Smart Home Challenges and Approaches to Solve Them: A Practical Industrial Perspective

Roland Eckl and Asa MacWilliams

Siemens AG, Corporate Technology, Software Architecture,
Otto-Hahn-Ring 6, 81739 Munich, Germany
{Eckl.Roland,Asa.MacWilliams}@Siemens.com

Abstract. Why are smart homes not successfully deployed on the market, even though there are many demonstrator systems in research and industry? Over the past years, we have built and improved one smart home platform ourselves and have been involved with many other comparable systems. In this paper, we share the problems we have come across, and present pragmatic approaches towards solving them.

Current smart homes tend to be too simplistic (limited in their application) or too complex (too hard to use or build or maintain). Many research-oriented systems disregard users' intent and make decisions for them, trying to "know better." Also, current systems require too high up-front investment in technology from owners, manufacturers and possible operators.

We suggest using a system architecture with centralized but subsidiary controllers, with open interfaces at software and network levels. Interaction should be based on different types of highly responsive control devices. Besides traditional interaction such as turning on lights, they should let the user select complex scenes, with unobtrusive assistance (prioritizing the presentation of scenes according to context).

We foresee the greatest chance for commercial success if market players work together to create the most valuable applications for end users, but keep their systems open for extension.

Keywords: Smart Homes, Pervasive Computing, Software Architecture.

1 Introduction

With the ever-increasing number of smart home demonstrators in research and industry, we would expect a greater market penetration of smart home technology. However, adoption of smart homes – by consumers and by manufacturers – has remained slow over the last years. Why is this so? In this paper, we analyze several of the problems that we believe make it hard to build successful smart homes, and suggest several possible solutions from our practical industrial perspective.

D. Tavangarian et al. (Eds.): IMC 2009, CCIS 53, pp. 119–130, 2009.

1.1 Our Background

As a corporate R&D group specializing in software architectures for pervasive computing, we have been involved in several real-world smart home installations and public-private research partnerships. One example is *ePerSpace* [1], a project funded by the European Union, which developed a highly extensible network-oriented service architecture to deliver multimedia smart home services to the residential and to the mobile user. Another is the *T-Com House* [2], a collaboration with Deutsche Telekom which involved building and equipping a single-family house in Berlin, which was occupied by test families for a year and half. At the same time, we have maintained two smart home integration laboratories at our Munich and Beijing research sites. From one project to another, we have re-used technologies and maintained hardware and software platforms for smart home devices. These projects gave us many insights and helpful feedback on user needs and technical stumbling blocks in such a complex evolving system.

2 State of the Art: Current Challenges

As we see it, current smart homes fall into three categories, each with its specific challenges (see Figure 1). On the one hand, there are optimized special-purpose niches. These are often *too simplistic* to be widely adopted in market. On the other hand, there are large platforms, which could support many different applications – but those are *too complex* – too hard to use, install or maintain. (We plead guilty to having designed

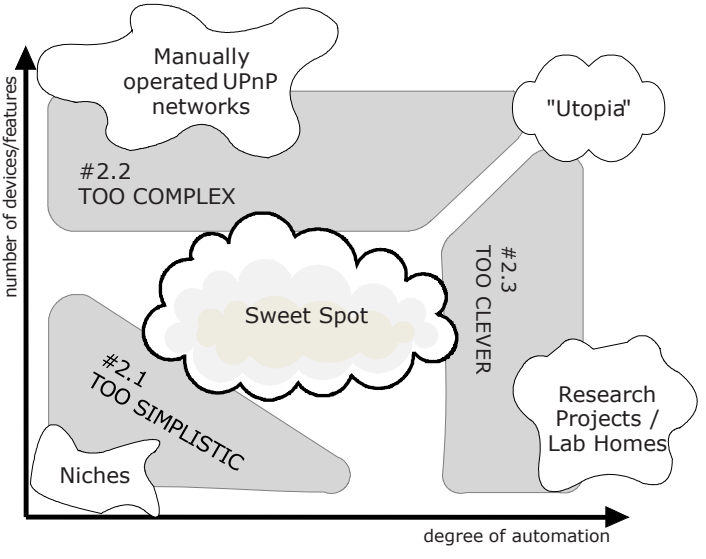

Fig. 1. Landscape of smart home solutions. Most current systems are too simplistic, too complex, or too clever, as described in sections 2.1-2.3

such systems ourselves.) As a third class, there are many research-oriented systems that disregard users' intent and make decisions for them, trying to "know better". These systems are *too clever* and fail to gain the user's trust. We believe that any system that tried to fully integrate all possible devices in a fully automatic fashion (labelled *Utopia* in the figure), would be *both* too complex and too clever for real use. Somewhere between all these mentioned systems lies a *sweet spot* – systems that users will actually find useful.

