Semantic-driven Goal-Oriented Development of AAL Environments

Marco Cameranesi, Claudia Diamantini, Domenico Potena, Emanuele Storti
e.storti@univpm.it

DII, Università Politecnica delle Marche, Ancona, Italy

CTS 2016, Orlando, Florida, Oct 31 - Nov 4

This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 License.
Outline

- Introduction
- Motivation
- Methodology
  - Elicitation of requirements
  - Knowledge representation in GoAAL ontology
  - Reasoning framework
- Applications
  - design of an AAL environment from scratch
  - redesign of an existing environment with changed requirements
- Conclusion & Future work
Introduction

Scenario: development of ambient assisted living (AAL) environments to support elderly people for:
- prevention and recognition of (medical) threats
- improvement of well-being
Motivation

Example: development of an AAL environment to detect falls

- monitoring of falls or patient’s stance?
- which measures to monitor?
- which sensors?
- where to locate them?
Motivation

Example: development of an AAL environment to detect falls

- monitoring of falls or patient’s stance?
- which measures to monitor?
- which sensors?
- where to locate them?

from a use case: 3 goals and 3 rooms → 6525 unique solutions!
Motivation

Main issues in AAL development

- choose among the multitude of sensors available in the market:
  - heterogeneous interfaces, standards, performances, price
- many sensors can be used for multiple purposes
- keep track of complex, interdependent requirements:
  - functional: e.g., detecting falls by monitoring floor or stance?
  - non-functional: e.g., privacy (avoid recognition of human body in bathrooms) or cost-effectiveness (install the minimal set of needed sensors)
- manage changing requirements:
  - of the patient (e.g., health conditions get worse)
  - of the environment
Methodology

1. Elicitation of requirements for reuse and documentation

2. Formal, modular, reusable knowledge base for representation of:
   - system requirements (goals/subgoals/tasks)
   - measures and relations with tasks
   - sensors, their capabilities and relations with measures

3. Logic-based reasoning framework to support development:
   - management of consistency
   - guidance to system design
1. Requirements elicitation

- Goal-oriented Requirement Language (GRL):
  - decomposition of goals in subgoals/tasks through qualified relationships (positive-negative contribution, correlation, dependency)
  - softgoals, to represent conditions with no clear-cut criteria for its satisfaction

- classification of high-level AAL goals according to degree of emergency (critical, dangerous, improvement of health conditions, comfort)

Advantages:

- high-level perspective on modeling current and future requirements
- support to the evolution and maintainance of requirements as engineering practice
1. Requirements elicitation

Example of goal diagram in GRL:
1. Requirements elicitation

Example of goal diagram in GRL:
2. Knowledge modeling (GoAAL ontology)

An ontology to formally represent relevant knowledge in AAL (goals/subgoals/tasks, measures and sensors) which integrates and extends various ontologies and vocabularies:

- **OntolStar** [Najera et al. 2011]: provides constructs for representing *goals* and their decomposition in subgoals/tasks

- **DogOnto** [Bonino et al. 2008]: focused on interoperation among systems within a smart environment (including environments and *rooms*)

- **KPIOnto** [Diamantini at al. 2014]: aimed at describing *measures* for monitoring of performances in enterprises

- **Semantic Sensor Network (SSN)** [Compton et al. 2008]: describes capabilities and properties of *sensors*
The ontology is conceptually divided in three modules:
ex:HeartRate.bpm :hasObject ex:Heartbeat.
Architecture: Measure module

Example of formulas

\[ \text{BodyMassIndex} = \text{BodyWeight} \times \text{BodyHeight} \]

\[ \text{Humidex} = \text{Temp} + \frac{5}{9} \times (6,112 \times 10^{\frac{7.5 \times \text{Temp}}{237.7 + \text{Temp}} \times \frac{\text{Humidity}}{100}} - 10) \]
3. Reasoning framework

- Custom reasoning tasks to derive non-explicit knowledge

- Logic Programming as unifying logic layer between OWL2RL and mathematic formulas
  - A) logic predicates for consistency maintenance of the knowledge base
  - B) predicates to support typical system design tasks

