



D7.2: Specification, Activity Management System

Bruno Rosa, Jürgen Bund, André Teixeira
Distribution: Public

i2home

i2home: Intuitive Interaction for Everyone with Home Appliances based on Industry Standards
FP6-033502 Deliverable 7.2

27/02/09



Project funded by the European Community
under the Sixth Framework Programme for
Research and Technological Development



The deliverable identification sheet is to be found on the reverse of this page.

Project ref. no.	FP6-033502
Project acronym	i2home
Project full title	i2home: Intuitive Interaction for Everyone with Home Appliances based on Industry Standards
Instrument	STREP
Thematic Priority	Information Society Technologies
Start date / duration	01 September 2006 / 36 Months
Security	Public
Contractual date of delivery	M16
Actual date of delivery	27/02/09
Deliverable number	7.2
Deliverable title	D7.2: Specification, Activity Management System
Type	Report
Status & version	Final, 27/02/09
Number of pages	
Contributing WP	7
WP/Task responsible	MET
Other contributors	J. Alexandersson, G. Zimmermann, A. Pfalzgraf
Author(s)	B. Rosa, A. Teixeira, J. Bund
EC Project Officer	Peter Wintlev-Jensen
Keywords	Task Engine, Activity Management

The partners in i2home are:

Deutsches Forschungszentrum für Künstliche Intelligenz	DFKI
Siemens Business Solutions	SAG
Czech Technical University in Prague	CTU
Swedish Handicap Institute	HI
Meticube	MET
Zentrum für Graphische Datenverarbeitung	ZGDV
Access Technologies Group	ATG
VICOMTech	VICOMTECH
INGEMA – Instituto Gerontológica Matia	UC3M

For copies of reports, updates on project activities and other I2HOME-related information, contact:

The I2HOME Project Coordinator
Dr. Jan Alexandersson
Deutsches Forschungszentrum für Künstliche Intelligenz GmbH
Intelligent User Interfaces Department
Stuhlsatzenhausweg 3
66123 Saarbrücken, Germany
janal@dfki.de
Phone +49 (681) 302-5347 - Fax +49 (681) 302-5020

Copies of reports and other material can also be accessed via the project's administration homepage,
<http://www.i2home.org>

© 2006, 2007, 2008, 2009 The Individual Authors

No part of this document may be reproduced or transmitted in any form, or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission from the copyright owner.

Contents

Contents	4
1. Introduction.....	5
2. Activity Management Specification	6
2.1.1. Internal architecture	7
2.1.2. Grounding mechanism	10
2.1.3. Interaction with the user.....	11
References.....	13

1. Introduction

This work package consists in the implementation and integration of an Activity Management System into the UCH. This system is responsible for providing task support to the user, either by scheduling and executing complex tasks or by providing ways to guide users through such tasks.

A definition of a task can be: “a usually assigned piece of work often to be finished within a certain time” [1]. A task can also be called "activity", "goal", "job", or "action". Examples of tasks may be switching on the HVAC, turn on the TV, send a instant message...

The interaction of the Activity Management System with the targets (devices or services) will be accomplished using the existent socket instances (for the existent targets). Interaction with the user will also be accomplished using sockets: each task model will have a socket for interaction with the user, created specifically for that purpose.

2. Activity Management Specification

The development process of the Activity Management System is based on the requirements collected from the user groups in WP1 scenarios. User-centered analysis (task analysis, social settings) methods will be employed to refine the requirements into a set of suitable scenarios.

The Activity Management System will include a task engine implementation, based on an already existent open source engine (provided to i2home by Charles Rich, Ph.D.). The modeling of tasks and the interaction between them is represented by task models. This representation follows the standard CEA-2018 which defines the semantics and an XML notation for task models.

A task in a task model can represent single or multiple operations to be executed. These operations can be related to interaction with targets or users. Examples of interactions with targets are: switch on the HVAC, set the temperature to 25°C, turn on the TV, switch to channel 4... Examples of interactions with the user are: guide the user in the process of executing some tasks (for instance to prepare a salad for lunch).

The task engine to be used will deal specifically with task models compliant with the CEA-2018 ANSI standard and will be responsible for (among others):

- Parsing the task model documents (task model descriptions);
- Creating a runtime representation of the task models and the tasks present in it;
- Maintain and update that representation to reflect the changes in the system (changes in targets or done by the user);
- Determine the next task to be executed at a given moment;
- Execute tasks marked for execution (if possible).

2.1.1. Internal architecture

As mentioned before, all the interaction between targets, users and the Activity Management System will be made using sockets that are placed in the Socket Layer.