Almost all systems require *too high up-front investment costs* in technology from owners, manufacturers and operators. We will examine these challenges one by one.

2.1 Too Simplistic: Limited Application Scope

Especially many of the industrial smart home projects are too limited in their scope and serve specialized niches. For example, the serve@Home [3] system offered a powerful, commercially available remote control and monitoring solution for home appliances (refrigerators, dishwashers, ovens, washing machines). However, extending this system to other crafts (e.g. lighting, entertainment) was not part of any commercially available package. serve@Home was discontinued after several years.

Another class of smart home research projects focuses on optimal user interaction or algorithms of behaviour inference. Results are often very promising. Regarding wider dissemination, the tone is, "we demonstrated this on basis of these selected devices. It could easily be extended to any other." This sounds good in theory, but proves very difficult in practice.

Why are the results of these projects hard to transfer into large-scale real-world adoption? One problem is that they often assemble various devices and sensors into complex but inseparable units – extending the devices or replacing parts is difficult. Will prediction algorithms for adaptive environments, such as the MavHome [4], still work properly, or do they require some additional calibration or learning phase? Another is that they chose communication media and protocols that are optimal for a specific application or class of devices (e.g. Powerline for serve@Home), but then aren't extensible to others (e.g. Powerline for mobile devices).

2.2 Too Complex: Too Hard to Use, Build or Maintain

The second class of smart home research projects comes from the area of large distributed systems, e.g. hosted by network operators. One example is ePerSpace, which we participated in ourselves. These systems, as we are well aware, are often too general to be of real practical use. The tone is, "our system is so extensible that we will be able to integrate any kind of application in the future. Just describe these 250 parameters in XML." This leads to systems that are, for any specific application, too hard to use, to build or to maintain – so far such systems remained in Utopia or restricted themselves to a limited scope as described above.

Too Hard to Use

The key challenge to consumer acceptance (besides price) is usability. An interface has to be as simple as the classic light switch, but of course more powerful. Within an

extendible system the connected user interface can quickly become inherently complex, especially if the user should be able to control everything.

We have developed interfaces for full control (e.g. on a PDA or smart phone), and many users were intimidated. Some of our other approaches concealed parts of the complexity and were more popular with lab visitors.

An example where specifically tailored solutions can have higher usability than general ones is multimedia control. Multi-room media streaming solutions, e.g. from Sonos, Philips, or Apple, are optimized for their purpose, although they cannot control other devices such as lights.

One key factor for usability is responsiveness. Commands must have immediate effects, and immediate feedback on the interaction device. System architectures that involve several layers of marshalling, converting and dispatching events in a distributed network are at a challenge here (we built one of those).

Too Hard to Build

The more device types are included and the more a system is capable of, the more heavyweight it will become. The architecture of such a generic system becomes increasingly difficult for developers, making systems hard to build. Also, such generic systems consume more resources (production cost, electricity) than specialized ones. For example, adding even a simple service (a TV program guide) to the ePerSpace service management platform required hundreds of lines of hand-written XML.

Sometimes these hurdles are sourced out to gateways and bridges, which reduce complexity to a common denominator, in most cases according to protocol or service type. But systems that abstract all their devices in this fashion only defer problems to the gateways; they do not solve them.

The KNX electrical installation standard [5] is successful in commercial buildings, but not yet so in residential markets. Besides the price of the devices, commissioning costs are an issue. Programming KNX devices (which can be programmed to do almost anything) requires an expensive software tool, ETS, and a professional training in how to use it. Not many professional electrical installers are trained yet.

