

# Homebird—Task-based User Experience for Home Networks and Smart Spaces

Olli Rantapuska  
Nokia Research Center  
P.O. Box 407  
FI-00045 Nokia Group

olli.rantapuska@nokia.com

Mia Lähteenmäki  
Nokia Research Center  
P.O. Box 407  
FI-00045 Nokia Group

mia.lahteenmaki@nokia.com

## ABSTRACT

Contemporary wireless networks in people's homes are already enabling consumer electronics devices to communicate with each other. Standards like Universal Plug and Play are being developed for interoperability between devices from different manufacturers. For example, a digital media player device is able to display video clips from a home PC or play music from portable devices. Development of the user experience is also needed to have devices perform tasks in concert. Homebird is a demonstration of a task-based user experience on a mobile phone. It discovers features of other devices automatically and suggests to the user that certain tasks can be performed together with those devices. This approach cuts down the number of steps needed to perform common tasks, and also makes it easier for users to find out what can be done in a particular environment. The implementation architecture makes it easy to add new tasks, and they can also have the phone perform actions in the background without user interaction. The task-based approach was evaluated with a small user study, and participants found it easy to understand and useful, if they were offered tasks that suit their daily life.

## Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces – *interaction styles, prototyping*. C.5.3 [Computer System Implementation]: Microcomputers – *portable devices*. J.7 [Computers in Other Systems]: *consumer products*.

## General Terms

Design, Experimentation, Human Factors

## Keywords

Task, remote control, mobile phone, UPnP, WLAN.

## 1. INTRODUCTION

Wireless local area networks are becoming prevalent in people's homes. A home network makes it possible to share content between computing and consumer electronics devices. The network acts as a communications backbone, removing the need to connect devices together individually. Content, such as photos, movies or even cooking recipes, can be moved around and processed by devices that support the type of content and the necessary protocols for handling it. There are also generic protocols like Universal Plug and Play for handling abstract pieces of content.

Consumer electronics devices are rapidly becoming networked. Stereo systems can download music from the Internet and play it on speakers in the next room, printers can include a scanner and they all work on a wireless network. The selection of networked home devices is expected to further broaden in the coming years.

With a multitude of devices and an expectation of interoperability comes a difficult challenge. Somehow, people at home would need to get information about what they can do with their selection of devices, and how those tasks can be controlled. Not only would several remote controls be needed to communicate with the devices, but they would also need to refer to other devices to complete tasks. People have mobile phones nowadays, and they would like to use their phone as a remote control—out of 10 most popular mobile application wishes, three were universal remote controls [7]. The call is open for an intuitive remote control interface for home networks and smart spaces.

## 2. PROBLEM AREA OF REMOTE CONTROL

It's straightforward to design a remote control for a single device and its features, but controlling a set of devices in concert is more challenging. There are already various universal remote control devices on the market. For example, the Logitech Harmony line [4] offers good flexibility in controlling many devices at the same time. However, universal remotes for the most part still need to be pre-configured with a certain set of actions that can be performed. In an interoperable networked home, new actions can become available dynamically. It should be up to the remote control to discover these actions, not the user. With bidirectional communication enabled through WLAN, remote control devices will be better equipped to make intelligent decisions on the user's behalf. Of course, interoperability depends on individual device manufacturers implementing the necessary control interfaces, which are generally not a key selling point for devices.

The home network consists of physical devices, but that does not necessarily need to be the user experience paradigm. A remote control can combine the features of the devices together and provide distilled, more meaningful information to the end user. One approach is a task-based system, where a single task that the user engages in can span multiple devices. For example, a task to view a DVD movie would involve at least a TV and a DVD player device. These tasks would be personalized for each user, and would be generic enough to avoid being bound to individual devices. The user would not need to think about devices; he would only need to interact with higher-level tasks.

The task-based paradigm has been the subject of some existing work. In [2] the authors found that it is possible to predict the next probable task based on history of the users' activity, and that in fact it is easier to predict the next task than the next device being used. In their work, a task is defined as a set of related commands to devices, and the aim is to group those commands in a meaningful way so that they can be easily used together as one task. Assuming individual devices provide broad enough bidirectional APIs, it is possible to use commonsense reasoning to suggest tasks, or goals, automatically as described in [3]. Their Roadie concept allows people to execute remote control actions starting from the goal that they want to accomplish. A small user evaluation indicated that working with Roadie required less clicks and took less time than controlling the devices without it.

People could also be allowed to customize tasks that the system can perform—a simple user interface would let the end user connect the dots to form new tasks. In [6], the Huddle system lets end users connect devices together in a user interface centered on content flows. For advanced users, a high level programming environment could be offered for developing tasks—in the TERESA tool [5], developers are able to concentrate on the conceptual task model and a user interface can be automatically generated for different platforms and devices. The same task can have a different presentation depending on the platform.