They rely on a mathematical LP library (PRESS, PRologic Equation Solving System) including predicates for basic math operation, equation solving and formula rewriting.
3. Reasoning framework

- Custom reasoning tasks to derive non-explicit knowledge

  → Logic Programming as unifying logic layer between OWL2RL and mathematic formulas
    - A) logic predicates for consistency maintenance of the knowledge base
    - B) predicates to support typical system design tasks

They rely on a mathematical LP library (PRESS, PRolog Equation Solving System) including predicates for basic math operation, equation solving and formula rewriting.
3. Reasoning framework

Support to development:

- **meaToSen(measures, sensors):** sensors needed to compute measures
- **tasksToSen(tasks, sensors):** sensors capable to (monitor measures to) achieve tasks
- **senToMea(sensors, measures):** measures that can be calculated from sensors

S:
- S₁: temp, lux
- S₂: humidity
- S₃: humidity, lux

\[ \text{senToMea}\{S₁, S₂, S₃\}, X \].

\[ X = \{ \text{temp, lux, humidity, humidex} \} \]
3. Reasoning framework

Support to development:

- \texttt{meaToSen(measures,sensors)}: sensors needed to compute measures
- \texttt{tasksToSen(tasks,sensors)}: sensors capable to (monitor measures to) achieve tasks
- \texttt{senToMea(sensors,measures)}: measures that can be calculated from sensors

\[ S_1: \text{temp, lux} \]
\[ S_2: \text{humidity} \]
\[ S_3: \text{humidity, lux} \]

?- \texttt{meaToSen(\{humidex\},X)}.
Support to development:

- `meaToSen(measures, sensors)`: sensors needed to compute measures
- `tasksToSen(tasks, sensors)`: sensors capable to (monitor measures to) achieve tasks
- `senToMea(sensors, measures)`: measures that can be calculated from sensors

\[ S_1: \text{temp, lux} \]
\[ S_2: \text{humidity} \]
\[ S_3: \text{humidity, lux} \]

?- `meaToSen(\{humidex\}, X)`. \[\rightarrow\] `meaToSen(\{temp, humidity\}, X)`. \\
`X = \{S_1,S_2\}, \{S_1,S_3\}`
3. Reasoning framework

Support to development:

- **meaToSen(measures, sensors)**: sensors needed to compute measures
- **tasksToSen(tasks, sensors)**: sensors capable to (monitor measures to) achieve tasks
- **senToMea(sensors, measures)**: measures that can be calculated from sensors
3. Reasoning framework

Support to development:

- `meaToSen(measures, sensors)`: sensors needed to compute measures
- `tasksToSen(tasks, sensors)`: sensors capable to (monitor measures to) achieve tasks
- `senToMea(sensors, measures)`: measures that can be calculated from sensors

\[ S_1: \text{temp, lux} \]
\[ S_2: \text{humidity} \]
\[ S_3: \text{humidity, lux} \]

?- `senToMea(\{S_1, S_2, S_3\}, X)`.  

\[ X=\{\text{temp, lux, humidity, humidex}\} \]
Development of an AAL environment

1. select one or more goals

2. decompose the goal(s) to all alternative sets of tasks $T_1, \ldots, T_n$

3. for each set of tasks $T_i$
   - get a list of sensors $S_i$ from $\text{taskToSen}(T_i, S_i)$

4. sort $S_1, \ldots S_n$ according to: #sensors, total cost or other criteria
Redesign of an existing system (changed requirements)

1. let $S$ is the set of already deployed sensors
2. get the set $M$ of available measures from $\text{senToMea}(S, M)$
3. decompose the new goals until all possible sets of tasks
4. for each alternative set of tasks $T_i$:
   - get the corresponding measures $M_i$
   - if $M_i \not\subseteq M$, for each measure in $M_i$ that $\notin M$:
     - get needed sensors with $\text{meaToSen}$
5. sort solutions according to: #sensors, total cost or other criteria
Applications

Redesign of an existing system (changed requirements)

Original requirement: \{temp\}

- @Kitchen, Sensor$_1$ (temp)
- @Living room, Sensor$_2$ (temp, humidity)
- @Bedroom, Sensor$_3$ (temp, humidex, lux)
Applications