Too Hard to Maintain

The more complex a system is, the harder is it to maintain. In the KNX example, ETS is good for protecting the market share of trained electrical installers, but bad for end users who want to reconfigure their smart home themselves. These users would go for a system such as Peha Easyclick [6], which is less powerful, but more easily configured. Or for a media system based on DLNA [7], which does not leave much space for frustrating configuration, but is restricted to its specific field of application.

2.3 Too Clever: Trying to Know Better Than the User

Many research systems address the issue of inferring the user's intent and attempt to control devices accordingly, by use of sensors, rules, and learning techniques. As an example, the Neural Network House [8] claims to essentially *program itself*. This is done by observing the inhabitants and anticipating and accommodating their needs. Discomfort has to be expressed by overriding the environment's choice.

If such a system achieves 100% accuracy, it's wonderful. But when it doesn't (and no system does), users want to know why it is acting the way it is. This can be further complicated by conflicting strategies, e.g. saving energy and meeting the users' intention. "Smart" is fine for users, but "haunted" is not. In our experience, trying to infer the user's intent just leads to frustrated users. This effect has been observed before in other application domains, such as with Microsoft's Office Assistant [9].

Some problems do not occur until the experimental system leaves the lab. Most prediction systems are limited to a single domain. Others suffer from different calibration techniques and potential error rates. It is hard enough to transform original data from different research groups to a common basis – and this gets worse when data comes from different types of devices. Most researchers use only identical devices when testing their systems, neglecting real-world heterogeneous systems. But the less reliable and comparable a prediction system, the worse the result is, which leads to poor user experience. Thus, most users face prediction systems very critically.

2.4 High Up-Front Investment Costs

Users, manufacturers and operators want value for their money from smart home systems, and they want it quickly. But currently, up-front investments are prohibitive.

There are several cases of companies discontinuing their smart home activities. serve@Home was described above. More recently, in May 2009, Nokia discontinued its own smart home activities and licensed the previously developed Home Control Center technology to *There Corporation* [10].

Although reasons for these product discontinuations are rarely given, we suspect the high up-front investment costs for the companies as the root cause.

Investing in a System Architecture

For a system to be successful (not too simplistic), it has to be extensible. But up-front investment in system architecture for this and creating standards is expensive. Companies (both manufacturers and operators) will shy away from this investment until their own prospective market share is promising enough. But business models are still unclear at the moment, and these determine how a system has to be designed with regard to its extensibility and its ability to integrate further crafts or services.

Architectures differ according to who may extend a smart home's capability; the user, an operator, both or others? This covers considerations about frameworks such as OSGi [11] for dynamic service handling at a central point, or restricting the integration of new devices to those being controlled through a single communication path, e.g. per UPnP. Creating a flexible system that offers a good mixture of everything, or even allows cutting out only the components that are required for a given use case, is incredibly challenging.

Investing in a System

Consumers currently have to pay a very high price in adding smart home technology to their new homes. Unfortunately, this applies to new buildings and retrofit solutions, and to homeowners as well as tenants. New homeowners are often short on money, and will not want to increase the size of their mortgage. Existing homeowners will shy away from the effort of retrofitting wiring in the home. Tenants do not wish to

invest in an apartment that does not belong to them. Thus, the technical solutions must make it easier for each of these groups of residents to invest in smart home technology (e.g. using easy-to-install wireless devices).

3 Approaches

Not claiming to have found the universal answer to all questions, we believe that the following six approaches can relieve of some of the above challenges. Of the six approaches, three affect the system architecture, two affect usability, and the final approach concerns business models.

As a running example to illustrate these approaches, we describe our current demonstration system installed in our labs in Munich and Beijing – either as it is, or as we now know it should be.

3.1 System Architecture with Centralized but Subsidiary Controllers

Whether to use a centralized architecture or one with distributed peer-to-peer controllers has been a matter of much debate in the past. Commercial systems often use a central *home server* or *home gateway* which controls all devices. Examples are Gira HomeServer [12] or the ePerSpace home gateway [1]. The advantage of a central server is that it provides a single point for implementing integrative applications.

Several research systems have suggested implementations based on peer-to-peer controllers, e.g. [13]. In principle, such systems can be more robust, since there is no single point of failure. In practice, manageability is a serious issue.