Eventually, tasks could be defined both by machine learning and by users themselves. As the authors found in [8], neither approach is optimal by itself. People forget to include every detail defining the tasks manually, and machine learning takes a long time to converge to an accurate model. By using these approaches to augment each other, however, better results could be achieved.

In previous work, there has been little focus on studying how end users actually perceive the task-based approach to remote control, and what kind of tasks would be appropriate for their home environment. This paper extends the work on task-based interaction with a study on end users and a demonstration implementation using a bidirectional control protocol such as Universal Plug and Play (UPnP) over WLAN.

### 3. HOMEBIRD: TASK-BASED REMOTE CONTROL ON A MOBILE PHONE

We set out to experiment with the task-based paradigm, and proceeded to develop *Homebird*, a prototype implementation of a user experience based on tasks. Homebird runs on a customized Nokia N95 smart phone, and can be used in home WLAN networks. It is implemented using Symbian C++.

The task-based implementation takes a step back from a list of devices and instead presents currently available tasks to the user. We define tasks as high-level goals the user wants to accomplish, in most cases making use of several devices simultaneously. The presentation of tasks allows them to change in response to arbitrary events—for example, a newly discovered network device can trigger changes to the set of available tasks. The user can activate each task by selecting it, soliciting the phone to perform that particular task. An example task is *Show latest photos*. Each task has logic behind it that describes what the task does—the Show latest photos task displays photos on a nearby TV screen. It is designed to (1) search the network for photos, and (2) search the network for TV screens capable of showing the

photos. The phone performs these actions in the background without user interaction. If the items in both (1) and (2) are found, only then does Show latest photos appear in the presentation of tasks, offering the user the possibility to activate it. When the user selects that task, the phone retrieves the photos it previously found and sends them to the TV it found. A particular TV screen can be selected in case many were discovered. Additionally, the phone displays a new user interface for switching to the next and previous photo, essentially letting the user refine the task by providing more information.

In Homebird, the logic of tasks is encapsulated in modules called plug-ins. The plug-ins are dynamically loaded libraries that adhere to a defined plug-in API. The Homebird framework loads these plug-ins when the phone is switched on, and lets them perform actions in the background. Homebird notifies plug-ins if the phone is in coverage of a known WLAN. This way, it is possible to save memory and improve battery life by shutting down plug-ins when the context is not relevant for them—the plug-ins are meant to run inside the coverage of the home WLAN.

The Homebird architecture is shown in Figure 1. The framework starts and manages plug-ins that can be written separately. Each plug-in can have background processes that it uses to perform inquiries without user interaction. Each plug-in can produce one or more tasks that appear on the Homebird user interface. When a task is selected in the UI, control is handed back to the respective plug-in that can then show its own UI customized for that task.

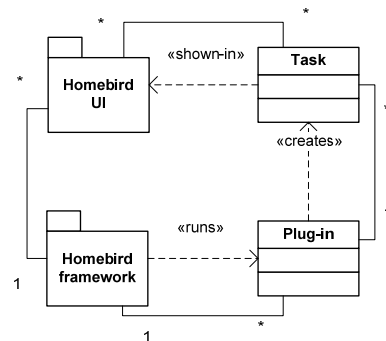


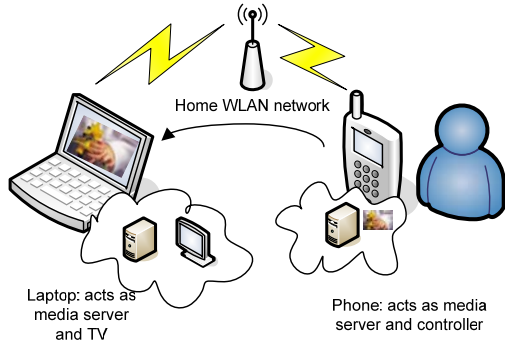
Figure 1: Homebird high-level architecture inside the phone

The demonstration includes three plug-ins: *Show latest photos*, *Show Flickr photos*, and *Adjust lights*. Each plug-in waits for a WLAN network to be found, and then performs some background checking on that network, deciding whether to display a task for the user. All the plug-ins use the UPnP protocol for communicating with other networked devices: media servers, TVs, and networked light controllers. These are standard UPnP device APIs with no changes needed to enable task-based control, so the same task can work with devices from different manufacturers. The demo setup is shown in Figure 2.

The plug-in for adjusting lights shows how the tasks can hide the device paradigm from the user. The Adjust lights task controls all the light devices in the network at the same time. The user only sees one slider control that adjusts all the lights.

The Flickr plug-in uses the Atom protocol to download photos from the Flickr photo-sharing service [1]. It allows users to

browse through new photos from their subscribed Flickr users, and select a photo to view it on a nearby TV.