The figure bellow shows the new system architecture now including the Activity Management System in the global UCH architecture:

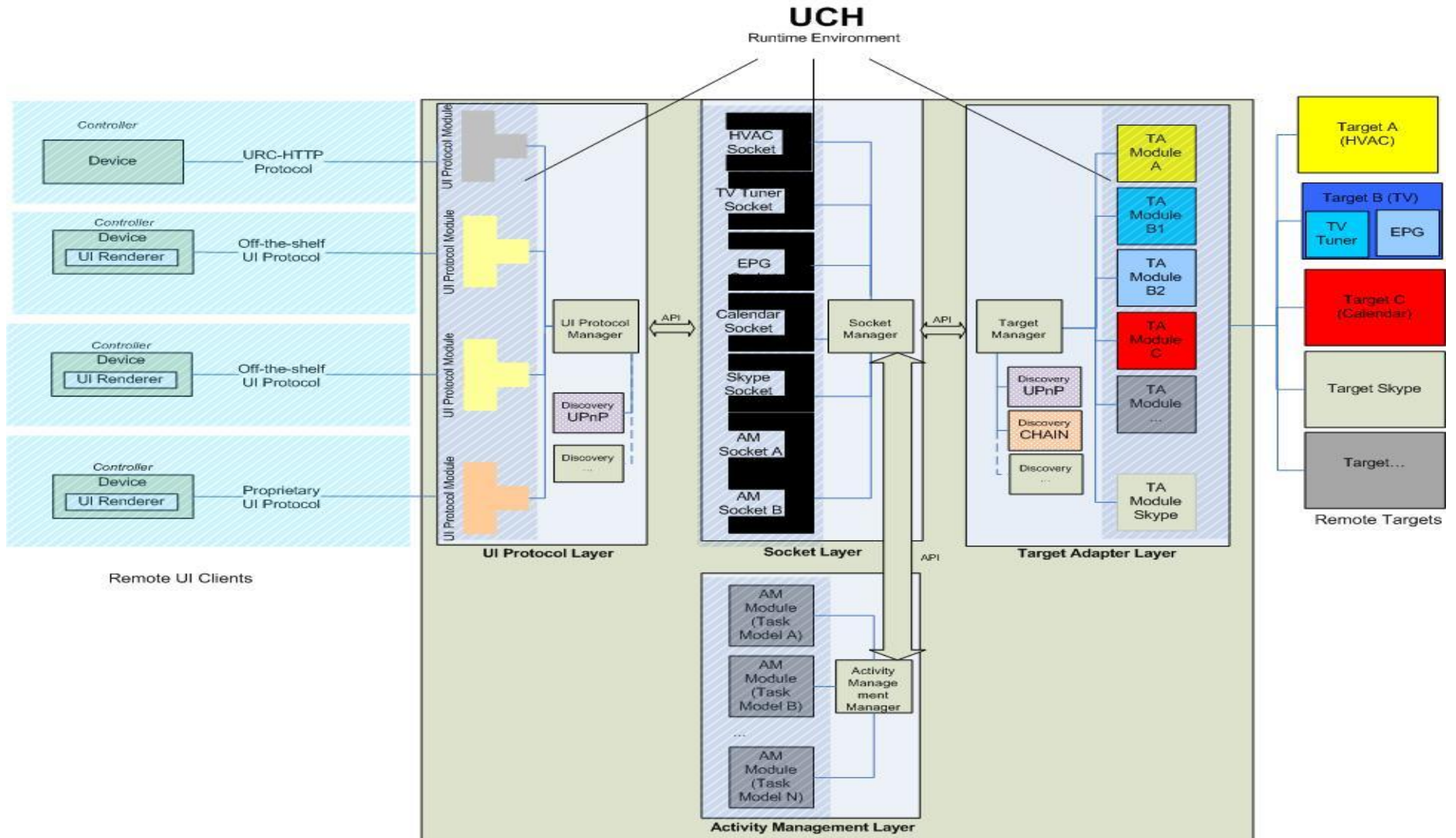


Figure 1 - Block Diagram with System Architecture

As showed in the above figure, the Activity Management System architecture will be modular. It will have a manager (Activity Management Manager) that will be responsible for each individual Activity Management Module.

Each Activity Management Module will be responsible for providing the runtime environment necessary to execute a CEA-2018 task model. To accomplish this, each individual AM module will have its own task engine instance and its own ecma script engine.

The task engine implementation will be responsible for parsing the original task model .XML file and to create its runtime representation. It is also responsible for obtaining tasks and task models referred in the original task model. It will also keep the runtime representation up to date with the latest changes and with the relevant changes on the surrounding environment.

The Ecma script engine will be responsible, for example, for evaluating:

- The scripts present in the tasks (not obligatory to be present): this scripts are present on the tasks that are meant to interact with targets, and its evaluation will lead to actual interaction with the targets.
- Task's pre-conditions (not obligatory to be present): this scripts must be evaluated in order to determine if a task can be selected for execution or not.
- Task's post-conditions (not obligatory to be present): this evaluation will be used to determine the success of the execution of the given task.
- Task's input/output slots (not obligatory to be present): this evaluation is meant to calculate the actual value of the slots of tasks.