Our approach follows standard practice in building automation: the automation pyramid. A central home gateway or server runs a software platform for its applications, exposes wired and wireless IP network interfaces, and provides the central integration point. In our lab system, this is based on OSGi technology. Powerful devices supporting IP networking can be connected to it directly. In our case, this includes UPnP media servers and user interface clients. Other devices, i.e. ones that do not support IP networking, require a bridge or subsidiary controller. For example, in our lab installations, lighting and blinds control are automated by KNX, and a KNX/IP interface provides the necessary physical network translation between the wired KNX and wired Ethernet interfaces. The home gateway runs a software module that handles the KNX communication at IP level and exposes its functionality to the rest of the software platform.

In practice, many functions will not require intervention of the home gateway. For example, switching on light using a KNX switch is handled entirely on the KNX twisted-pair bus. The home gateway only observes the KNX actions for monitoring purposes, in order to keep its internal representation of devices' status up to date.

This principle of *subsidiarity* keeps the system simple. Our design rule is to use the simplest technology that will perform the task and allow further integration.

Of course, this principle applies to other technologies as well – DALI lighting automation, ZigBee wireless sensor nodes or LON devices. In each case, a subsidiary controller and network bridge is required, as well as an appropriate driver module for the home gateway.

For special-purpose markets, a home gateway (or a home server without additional networking functionality) can be built including hardware interfaces for KNX, DALI, ZigBee, etc. Since there are so many competing bus technologies on the market, this will only pay off once the installed base reaches a critical size.

The advantage of running controlling software on a central gateway is that it is always on and connected to the home IP network as well as the internet. This allows attractive business models for telecom operators. However, for business models that should be independent of telecom providers, the central software can run on any other embedded device in the home that is always on and connected to the home network.

The central home gateway and its connected devices are shown in Figure 2.

3.2 Open Interfaces at Software and Network Levels

No single manufacturer will be able to supply all components of a complete smart home system. Thus, interoperability is of the essence to succeed on the market. We suggest that interoperability be standardized at three different levels of interfaces. One is a software API level on the home gateway. The other two are network levels between devices. All of these are based on open standards, of which there are many in the smart home arena. A forthcoming ISO specification document [14] shows a taxonomy of standards in the smart home domain.

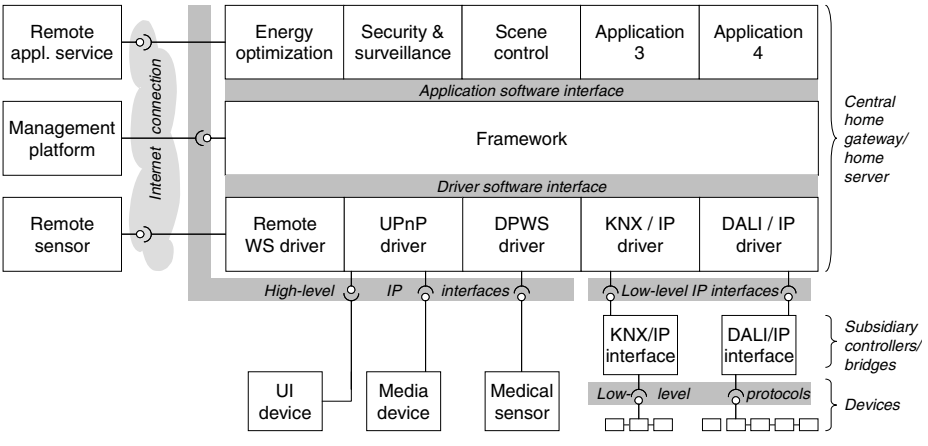

Fig. 2. Proposed smart home software and system architecture with example devices, drivers and applications. The different types of interfaces are shown in gray.

Software Interfaces: Application and Driver Interfaces

For APIs on the home gateway, we suggest standards based on a commonly accepted component platform such as OSGi or .NET. The software interfaces must cover at least two layers, shown in gray in Figure 2: an application interface, and a interface supporting different types of drivers. In our demonstration systems, we have created a family of OSGi-based *smart home* interfaces. Other research groups have done the same thing. Vendors such as Prosyst [15] have started creating their own implementations. Domain-specific standardization, e.g. within the OSGi Alliance, is required here.