**Figure 2: Demo setup for showing photos on a TV device, simulated by a laptop**

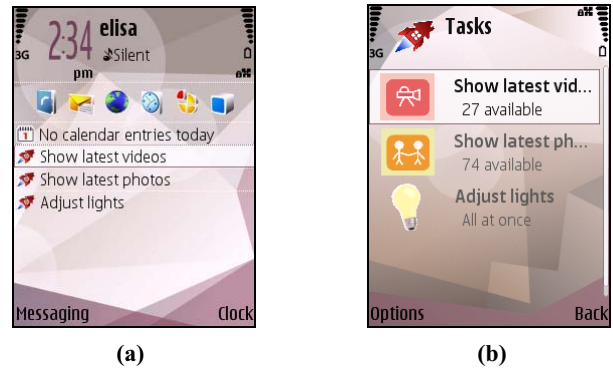
Once the task has appeared, the user can select it to e.g. start showing photos on the TV. The tasks can be presented in different views, two of which are depicted in Figure 3. In the figure, a list of tasks is either visible all the time on the phone standby screen, or only visible when a particular application is started.

#### 4. USER STUDY AND DISCUSSION

We wanted to study how people perceive the use of the phone standby screen for presenting information about their home and the available tasks. Specifically, tasks that become available in the current context would appear on the standby screen to get people's attention. We also wanted to understand their experiences on the corresponding tasks in their current life and their priorities there in order to be able to provide feedback for further development. To that end, we conducted a small user study. Seven participants were recruited in Finland—people who had previous experience with smart phones and who lived in houses, i.e. had their own property to maintain. Each participant was invited to a one-and-half hour session that consisted of a semi-structured interview and a few assignments. The participants were interviewed about their current practices in selected situations, and were afterwards introduced the idea of having the task-based user interface on the mobile device available for the related tasks. For each task, three alternative task-based UI presentations in the form of paper prototypes were presented and discussed. Finally, participants were asked to prioritize some potential tasks for automation based on their personal importance and frequency of use. We specifically focused on three use cases, selected based on our understanding of the main advantages of the task-based paradigm: (1) leaving home for an extended period of time, (2) finding out what the mobile phone as a remote control can do, (3) scheduling a TV recording remotely on your mobile device. These use cases for home control are meant to be more futuristic than the actual Homebird demonstration, which relies on existing standards and interfaces.

In the user study, we found that participants understood the task-based paradigm easily, and were able to discuss using it in their daily routines. People preferred the UI option where tasks appeared on the standby screen on the phone (Figure 3a), but it became clear that they also wanted to be able to customize which tasks appear there. The relative importance of particular tasks

varied from person to person, and some people did not like cluttering their standby screen with these items. Many felt it important to also make sure that if the phone is stolen, the perpetrator is not able to use the phone to remotely hack into the phone owner's smart home.



**Figure 3: Two example views to the available tasks**

For use case (1), we offered a task to check with one click that everything is secure when you leave home. People said they generally go round the house to check that electrical devices are switched off before they leave. They especially wished that the phone would automatically confirm that everything is secure as they close the front door and walk to the car. The task was also seen useful for peace-of-mind, because it could also be activated remotely.

With use case (2), people indicated that they often just browse the device menus to learn what features it has. We envisioned that with smart homes, a tutorial application would be used to learn about features that are available using several devices at once. In practice the people had rarely used the tutorials of any applications on either mobile phones or other computers. The more popular approach seemed to be to solve the problems as they come up. Any introductory tutorial would have to be readily available to remind the user to take a look at the tutorial.

For use case (3), scheduling TV recordings remotely, we asked how participants would like to initiate a remote connection to the home in order to perform the task. We found that people we interviewed prefer not to explicitly start a network connection to the home. Instead, they would like most to just tell the phone to schedule a recording and it should automatically happen at home. This calls for a user experience where the phone offers the user a selection of tasks that could be done, and the user merely needs to activate one of the tasks to do it. This approach comprises a technical challenge: since the tasks have to be cached in order for them to be available also outside the home network, it is not guaranteed that they are really available at the time. For example, the relevant devices may be switched off at home, and that predicament is only discovered after the network connection to the home is started. The user interface should take this into account, perhaps by indicating remote tasks as being only prospectively available.

We also asked the participants to categorize a set of example tasks according to importance in their environment. From that categorization, some tasks like *Secure empty home*, *Check if home is OK*, *View calendar* and *View shopping list* stood out as most

important, whereas e.g. *Show latest video clips [on TV]* or *View phone screen on TV* did not strike people as being very important. The participants also saw that they could select the relevant tasks for themselves from a comprehensive list, and try out which tasks work for them. Other than that, additional customization of the tasks was seen as too much work.