The Ecma script engine will also be responsible for keeping the bindings up to date: bindings are the relations between the slots of the different tasks, for instance, an output slot of a task may be used as an input slot of another task.

Each of the Activity Management Modules will also expose a socket to allow interaction with the user. Each socket will be made specific to the task model to be executed in the AM module. This way the socket will have all the socket elements necessary to display and request all necessary information to/from the user (see 2.1.3 for more details).

2.1.2. Grounding mechanism

As mentioned above, tasks may contain ecma-script scripts that may be used for interaction with the targets. It will be created a specific ecma-script library with the purpose of allowing the execution of operations on the targets triggered from the scripts present in the task model. This library is called URC library and it will interact with the Socket Layer of the UCH, which has access to all sockets from all targets available in the environment, and also to all sockets from other task models. It can be used in any of the scripts present in the task model, and it contains, among others, the following methods:

- `URC.getValues()` – used to retrieve the actual value of a socket element of a specific socket.
- `URC.setValues()` – used to change the value of socket elements present in a specific socket.
- `URC.invokeCommand()` – allow the invocation of commands present in a specific socket.

2.1.3. Interaction with the user

With the purpose of interacting with the user it will be created a task model that contains specific pre-defined tasks that allow that interaction. This task model can be referenced from any other task model (it is always available in the UCH).

Some of the tasks that will be available in this task model are:

- **Input Request**: this task will be used to request an input value from the user. It has an input slot that will contain the value inserted by the user, and an output slot that contains the message to be displayed to the user explaining what he needs to insert.

```
<task id="InputRequest">
  <input name="message" type="string"/>
  <output name="response" type="string"/>
</task>
```

- **Confirmation Request**: this task will be used to request a confirmation from the user. It has an output slot that contains the message that will be displayed to the user requesting a confirmation, and it has an input slot to contain the response given by the user to this request (in this case 'Ok' or 'Cancel').

```
<task id="ConfirmRequest">
  <input name="message" type="string"/>
  <output name="response" type="string"/>
</task>
```

- **Instruction Notify**: this task is used to display an instruction to the user in the form of a notify (alarm). The message that contains the actual instruction must be available through resources [6].

```
<task id="InstructionNotify" />
```

In order to allow the interaction of an Activity Management module with the users, the task model that is being executed in the Activity Management module will have its own socket description. This socket description must contain mapping elements to map socket elements present in this socket to tasks, steps and input/output slots present in the task model. There will be two types of mappings:

- Step mapping: this mapping will be used to map all the existent input/output values in a step in the task model. Imagine that we have an input slot from a step referent to a task that represents the volume that the user wants to change the TV to. To allow the user to insert this value we add a variable to the socket description that will contain it. By using this mapping when the user inserts the volume value in its interface, the Activity Management assigns that value to the proper step input slot.
- Task mapping: this mapping is used to map command elements to tasks (or steps), so that when a user invoke the command in its user interface the AM knows what task (or step) to execute (if possible).

References

- [1] Merriam-Webster Dictionary
<http://www.merriam-webster.com/dictionary/task>

- [2] i2home Deliverable D1.3: Scenario Description Report, i2home project
<http://intranet.i2home.org/wiki/project/Deliverables/D1.3/>

- [3] CEA-2018 (ANSI Standard)
http://www.ce.org/Standards/browseByCommittee_4467.asp

- [4] ISO/IEC 24752-1:2008 Information Technology — User Interfaces — Universal Remote Console. Part 1: Framework. ISO, 2008.
http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=42309

- [5] ISO/IEC 24752-2:2008 Information Technology — User Interfaces — Universal Remote Console. Part 2: User Interface Socket Description. ISO, 2008.
http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=42310

- [6] ISO/IEC 24752-3:2008 Information Technology — User Interfaces — Universal Remote Console. Part 3: Presentation template. ISO, 2008.
http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=42311

-
- [7] ISO/IEC 24752-4:2008 Information Technology — User Interfaces — Universal Remote Console. Part 4: Target Description. ISO, 2008.
http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=42312
- [8] ISO/IEC 24752-5:2008 Information Technology — User Interfaces — Universal Remote Console. Part 5: Resource Description. ISO, 2008.
http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=42313
- [9] URC Technical Primer 1.0, Draft Technical Report 2008-08-14
<http://myurc.org/TR/urc-tech-primer1.0-20080814/>
- [10] Universal Control Hub 1.0, Draft Technical Report 2008-08-14
<http://myurc.org/TR/uch1.0-20080814/>
- [11] URC-HTTP Protocol 2.0, Draft Technical Report 2008-08-14
<http://myurc.org/TR/urc-http-protocol2.0-20080814/>
- [12] Resource Property Vocabulary 1.0, Draft Technical Report 2008-08-14
<http://myurc.org/TR/res-prop-vocab1.0-20080814/>
- [13] Resource Server HTTP Interface 1.0, Draft Technical Report 2008-08-14
<http://myurc.org/TR/res-serv-http1.0-20080814/>