High-Level Network Protocols Based on IP
Between devices, network-level integration standards are required. For the most general compatibility, these should be based on IP.

In our demonstration system, we use the UPnP family of standards (including DLNA extensions), which have become state of the art in industry and in the consumer market. Thus, the user (or technician) can easily add new crafts, as long as they support UPnP. Other emerging, sometimes competing standards here are DPWS (Device Profile for Web Services) or IGRS (Intelligent Grouping and Resource Sharing). In practice, a home gateway may well need to support several of these.

Subsidiary Control Networks: Low-Level Bus and Low-Level IP Protocols
Addressing each light in a house directly is prohibitively expensive, and raises scalability issues. Thus, subsidiary control networks, e.g. DALI for lighting, are still required, and as described above, different industry standards exist for low-level protocols. Device manufacturers should adhere to these, and they should be extended, where necessary, to allow better control and monitoring by central controllers. To connect subsidiary networks to the gateway, a bridge or controller must expose this protocol at least partially as IP. Some protocols, e.g. KNX, already support mappings to IP. These are generally distinct from the high-level IP protocols such as UPnP.

Better Inter-Standard Harmonization Needed
Unfortunately, different standards are governed by different internation bodies with different members and interests. Work between the standardization bodies, e.g. UPnP Forum and OSGi Alliance, has been done, and has resulted in interoperability standards such as the OSGi UPnP base driver specification [16]. However, more work is required here, since such inter-standard standards often end up as a lowest common denominator. For example, in the OSGi UPnP base driver, there is no standard "language mapping" between UPnP actions and Java methods, which would be fairly simple and could greatly improve the standard's usability for programmers.

3.3 Optimize for Responsiveness

Users want immediate response from devices in their homes; they're used to it from simple light switches. Turning on all lights in a house, for example, should be instantaneous. (For effect, dimming them on simultaneously might be even more pleasant, but reaction to user commands should still be instantaneous.)

Therefore, when choosing implementation technologies, one should optimize towards responsiveness. This means using lightweight technologies where possible; re-marshalling method calls from one XML format to another is likely to cause unnecessary delays, especially when operating on an embedded system. Commands should be sent concurrently instead of sequentially, if the target device or bus allows this. Even waiting for acknowledgments should be considered carefully.

For event handling, it means avoiding chattiness and optimizing for event bursts. In this context, event handling (e.g. from environmental sensors) is harder than one might think. On average, eventing is rather infrequent. But when the environment is modified, multiple devices have to be controlled, and this leads to a cumulative firing of events. For simple events (as they occur for example on the KNX bus) this may be

easy to deal with. But for a real smart home, event types are more complex, carrying more detailed content. Batched events are beneficial here, that are sent to subscribed parties only. These should be compacted and allow parsing only details of interest.

An open question is the missing standard for such an eventing system. The eventing model of UPnP, as we have discovered ourselves, does not scale beyond several dozen devices. Thus, turning on 30 UPnP binary lights simultaneously was prohibitively slow in one of our demonstration systems and forced us to use non-standard UPnP device profiles. One interesting option would be a "smart home profile" or "industrial profile" for UPnP or DPWS, which would be backward-compatible, but specialized for scalability, just as DLNA is for multimedia purposes.

3.4 Interaction Principle: Intentionally Invoked Scenes, Prioritized by Context

In the effort to make increasing complexity manageable without trying to know better than the user, we chose the following control principle. We bundled all control possibilities into *scenes* according to recurring activities and typical environmental states. The user can manually invoke scenes ("dinner", "relax", etc.). Of course, in practice, a list of scenes can become very long. To make selection easier, we use prediction of what scenes will most likely be used next. This produces a weighting for all scenes, allowing the system to display only a clearly arranged number of scenes, emphasized in style to express the assumed probability.

In contrast to systems that act autonomously and assuming a user's intervention as training data only [8], we propose that the user makes a conscious choice. This may be simplified to an acknowledgment which starts the most probable scene, e.g. through a gesture. This follows the recommendation in [17].

Our concept also leaves room to fine-tune several elements of a scene through special devices. There can be knobs used to influence music's beat, or digital photo frames presenting photos according to tags placed on some kind of pinboard. This usually affects devices that would tend to become boring when simply replaying statically programmed properties any time a scene starts.

Our guiding mental model was that of polite butler. "May I serve the tea, or would you like the evening paper?" – "Tea please, James. But today I'd prefer green tea."

An open question remains: how to program these scenes easily? For the end user, we designed a *snapshot system* that can save devices' current state as a scene. More powerful programming, with sequences of control commands and conditional behaviour, require more complex tools. As few users want to program complex scenes themselves, a business model for technical support or remote service is needed.

3.5 Separate Control Devices with Differing Richness and Modalities According to Situation and User Expertise

The optimal interaction device for a smart home is a matter of much debate. One approach is the universal remote control, offered by several companies. Logitech offers remote controls with infrared and Z-Wave interfaces. Universal Electronics' Nevo series has additional Wi-Fi networking. Another approach is the set-top box or media PC as user interface, e.g. Microsoft's Media Center.

We contend that there is no such one-fits-all interaction device for smart homes; instead, users should be able to operate the home through various very distinct interfaces. This includes tangible devices (e.g. wall buttons, switches), gesture and voice control, small displays (e.g. integrated 1-line display on a refrigerator), mobile rich clients (universal remotes, smart phones) and integration into set-top-boxes and media PCs. The system should be able to interoperate with all control device classes, covering units for the different target group. The devices differ in modalities and richness, and with different devices, different sets of functionalities can be supported.

In our Lab system, we discovered that elderly people prefer to have only concrete and straightforward interfaces; interfaces with big display and keys but only a small choice of selection and a very few hierarchies are good candidates here. Seniors also liked speech control. On the other hand, technically affine people (especially the younger generation) favour the possibility of rich options, as long as they can also make simple and quick adjustments.

We developed interfaces for full control (e.g. on a PDA), and all but the most technical users were intimidated. We then developed other approaches with fewer options, but less complexity, optimized for use with the scene control system (see Figure 3) described in the previous section. These were more popular with lab visitors. This simplification is a two-edged sword – you must hide enough complexity without trying to know better than the user.

An open question remains: how to ensure consistent look and feel across such a range of devices?

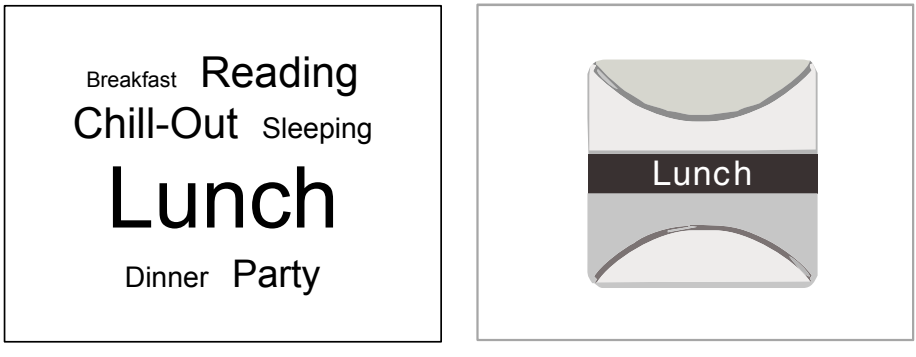

Fig. 3. Scene cloud and display-enriched switch as different user interfaces for scene control in a smart home environment. The most probable next scene, *Lunch*, is highlighted.

3.6 Business Models: User Value and Extensibility

Beyond technical issues, getting the business model right has proven very difficult for smart homes. Different models have been tried – retail market, hosted operations, custom-built luxury homes, and various others. Perhaps the most advanced are emerging luxury markets with much new building construction. One example is Dubai, where integrators such as Smart Home UAE [18] offer custom solutions to the housing industry and to individual investors. However, regional markets such as Europe

and the USA are very different, both regarding customer willingness to invest in different applications, and the relative value of new construction vs. retrofit solutions.

The only promising approach we see here is to focus on applications with concrete added value for the user, but to keep the systems open for extension. For example, one market study [19] shows that within Europe, end users show the highest willingness to invest in energy-saving and security solutions, as opposed to comfort systems. Thus, the key to market success is to offer a system for saving energy or an integrated security system, but to design this so that it uses the open standards described in Section 3.2. Then, once the system is installed, users will have the option of adding on additional applications such as for comfort or home healthcare.

