



Extending a smart home multi-agent system with role-based access control

*International Conference on
Internet Technologies & Society 2014*

*10 – 12 December
New Taipei City, Taiwan*

Enrico Denti, Roberta Calegari and Marco Prandini



Outline

- ◆ Introduction and context
- ◆ Background
- ◆ Home Manager: new scenario
- ◆ Future Work
- ◆ Conclusion



Outline

- ◆ Introduction and context
 - Context
 - Home Manager 1.0
 - The new objectives
- ◆ Background
- ◆ Home Manager: new scenario
- ◆ Future Work
- ◆ Conclusion

- ◆ Home appliances / devices becoming smarter and connected
- ◆ New pervasive scenario in many application fields
 - Remote appliance monitoring /control for
 - ◆ energy saving
 - ◆ air conditioners control
 - ◆ ...
 - Domotics
 - Ambient Intelligence (Aml)
 - Smart Environments
 - ...

Focus onto *people's need*:

- User-friendliness
- User-empowerment
- Support for human interaction

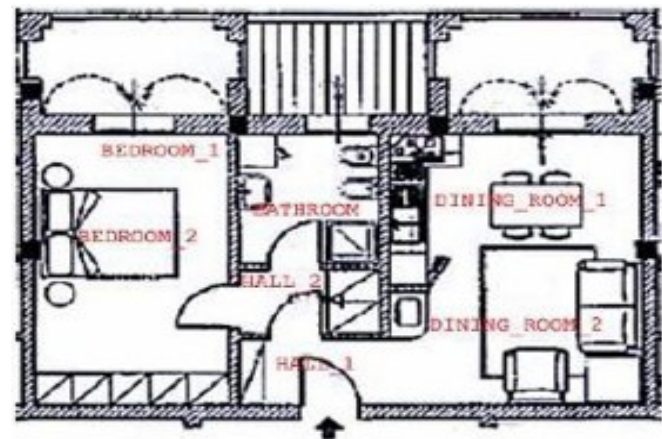


Home Manager 1.0

- ◆ Agent-based application for the control of an intelligent home
- ◆ Designed according to the **SODA** agent-oriented methodology
- ◆ Implemented on the top of **TuCSon** multi-agent infrastructure

- 😊 Basic set of home appliances / devices
- 😊 Basic support to policies and user goals
- 😊 Users preferences

- 😐 Basic access control model
- 😞 Very limited policies
- 😞 (Too) simple scenario



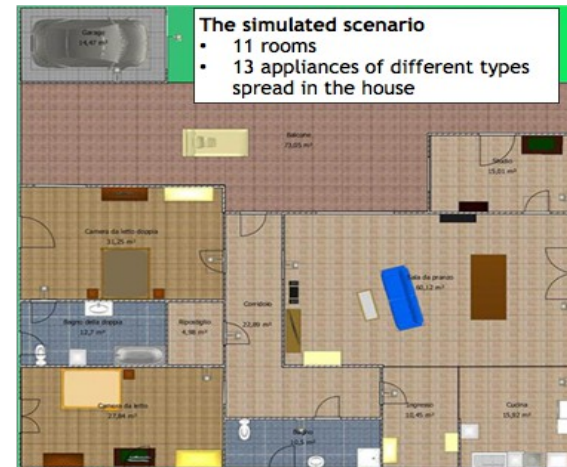
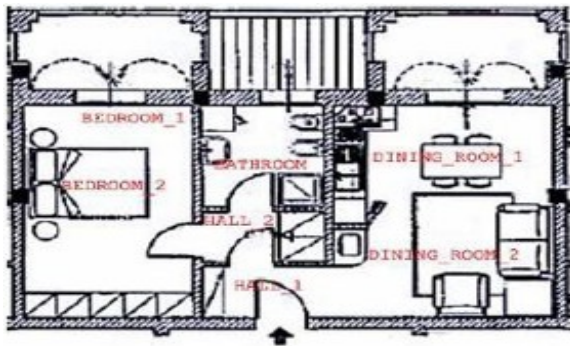
Extending Home Manager

Home Manager 1.0

- ☹️ Access control model
- ☹️ Limited policies
- ☹️ Simple scenario

New Home Manager

- 😊 RBAC Engine
- 😊 Configurable RBAC
- 😊 Larger scenario



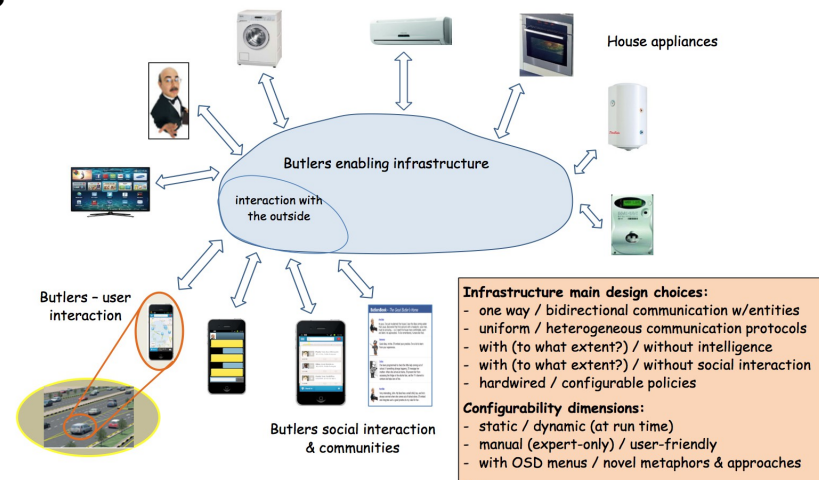
New Objectives

- ◆ Designing an agent-based RBAC engine taking into account
 - ◆ User Role(s)
 - ◆ Presence/State of the room
 - ◆ Hour of the day and Date/Days of week
- ◆ Further stressing the SODA methodology in a richer & larger testbed
- ◆ Testing effectiveness & robustness of TuCSoN in a more complex scenario
- ◆ Laying the foundations for “pervasive” and “ubiquitous” extensions according to the Butlers perspective