Redesign of an existing system (changed requirements)

Original requirement: \{temp\}
New requirements: \{temp, humidity\}

- @Kitchen, Sensor$_1$ (temp)
- @Living room, Sensor$_2$ (temp, humidity)
- @Bedroom, Sensor$_3$ (temp, humidex, lux)
Applications

Redesign of an existing system (changed requirements)

Original requirement: \{temp\}
New requirements: \{temp, humidity\}

- @Kitchen, Sensor_1 (temp)
- @Living room, Sensor_2 (temp, humidity) ✓
- @Bedroom, Sensor_3 (temp, humidex, lux)
Applications

Redesign of an existing system (changed requirements)

Original requirement: \{\text{temp}\}
New requirements: \{\text{temp, humidity}\}

- @Kitchen, Sensor_1 (\text{temp})
- @Living room, Sensor_2 (\text{temp, humidity}) ✓
- @Bedroom, Sensor_3 (\text{temp, humidex, lux})

\[
\text{Humidex} = \text{Temp} + \frac{5}{9} \times (6,112 \times 10^{\frac{7,5 \times \text{Temp}}{237,7 + \text{Temp}}} \times \frac{\text{Humidity}}{100} - 10)
\]
Applications

Redesign of an existing system (changed requirements)

Original requirement: \{temp\}
New requirements: \{temp, humidity\}

@Kitchen, Sensor_1 (temp)
@Living room, Sensor_2 (temp, \texttt{humidity}) ✓
@Bedroom, Sensor_3 (temp, humidex, lux, \texttt{humidity}) ✓

\[
\text{Humidex} = \text{Temp} + \frac{5}{9} \times (6.112 \times 10^{\frac{7.5 \times \text{Temp}}{237.7+\text{Temp}}} \times \frac{\text{Humidity}}{100} - 10)
\]
Applications

Redesign of an existing system (changed requirements)

Original requirement: \{\text{temp}\}
New requirements: \{\text{temp, humidity}\}

- @Kitchen, Sensor_1 (\text{temp})
- @Living room, Sensor_2 (\text{temp, humidity}) \checkmark
- @Bedroom, Sensor_3 (\text{temp, humidex, lux, humidity}) \checkmark

Solutions:
- deploy a new Sensor_x (\text{humidity}) in Kitchen
- deploy a new Sensor_y (\text{humidex}) in Kitchen
Conclusions & Future work

A goal-oriented methodology for AAL environment development, which exploits requirement elicitation and formalization, and LP reasoning to support early stages of development

Comments

- reusable, incremental knowledge base for reference
- focus on cost-effectiveness and dinamicity of (changing) requirements

Future work

- evaluation of the methodology on a real-case
- tools & GUIs to ease the usage of the framework
- collaborative development of the ontology
Acknowledgements

This work has been partially funded by HicMO project
3. Reasoning framework

A) Consistency management

Predicates check whether a new formula is *identical*, mathematically *equivalent* or *incoherent* with other formulas.
3. Reasoning framework

A) Consistency management

Predicates check whether a new formula is *identical*, mathematically *equivalent* or *incoherent* with other formulas.

**Example**

BodyMassIndex = BodyWeight * BodyHeight

**New formula:** \( F = \frac{\text{BodyMassIndex}}{\text{BodyWeight}} \)

?- equivalence('F',X).

X = \{BodyHeight\}
3. Reasoning framework

A) Consistency management

Predicates check whether a new formula is *identical*, mathematically *equivalent* or *incoherent* with other formulas.

**Example**

BodyMassIndex = BodyWeight * BodyHeight

**New formula:** \( G = \text{BodyMassIndex} - \text{BodyWeight} \times \text{BodyHeight} \)

?- incoherence('G',X).

\( X = \{ \text{BodyMassIndex} \} \)
Experimentation: consistency

Experiments with a set of synthetic ontologies with increasing size\(^1\)

- # operands: 2..4, # nested formulas: 2..5
- 73980 unique tests

\(^1\) Intel Xeon CPU 3.60GHz, 3.50 GB RAM, Windows Server 2003 SP2