Many new applications will require new collaborations between different market players. For example, smart metering and energy-saving solutions are currently bringing previously unrelated market players together, such as Google and electrical utility companies [20].

4 Conclusion

In smart homes, as everywhere else, there is no such thing as a perfect solution. However, we believe that smart homes are possible for the mass market in the near future, if a significant number of players in the market and in research act together, following the recommendations Section 3, and create valuable applications for end users while keeping their systems open for extension.

This cooperation of disciplines and of industry and academia is vital for success. On the industrial side, the lead role could be taken by one global player, but more likely, by a small consortium which can supply the home equipment, the back-end management infrastructure, and the services (e.g. security) adding value for the user. If, and only if, the system is kept open for extension, this can lead to widespread adoption and commercial success, with many other parties contributing device extensions or future services.

References

1. ePerSpace (Towards the era of personal services at home and everywhere). Integrated Project under the EU 6th Framework Programme (2006), http://www.ist-eperspace.org/
2. Pictures of the Future. T-Com House: At Home in the Future. In: PoF Fall 2005, Siemens AG (2005), http://www.siemens.com/pof
3. serve@Home Product Brochure. Siemens Electrogeräte GmbH, Munich (2006)
4. Cook, D.J., Youngblood, M., Heierman, E.O., Gopalratnam III, K., Rao, S., Litvin, A., Khawaja, F.: MavHome: An Agent-based Smart Home. In: PerCom 2003. LNCS, pp. 521–524. Springer, Heidelberg (2003)
5. ISO/IEC 14543-3. Konnex Association, Standard (2007), http://www.knx.org/knx-standard/
6. PEHA EasyClick: Electrical installations without miles of wiring, http://www.peha-elektro.com/Easyclick.aspx
7. Digital Living Network Alliance (DLNA), http://www.dlna.org/

8. Mozer, M.C.: The Neural Network House: An Environment that Adapts to its Inhabitants. In: Proceedings of the American Association for Artificial Intelligence Spring Symposium on Intelligent Environments, pp. 110–114. AAAI Press, Menlo Park (1998)
9. Bickmore, T.W., Picard, R.W.: Establishing and maintaining long-term human-computer relationships. ACM Transactions on Computer-Human Interaction (TOCHI) 12(2), 293–327 (2009)
10. There Corporation. Press Report from, May 29 (2009), http://www.therecorporation.com/, http://www.smarthomepartnering.com/cms/?p=100
11. OSGi Alliance. Specifications, http://www.osgi.org/, http://www.osgi.org/Specifications
12. Gira HomeServer3, Gira FacilityServer: Intelligent building management via KNX/EIB and TCP/IP. Order no. 1887 90 08/08 5. 16. Giersiepen GmbH & Co. KG, Duisburg (2008), http://download.gira.com/data2/1887900508.pdf
13. Turcan, E., Graham, R.L., Hederen, J.: Peering the smart homes. In: Proceedings of First International Conference on Peer-to-Peer Computing, 2001, pp. 103–104 (2001)
14. ISO/IEC PDTR 29107-1: Information technology, Intelligent homes, Taxonomy of specifications – Part 1: The taxonomy method. ISO/IEC JTC 1/SC 25 N 1652, Working document (2009)
15. Prosyst mBS Smart Home Extension. Prosyst Software (2009), http://www.prosyst.com/products/osgi_ext_smart.html
16. OSGi UPnP Device Service Specification, OSGi R4, Version 4.1. Service Compendium 111, pp. 253–280 (2007)
17. Intille, S.S.: Designing a Home of the Future. IEEE Pervasive Computing 1(2), 76–82 (2002)
18. Smart Home Automation Control, Digitcom Technology, Dubai, UAE, http://www.smarthomeuae.com/
19. Szuppa, S.: Marktforschung für komplexe Systeme aus Sach- und Dienstleistungen im Privatkundenbereich. Entwicklung und Überprüfung eines Vorgehenskonzeptes am Beispiel des Intelligenten Hauses. Ph. D. thesis, Verlag Dr. Kovac, Hamburg (2007)
20. Google Power Meter, http://www.google.org/powermeter/