The Butlers vision (Denti, 2014)

the intelligent *butler* handles the appliances w.r.t. user goals and configurable policies

anticipates the user's needs by exploiting any kind of user information (position, habits, mood..)





Outline

- ◆ Introduction and context

- ◆ Background

- SODA
- TuCSoN
- RBAC

- ◆ Home Manager: new scenario

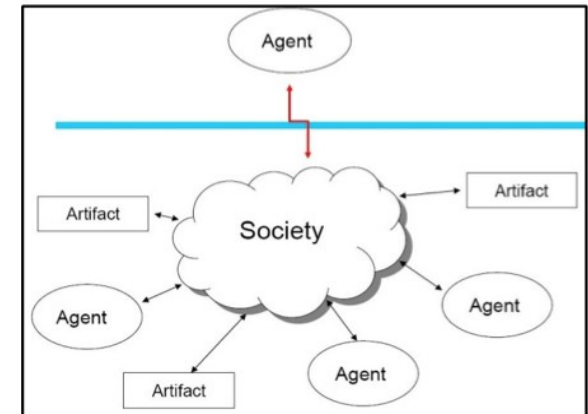
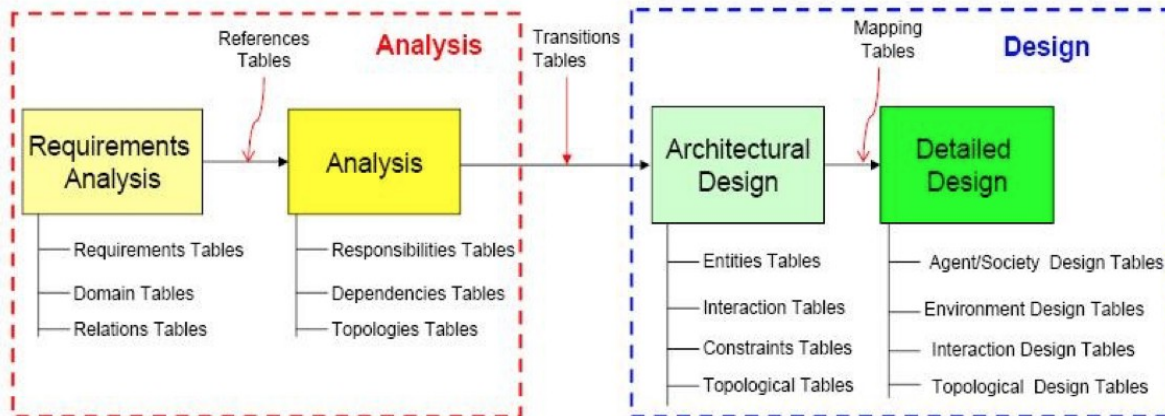
- ◆ Conclusion

The SODA Agent-Oriented Methodology

SODA (Societies in Open and Distributed Agent spaces): AOSE methodology for the analysis and design of *agent-based systems* based on the *Agents & Artifacts* (A&A) meta-model [Molesini et al. 2008]

The **SODA process** is organised in two phases (each made of two sub-phases)

- *Analysis phase* (Requirements Analysis and Analysis steps)
- *Design phase* (Architectural Design and Detailed Design steps)



Main concepts:

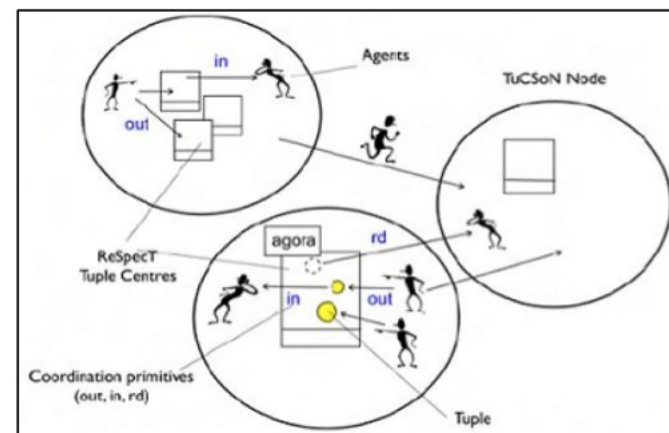
- workspace as an open set of *agents* (*active* part) and *artifacts* (*reactive* part)
- explicit specification on how they *relate* (*interaction* rules)

The TuCSoN model and infrastructure

TuCSoN (Tuple Centres Spread over the Network) is a model and infrastructure for the coordination of distributed processes, as well as of autonomous, intelligent & mobile agents [Omicini and Zambonelli, 1999]

- ♦ TuCSoN agents are the *coordinables*
- ♦ Tuple centres (TC) are the **programmable** coordination media
- ♦ TuCSoN nodes represent the basic *topological abstraction* (host the TCs)

Agents interact by exchanging tuples on tuple centres, via the **TuCSoN coordination primitives**



RBAC for Multi-Agent Systems

- ♦ RBAC is an ANSI/INCITS standard (# 354-2004)
- ♦ *Role* as the key concept for defining and enforcing permissions & rules
- ♦ Policy-neutral, easier to extend, can express DAC/MAC models
- ♦ Supports both static and dynamic *separation of duties* (SSD / DSD)

RBAC-MAS is a model for an RBAC-like organisation of MAS...ref

- RBAC general concepts tailored to MAS specificities
- (and to the SODA process..)

- ♦ Roles, sessions, and policies mapped on MAS *runtime entites*
- ♦ Dynamically management via infrastructural services
- ♦ Clear separation between mechanisms and policies
- ♦ Two sub-systems:
 - ♦ RBAC as Agent Classes,
 - ♦ Role (Agent) behaviour defined as *Actions* and *Perceptions*



Outline

- ◆ Introduction and context
- ◆ Background
- ◆ Home Manager: new scenario
 - Design
 - Logical architecture
 - Implementation
- ◆ Conclusion
- ◆ Future Work

Home Manager: policy design

♦ **An Access Control Model for Home Manager**

♦ “Positive permission” approach

- Roles & permissions

- ♦ Parent the house administrator
- ♦ Son can use any appliance, but has no administrator rights
- ♦ Child restrictions apply on devices and room access
- ♦ Cleaning lady
- ♦ Guest
- ♦ Unknown user an unrecognized person, possibly an intruder)

- Device types

- Ambient constraints for both rooms and appliances

♦ **Specific policies of interest: energy saving & user comfort**

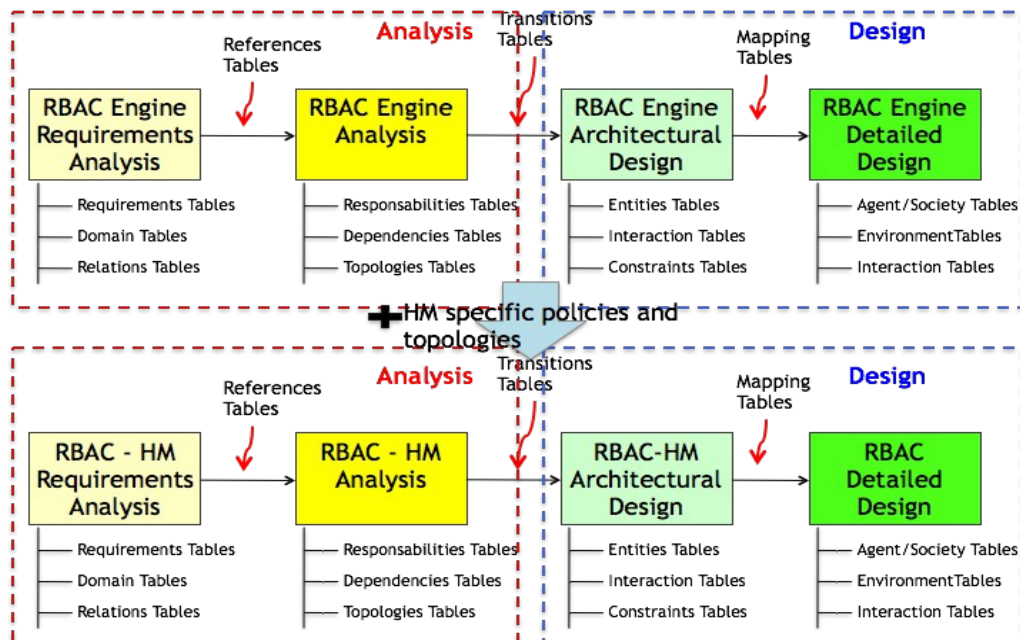
- Environment aspects → lighting & heating
- Essential vs unessential appliances

The (Home Manager) RBAC engine

GOAL: keep the RBAC engine as independent as possible of the HM case

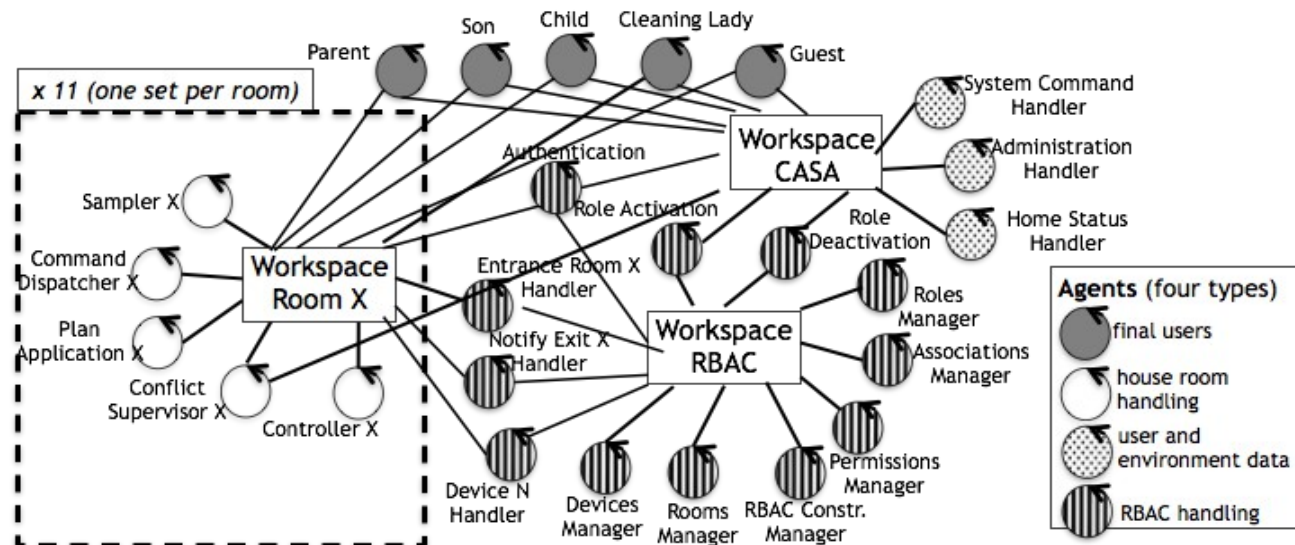
APPROACH: apply the SODA process twice:

- **First**, to design an RBAC control mechanism for a (*generic*) private building
- **Then**, to add the *Home Manager-specific* policies and house topology.



The new Home Manager Architecture

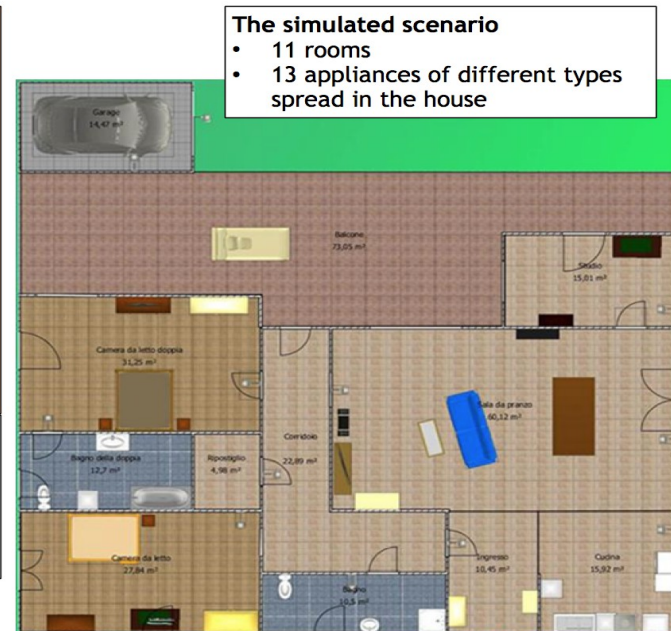
- ◆ **SODA design entities are mapped onto proper TuCSoN abstractions:**
 - Coordination laws and system policies onto suitably programmed tuple centres
 - Environment information: one tuple centre “HOUSE”
 - Data and RBAC policies: one tuple centre “RBAC”
 - One workspace per room, one tuple centre per workspace
 - Four agent types: the three above + final users



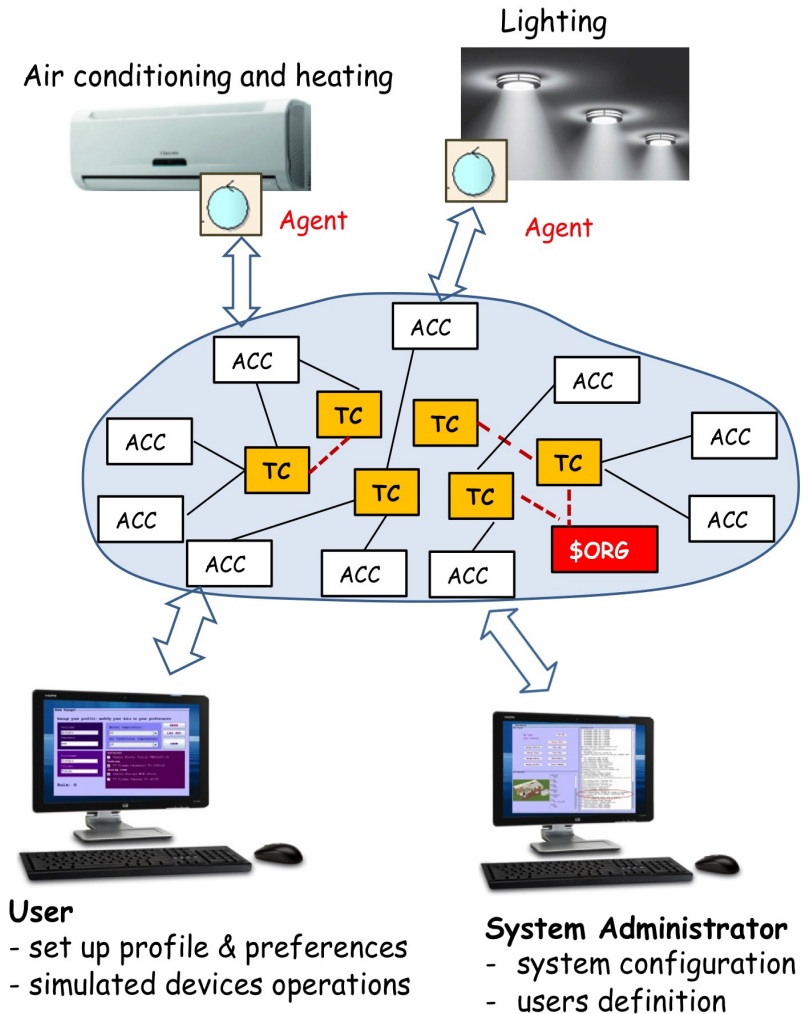
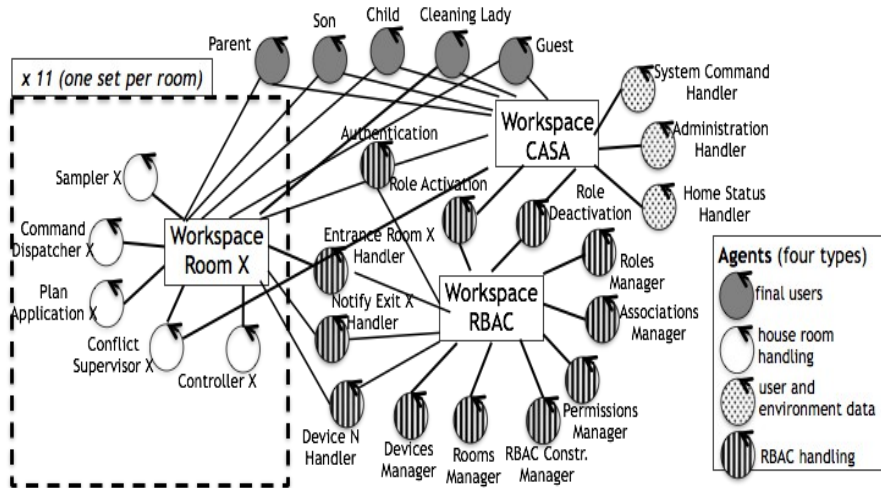
The new scenario in detail

- ◆ Details
 - 11 rooms
 - 13 devices
 - Lighting, Conditioning and Rolling Shutters control
- ◆ Role Based Access Control on appliances and rooms
- ◆ More complex policies for user comfort and energy saving

Roles and permissions - parent, child, baby, cleaning lady, visitor
Appliances classes and constraints - switchable / un-switchable devices (A/C, fridge, ...) - environmental constraints for room and appliances - private bathroom reserved by selected user(s) at given times - TV accessible to children after 8pm only with parental control
Lighting and comfort policies / preferences - user preferences conjugated with house policies - external factors (light, temp) taken into account - conflict management policies
The overall system structure • 13 tuple centres • 4 agent types with dozens of agents • 1 DBMS to store user info and appliance info

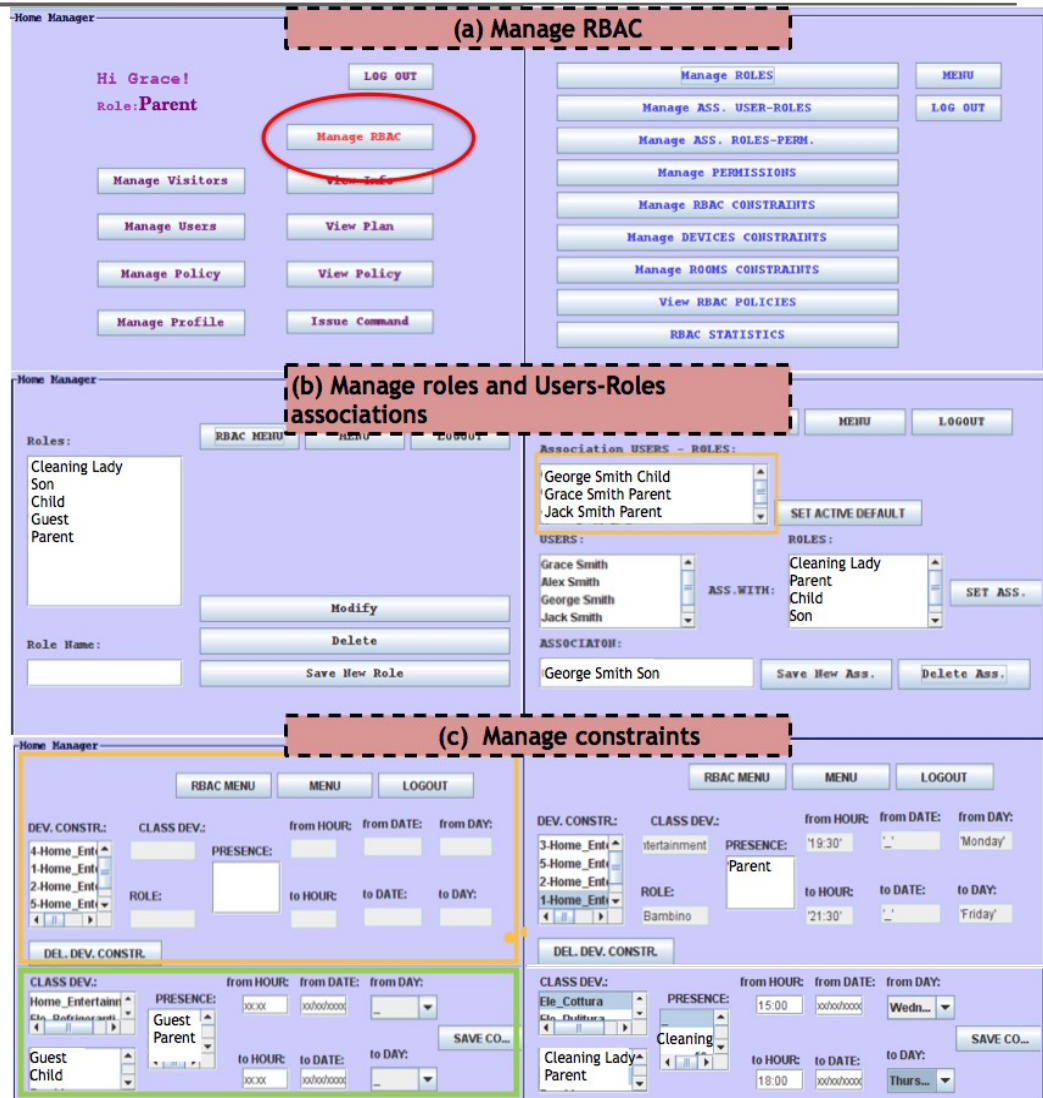


From architecture to implementation



Home Manager: implementation

- ♦ *GUI* to operate on roles, policies, and constraints
- ♦ Configurability
- ♦ No need to know the TuCSoN specification languages



The image displays three screenshots of the Home Manager GUI, illustrating its implementation for managing roles, policies, and constraints.

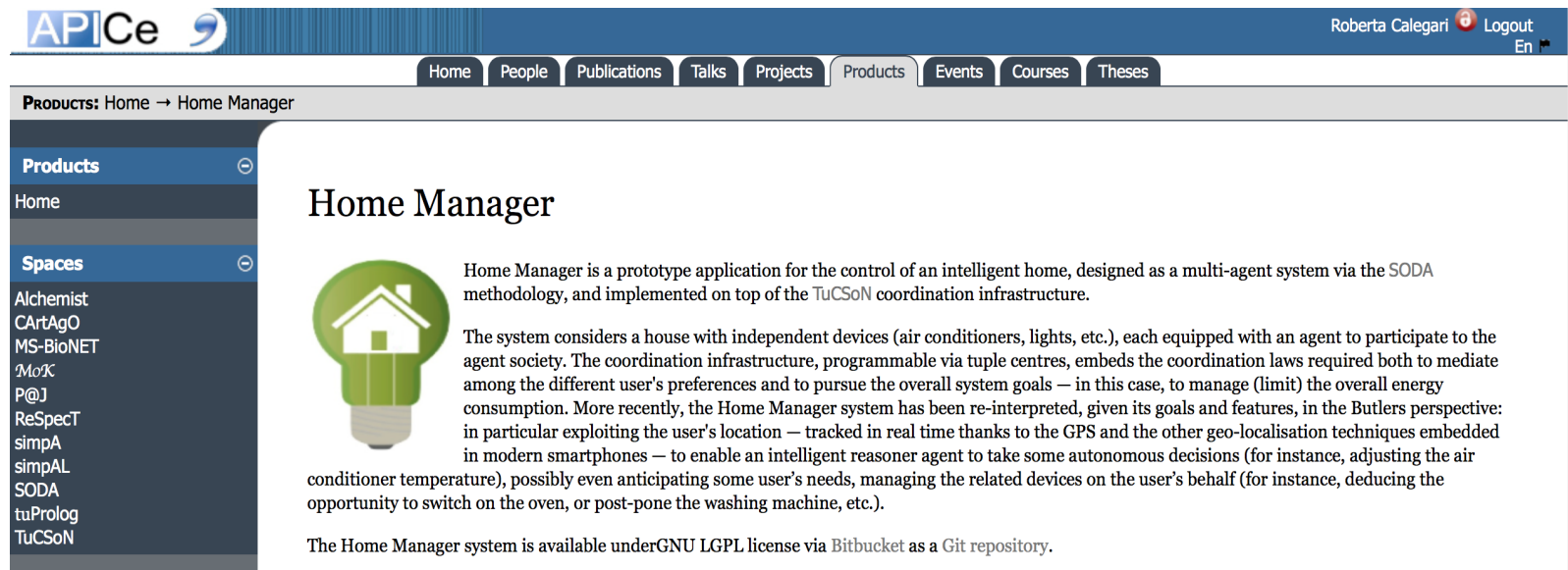
(a) Manage RBAC: This screenshot shows the main dashboard for managing RBAC. It includes a user greeting "Hi Grace! Role: Parent" and a "LOG OUT" button. A red circle highlights the "Manage RBAC" button. The interface is divided into two main sections: "Manage RBAC" (left) and "Manage RBAC" (right). The left section contains buttons for "Manage Visitors", "Manage Users", "Manage Policy", "Manage Profile", "View Info", "View Plan", "View Policy", and "Issue Command". The right section contains buttons for "Manage ROLES", "Manage ASS. USER-ROLES", "Manage ASS. ROLES-PERM.", "Manage PERMISSIONS", "Manage RBAC CONSTRAINTS", "Manage DEVICES CONSTRAINTS", "Manage ROOMS CONSTRAINTS", "View RBAC POLICIES", and "RBAC STATISTICS".

(b) Manage roles and Users-Roles associations: This screenshot shows the "Manage roles and Users-Roles associations" section. It includes a "RBAC MENU" button and a "LOGOUT" button. The "Roles:" list contains "Cleaning Lady", "Son", "Child", "Guest", and "Parent". The "Association USERS - ROLES:" list contains "George Smith Child", "Grace Smith Parent", and "Jack Smith Parent". The "USERS:" list contains "Grace Smith", "Alex Smith", "George Smith", and "Jack Smith". The "ROLES:" list contains "Cleaning Lady", "Parent", "Child", and "Son". The "ASS. WITH:" list contains "George Smith Son". The "ASSOCIATION:" list contains "George Smith Son". The "SET ACTIVE DEFAULT" button is highlighted. The "SET ASS." button is also visible.

(c) Manage constraints: This screenshot shows the "Manage constraints" section. It includes a "RBAC MENU" button and a "LOGOUT" button. The "DEV. CONSTR.:" list contains "4-Home_Ent", "1-Home_Ent", "2-Home_Ent", and "5-Home_Ent". The "CLASS DEV.:" list contains "Home_Entertainment", "Ele_Cottura", "Ele_Distribuzione", and "Guest Parent". The "PRESENCE:" list contains "Guest Parent". The "from HOUR:" list contains "19:30". The "from DATE:" list contains "Monday". The "from DAY:" list contains "Monday". The "to HOUR:" list contains "21:30". The "to DATE:" list contains "Friday". The "to DAY:" list contains "Friday". The "DEL. DEV. CONSTR." button is highlighted. The "SAVE CO..." button is also visible.

Home Manager: home sites

- ◆ <http://apice.unibo.it/xwiki/bin/view/Products/HomeManager>
- ◆ <https://bitbucket.org/tuprologteam/homemanager>



APICe Roberta Calegari Logout En

Home People Publications Talks Projects Products Events Courses Theses

Products: Home → Home Manager

Products

Home

Spaces

Alchemist
CArtAgO
MS-BioNET
MoK
P@J
ReSpecT
simpA
simpAL
SODA
tuProlog
TuCSO

Home Manager



Home Manager is a prototype application for the control of an intelligent home, designed as a multi-agent system via the SODA methodology, and implemented on top of the TuCSO coordination infrastructure.

The system considers a house with independent devices (air conditioners, lights, etc.), each equipped with an agent to participate to the agent society. The coordination infrastructure, programmable via tuple centres, embeds the coordination laws required both to mediate among the different user's preferences and to pursue the overall system goals — in this case, to manage (limit) the overall energy consumption. More recently, the Home Manager system has been re-interpreted, given its goals and features, in the Butlers perspective: in particular exploiting the user's location — tracked in real time thanks to the GPS and the other geo-localisation techniques embedded in modern smartphones — to enable an intelligent reasoner agent to take some autonomous decisions (for instance, adjusting the air conditioner temperature), possibly even anticipating some user's needs, managing the related devices on the user's behalf (for instance, deducing the opportunity to switch on the oven, or post-pone the washing machine, etc.).

The Home Manager system is available under GNU LGPL license via Bitbucket as a Git repository.



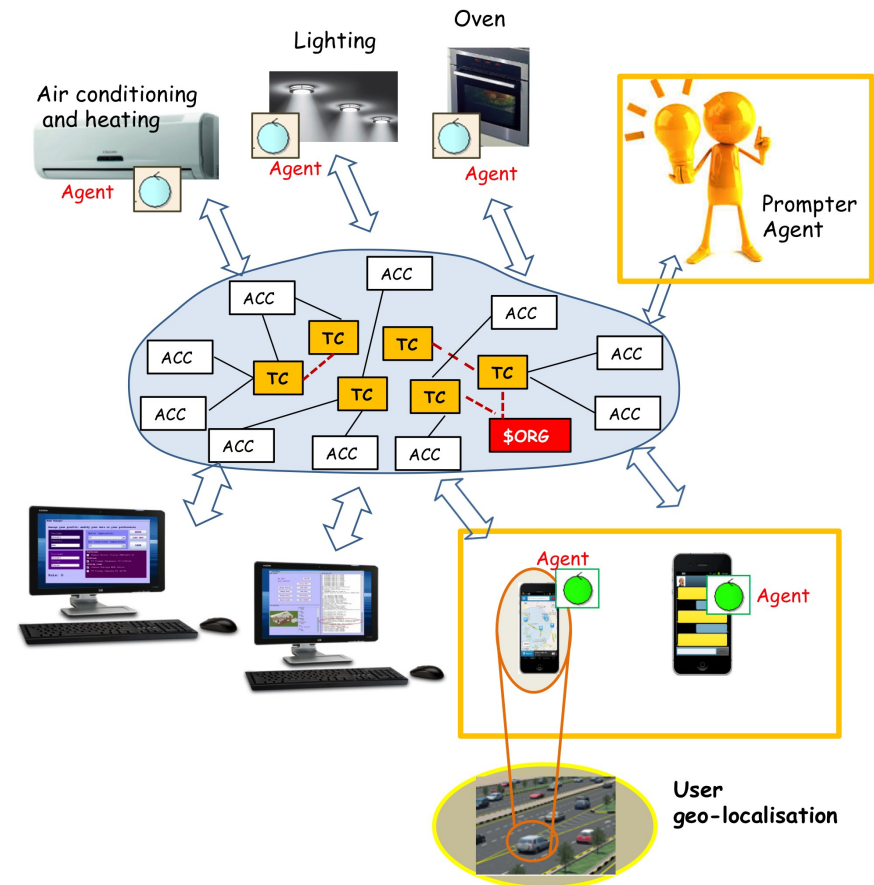
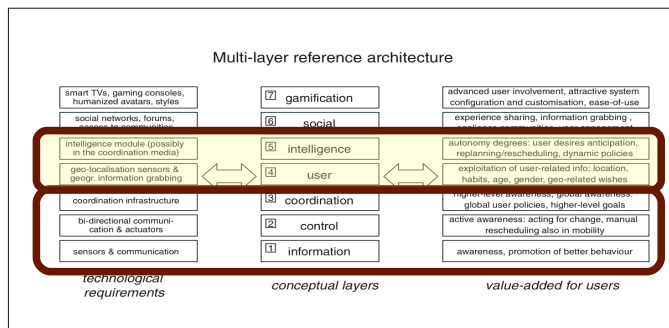
Outline

