtime [ICSE’09]

- Keep synchronized an independent model of the system
- Reasoning, checks can be performed at runtime with no impact on the running system
- Drives the adaptations at runtime
Pluggable checkers
Pluggable checkers

Design

Runtime

1. Development
2. Assembly Model
3. Sending for Deployment
4. Received for Deployment
5. Deployment

Overview - Component Model - Software tools - Outcomes
Overview - Component Model - Software tools - Outcomes

- Extraction of model from the code (CodeToModel)
  - Java Annotation Parser process
  - Integrated in the compilation phase (Maven)
  - Model embedded with compiled code

- From Model to Code
  - Java AST Parser/PrettyPrinter (Javaparser project on Google Code)
  - Generation of the component skeleton
  - Update of annotations and mappings
Introduction :: State-Of-The-Art :: Contribution :: Validation :: Conclusion

Overview - Component Model - Software tools - Outcomes

Synchronized Independent model
Drives Adaptations

Adaptable Checkers

Export component instances on different protocols

Common frame for components’ descriptions and links

Model@Runtime

Checkers

Wrappers

Component model

Device Interoperability

Service-Oriented Runtime

Models of components, Implementations, Physical Communications

Adaptable runtime environment
Hot deployment
• A runtime has been developed on top of an OSGi platform

• EnTiMid, specific for Home Automation and AAL, generalized in Kevoree to be usable in different contexts

• ~22% de participation sur Kevoree, ~83% sur EnTiMid
Members: Industrials, Carers, Academics/Laboratories

The case study: Definition

1. Interoperability
2. Evolution of user preference
3. Adaptation to the situation
4. Openness
Test environment

- Delta Dore Remote Control
- 3G Modem
- Nabaztag Rabbit
- Light controller (KNX)
- Meteo Station
- Touch Screen
Introduction :: State-Of-The-Art :: Contribution :: Validation :: Conclusion

Context - Experimental Setup - Tests - Conclusion

Interoperability

Evolution of user preference

Adaptation to the situation
Openness - UPnP Wrapper [NDBJ08]
Limitations

- Safety & Security (partially addressed)
- Scalability (cloud)
- Validity of the scenario
- Communication with smart devices (requires a gateway)
Decouple the component type and move it from the implementation to the model.

A set of tools to ease the component management, and check assemblies conformance.

A runtime to execute this new component model.

Approach validated on a scenario defined with caring professionals of the AAL.
Research

- Complete the tools with DSPL [INPJ09], Decision helping tools, Architecture Synthesis (CVL, etc.)
- Distribution, Model@Runtime for Cloud infrastructure management (PhD Fouquet, PhD Daubert, PhD Mattos)
- Automatic generation of Graphical User Interface [BBB+11]
- End-User programing [CC09]
Projects / Colaborations

Daum : Improve and Support on-field information system of Firefighters

Collaboration projects with Kerpape, acceptability/accessibility, Resources/Interactions/Process models (UBS-Lorient)

Business

EnTiMid received fundings from the Britany region for maturation (FEDER)

Neociti : Project of Spin-off (currently incubating), should start in January
Questions?

- Using MDE to Build a Schizophrenic Middleware for Home/Building Automation
  Grégory NAIN, Olivier BARAIS, Erwan DAUBERT, Jean–Marc JEZEQUEL
  In ServiceWave'08: Networked European Software & Services Initiative (NESSI) Conference, December 2008, Madrid, Spain

- EntiMid : un middleware aux services de la maison
  Grégory Nain, Olivier Barais, Régis Fleurquin, Jean–Marc Jézéquel
  In Conférence francophone sur les architectures logicielles : CAL 2009, March 2009, Nancy, France

- Taming Dynamically Adaptive Systems with Models and Aspects.
  Brice Morin, Olivier Barais, Grégory Nain, Jean–Marc Jézéquel
  In 31st International Conference on Software Engineering (ICSE'09), Vancouver, Canada, May 2009.

- Dynamic Software Product Lines for Service-Based Systems
  Paul Istoan, Grégory Nain, Gilles Perrouin, Jean–Marc Jézéquel
  In 9th IEEE International Conference on Computer and Information Technology, Xiamen, China, Oct 2009.

- Leveraging Models From Design–time to Runtime. A Live Demo.
  Brice Morin, Grégory Nain, Olivier Barais, and Jean–Marc Jézéquel.
  In 4th International Workshop on Models@Run.Time (at MODELS'09), Denver, Colorado, USA, Oct 2009.

- Integrating IOT and IOS with a component–based approach.
  Grégory Nain, François Fouquet, Brice Morin, Olivier Barais, and Jean–Marc Jézéquel.

- F4Plan: An Approach to build Efficient Adaptation Plans
  Francoise Andre, Erwan Daubert, Grégory Nain, Brice Morin, and Olivier Barais
  In Proceedings of the 7th International ICST Conference on Mobile and Ubiquitous Systems (MobiQuitous 2010), Sydney, Australia

- Combining Aspect-Oriented Modeling with Property-Based Reasoning to Improve User Interface Adaptation
  Arnaud Blouin, Brice Morin, Olivier Beaudoux, Grégory Nain, Patrick Albers and Jean-Marc Jézéquel
  EICS'11: Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems