

Planning in a Smart Home: Visualization and Simulation ^{*}

Alexander Lazovik^{**}, Eirini Kaldeli^{**}, Elena Lazovik^{***}, and Marco Aiello^{**}

{a.lazovik,e.kaldeli,e.lazovik,m.aiello}@rug.nl
University of Groningen
Nijenborgh 9, 9747AG Groningen, The Netherlands

1 Abstract

The area of pervasive systems is characterized by high heterogeneity, with thousands of autonomous devices living together, and requiring high level of interoperation. In particular, domotics is concerned with the pervasion of technology into housing, in order to make homes more pro-active and improve the level of security and comfort of their inhabitants.

In this work, we bring intelligence into smart homes, that is, homes that contain interactive and pro-active devices, and are able to adapt their behavior to the needs of the home inhabitants through extensive interoperation and user interaction. For example, a movie may be automatically paused when the user leaves the room, and then launched again when he/she is back; windows are automatically opened to regulate the air condition or as a reaction to gas leak, and so on. To deal with such complex scenarios we use a planning system that synthesizes plans on-the-fly, based on goals given by the home inhabitants.

2 General architecture

A general architecture of the system is shown in Figure 2. The goal is specified through one of the possible interfaces, be it a brain-computing interface, a mobile device, voice-recognition, or some other piece of software. Given a goal, the planner collects the information about the current state of the house, e.g. the available services and their current states, through the context module, which may possibly prefetch the data for better performance.

A plan that satisfies the specified goal is synthesized, and then passed to the orchestration component which is responsible for the plan execution. To execute a particular action, the orchestration component finds one of the possible services that implement the desired action. Some extra constraints may be associated in this case to reduce possible instantiations. For example, the alarm action may

^{*} This research is supported by the EU through the STREP project FP7-224332 Smart Homes for All, <http://www.sm4all-project.eu>.

^{**} Distributed Systems Group

^{***} Master in Computer Science.

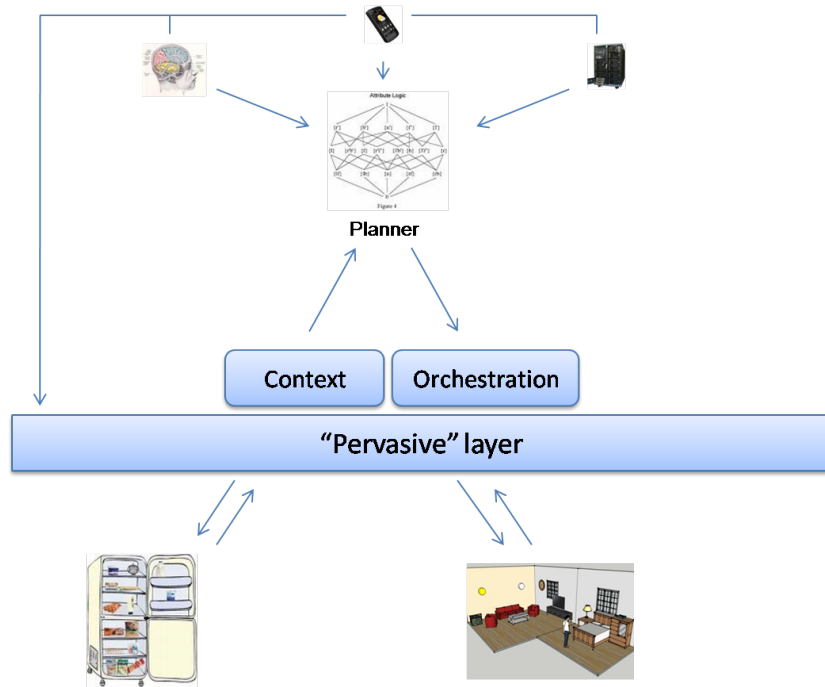


Fig. 1. Global view

instantiate the corresponding implementation which is close to the user, e.g. by showing a message on the TV screen if the user is watching it, or by invoking the alarm of the alarm clock if the user is sleeping in his bed.

The user himself is represented as one of the services at the pervasive layer. That is, whenever an interaction with the user is needed according to the plan, the orchestration component invokes one of the services that represent the user.

Having separate components for context and orchestration allows us to develop planning algorithms in a domain-independent fashion, thus making our work also applicable to a wide range of other domains that require high heterogeneity, e.g. automated web service composition [4].

The actual services (represented either by a visualization and simulation environment, bottom-right in Figure 2, or installed in the actual hardware, bottom-left) are accessed through a separate pervasive layer. The pervasive layer allows us to decouple plan execution from the knowledge where and how the actual services are accessed and invoked. This way, the actual devices installed in a smart home can be easily replaced by services represented in a simulation environment on-the-fly, as well as the other way around. This makes our framework suitable for testing and simulating different smart homes scenarios, since it allows

for a seamless service transition between the actual world and the visualization environment.

3 The planner component

The main goal of the planner component is to provide for a domain-independent planner, with emphasis on extended goals, non-determinism, failure recovery, and incomplete knowledge - features important for real life planning problems such as smart homes. Though the planner is presented in the context of smart homes, it is domain-independent, and its techniques can be applied in a number of other topics with similar requirements, e.g. automated web service composition [4] and, to a less extent, e-government regulations systems [2].

The algorithms we currently use are based on an encoding of planning problems to constraint satisfaction (with Choco¹ as a constraint solver) or boolean satisfiability problems (with SAT4J² as a SAT solver).

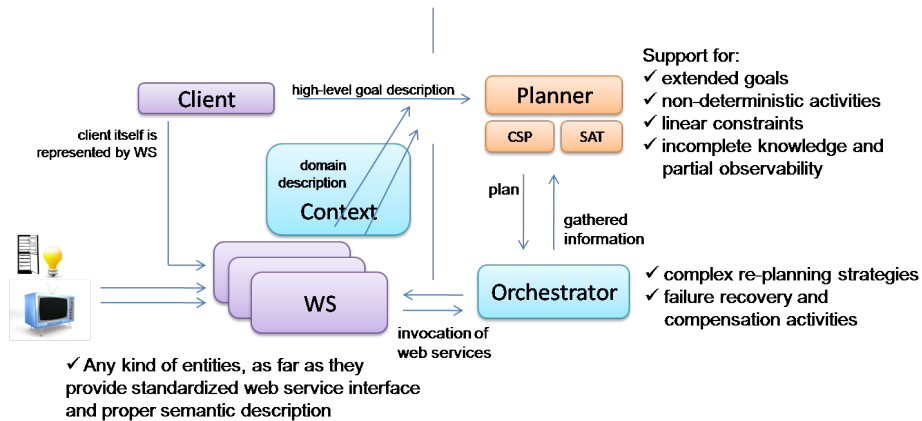


Fig. 2. Planner internal architecture

Figure 3 shows how the planner interacts with other components. It receives the goals from the user, and the domain description from the context module. Then, the goal and the domain are converted into a CSP (or SAT) problem, whose solution corresponds to the synthesized plan. The plan is then given to the orchestration component, which is responsible for its execution.

Sometimes, even valid plans cannot be executed due to some change in the environment, e.g. a person has in the meantime rearranged something in the house, and the orchestration component cannot execute the plan any more. In this case,

¹ <http://choco.sourceforge.net>

² <http://www.sat4j.org>

the plan is given back to the planner for revision. The planner senses again the environment through the context component, performs re-planning, and gives the anew computed plan to the orchestrator. The need for interleaving planning and execution is also associated with the fact that sometimes the context component cannot sense the whole home. For some sensing actions to be realized, a number of other actions have to be executed first, e.g., the video camera cannot record anything at night if the lights are not already turned on. In such a scenario, the system first synthesizes and executes “precondition” actions, and then, when having more information, it proceeds to the actual planning. More information on the planner component may be found in [4, 1].

4 Visualization and simulation environment

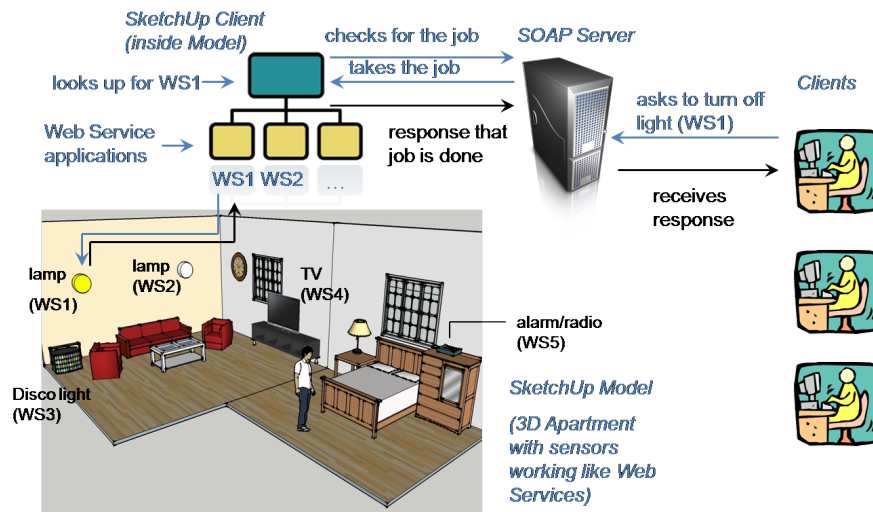


Fig. 3. Sketchup architecture

One of the greatest difficulties in developing middleware for smart homes is that this kind of systems are extremely difficult to verify. Proper testing with a number of different scenarios would require a dedicated physical home, furnished and equipped with expensive devices. We propose to reduce the testing costs by replacing the actual home services with virtual stubs, that behave as if the corresponding hardware was actually installed in the house. The main purpose is to visualize the environment and behavior of a smart home, give the user the impression of being in a real home, and get some useful feedback. To this

Icon	Goal
TV	vital tv is ON and person is AT_TV and lamp1 is ON and lamp2 is ON
Hot	vital livingWnd is ON
Gas leak	vital tv is NOTIFYING and kitchenWnd is ON
Repair	vital gasSensor is OFF and vital tv is PAUSED
Check	vital beer > 0 and fridgeDoor is OFF
Beer	vital beer is AT_PERSON and fridgeDoor is OFF

Table 1. Pre-defined goals

end, we extend Google SketchUp³ with a set of tools that extend its visual representation of a house with virtual interactive home Web Services supporting SOAP messages. This allows us not only to visualize the potential smart home, but also to provide a full featured simulation of any possible domotics scenario. In addition, it is possible to model the user and its interaction with the home. The realized simulation and visualization environment is named ViSi (Smart Home Visualization and Simulation) [3].

The realized simulation environment can be used in conjunction with some actual hardware supporting the Web Service stack. For instance, we have included in our test a controller of a fridge implementing WSDL and SOAP over HTTP on an Ethernet connection. Figure 4 provides an overview of the implementation: the visualization component is written as a set of plug-ins for Google SketchUp, while the communication mechanism is based on a SOAP implementation in Ruby. The clients are language-independent and can be written in any language that supports the Web Service stack. In this demo, the orchestration component transforms the actions computed by the planner into Web Service invocations.

The realized simulation and visualization framework ViSi is not limited to domotics applications. Google SketchUp is a domain-independent drawing tool, and our framework can be used to simulate and visualize application from a number of different areas, e.g. telecommunications distributed systems, network applications etc. In fact, we plan to use the ViSi framework for the energy sector. More information on the visualization and simulation tool is provided in [6].

5 Demo case study

In the demo we have 6 pre-defined goals, which correspond to possible user wishes within the house. Having the goals pre-defined reflects the expected use of the system by users who have no knowledge or abilities to provide correct AI planning goals. For example, if the system is used in hospitals by people that are connected to the system through brain-computing interfaces, a single command represents the corresponding pre-defined goal. Of course, in general, users may specify the goals themselves, if they want.

³ <http://sketchup.google.com>.

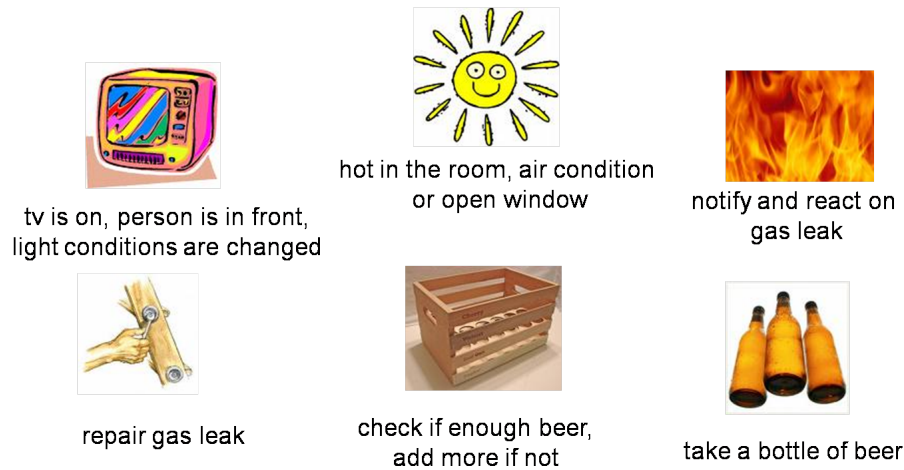


Fig. 4. Sample goals used in demo

The input panel for the demo is shown in Figure 5. Each icon represents one of the goals as defined in Table 1. The last column of the table shows the expected behavior of the system. For more information on the goal semantics please refer to [5, 4].

References

1. RuG planner. <http://www.sas-leg.net/flair.trac/wiki/Planner>.
2. Software As Service for Local eGovernments (SAS-LeG). <http://www.sas-leg.net>.
3. ViSi demo. www.sm4all-project.eu/index.php/activities/videos.html, 2009.
4. E. Kaldeli, A. Lazovik, and M. Aiello. Extended Goals for Composing Services. In *Int. Conf. on Automated Planning and Scheduling (ICAPS-09)*, 2009.
5. A. Lazovik, M. Aiello, and M. Papazoglou. Planning and monitoring the execution of web service requests. *Journal on Digital Libraries*, 2005.
6. E. Lazovik, A.C.T. den Dulk, M. de Groote, A. Lazovik, and M. Aiello. Services inside the Smart Home: A Simulation and Visualization tool. In *Int. Conf. on Service Oriented Computing (ICSOC-09)*, 2009. to appear.