The user study participants gave us feedback on the overall concept. They indicated that the task-based solution would be suitable for them, assuming they would have tasks that work in their home and are able to provide useful service. In general, the advantage of a task-based approach is the minimal user interaction needed to perform complex tasks. With one or two clicks the user can achieve what earlier took almost ten clicks. Instead of having to select a TV device, a media server device, particular photos on a media server device and then starting the process of showing them on the TV, the user only needs one click to show photos. Once the photo viewing has started, the user can fine tune the experience by selecting what photos to show next. The user gets instant gratification because the feature “just works”, but still more complex tasks remain possible by continuing the task through its own UI.

Homebird is an example of a remote control for future homes, where devices are networked and bi-directional communication is available. Instead of having to manually configure infrared settings for a universal remote control, the remote control itself can detect what is available and suggest available tasks automatically. The home smart space is brought to the user instead of the user having to create it through configuration.

In addition to making it easier for end users, Homebird also aims to make it easier for software developers to add functionality to be performed in the home network. By creating a Homebird plug-in, the developer does not have to care about WLAN detection or running his application in the background, because the framework provides both of these features. Moreover, plug-ins are able to use the Homebird user interface to present tasks to the user. The tasks are useful for notifying the user of new items—they can be displayed on the phone standby screen, which gives the plug-in valuable screen real estate without being overly intrusive.

The weakness of a task-based approach is the difficulty of creating the logic that formulates tasks based on features of several discovered devices. In Homebird, the logic is encapsulated inside plug-ins. For simple use cases like media playback, it is possible to develop a generic plug-in that works with all standards compliant media devices. These use cases can also readily be formulated as tasks. In the future, work is needed to refine that capability with more ways to personalize the list of tasks. Automated or manual methods of defining tasks, as seen in existing work, may be integrated to the system and subsequently trialed with end users.

## 5. CONCLUSION

The Homebird demonstration provides an example of a task-based user experience for home networks, running on a mobile phone. It

uses WLAN and UPnP to communicate with other devices in the network, and suggests things to do with those devices in the form of high-level tasks. The user can perform these tasks with one click of a button—their phone has done all the preliminary work already in the background. The task-based approach is easy to understand. A small user study indicates that people would like to personalize how the tasks are presented—this is an area of further work. In the end, interoperability between consumer electronics devices depends on device manufacturers. We hope to make Homebird publicly available and so contribute to a better user experience of smart homes.

## 6. REFERENCES

- [1] Flickr photo sharing service, <http://www.flickr.com/>
- [2] Isbell, C. L., Omojokun, O. and Pierce, J. S. 2004. From devices to tasks: automatic task prediction for personalized appliance control. In *Personal and Ubiquitous Computing*, Volume 8, Issue 3-4 (July 2004). Springer-Verlag, London, UK, 146-153. DOI=<http://dx.doi.org/10.1007/s00779-004-0273-z>
- [3] Lieberman, H. and Espinosa, J. 2006. A goal-oriented interface to consumer electronics using planning and commonsense reasoning. In *Proceedings of the 11th International Conference on Intelligent User Interfaces* (Sydney, Australia, January 29 – February 1, 2006). IUI'06. ACM Press, New York, NY, 226-233. DOI=<http://doi.acm.org/10.1145/1111449.1111497>
- [4] Logitech. Universal Remotes. [http://www.logitech.com/index.cfm/remotes/universal\\_remotes/&cl=roeu.en](http://www.logitech.com/index.cfm/remotes/universal_remotes/&cl=roeu.en)
- [5] Mori, G., Paternò, F. and Santoro, C. 2004. Design and Development of Multidevice User Interfaces through Multiple Logical Descriptions. In *IEEE Transactions on Software Engineering*, Volume 30, Issue 8 (August 2004). IEEE Computer Society, 507-520. DOI=<http://dx.doi.org/10.1109/TSE.2004.40>
- [6] Nichols, J., Rothrock, B., Chau, D. H. and Myers, B. A. 2006. Huddle: Automatically Generating Interfaces for Systems of Multiple Connected Appliances. In *Proceedings of the 19th annual ACM symposium on User interface software and technology* (Montreux, Switzerland, October 15 – 18, 2006). UIST'06. ACM Press, New York, NY, 279-288. DOI=<http://doi.acm.org/10.1145/1166253.1166298>
- [7] Nokia, S60 Application Wishlist, <http://www.s60.com/applicationwishlist> [referred Mar 18, 2008]
- [8] Omojokun, O., Pierce, J. S., Isbell, C. L. and Dewan, P. 2006. Comparing end-user and intelligent remote control interface generation. In *Personal and Ubiquitous Computing*, Volume 10, Issue 2-3 (January 2006). Springer-Verlag, London, UK, 136-143. DOI=<http://dx.doi.org/10.1007/s00779-005-0019-6>