- ◆ Introduction and context
- ◆ Background
- ◆ Home Manager: new scenario

- ◆ Future Work
- ◆ Conclusion

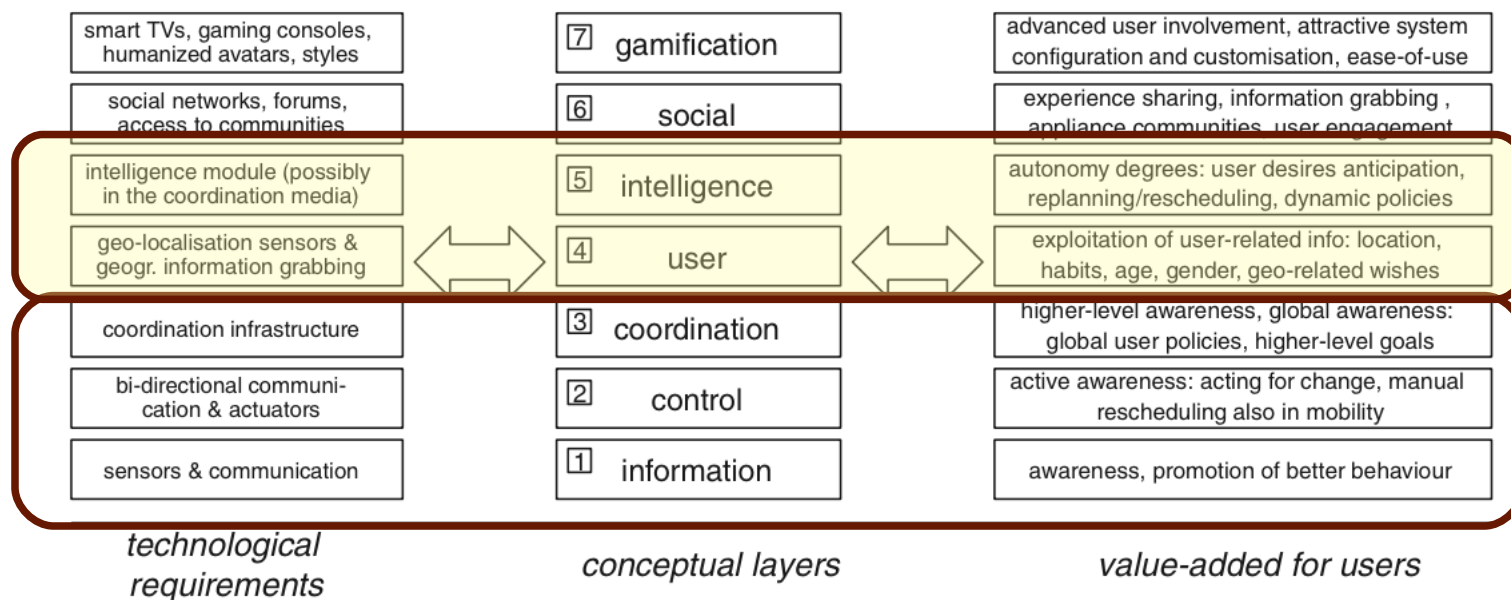
Extension towards **pervasive** and **ubiquitous** contexts

- Remote control capabilities
- Exploitation of the user's profile (habits, personal knowledge, etc.)
- Exploitation of dynamic data (location, situation, etc.)
- Smarter behaviour according to the *Butlers* perspective



Next step: Home Manager “Butler-isation”

Multi-layer reference architecture



Conclusion

- ◆ RBAC engine successfully deployed onto an Aml scenario
- ◆ SODA & TuCSoN pros & cons
 - SODA proved effective, two-phase process also supported
 - TuCSoN proved effectively able to govern a heterogeneous system
 - Huge amount of SODA tables is a critical limit → *tools needed*
 - Tuple centre specification languages out-of-reach for end users → GUI
- ◆ Pervasive extensions
 - Butlers perspective
 - Next step: exploit user's geo-localisation to provide advanced services (anticipate some possible user's needs)



Thank you.

marco.prandini@unibo.it



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA