CARUSO - Towards a Context-Sensitive Architecture for Unified Supervision and Control

Rolf Kistler, Stefan Knauth, Daniel Käslin and Alexander Klapproth

***Abstract*— This paper discusses an architecture that aims to unify and simplify the supervision and control of networked devices in commercial building environments. To be adoptable for real-life applications, the technical key features of such an architecture are mainly derived from four high-level requirements: The system must add true user value, still be easy to use, work for a broad range of quite different devices and minimize the engineering costs. This results in a distributed, ad hoc capable and scalable hard- and software infrastructure with the ability to adapt to the context of use and goal-centric services provided by the numerous underlying devices of the heterogeneous building network. Preferably, standard mobile clients such as smartphones and PDAs act as control points providing a graphical end user interface generated on the fly. This text explores the different building blocks that make up such a system, elaborates topics that are currently under research and proposes a solution.**

I. Introduction

Numerous electronic devices populate a modern building today. Many of them are network enabled and allow some form of remote control. The need to reduce operational costs led to more flexibility and optimized management through networked solutions in the domain of commercial building automation at an early stage of the development. KNX/EIB and Echelon LonWorks systems are now implemented in thousands of buildings. More or less recently, other domains (security and safety, home automation, consumer electronics, domestic appliances...) have come up with their own network solutions. So in the buildings of the next generation, all these networked devices will work seamlessly together for the convenience of the end user and to the benefit of the solution provider as well as the investor? Well, looking at the current state, the sheer number of different technologies and standards waiting to be adopted in the networked building is overwhelming. Technological trends such as wireless systems or service oriented architectures (SOA) and the well known strengths of TCP/IP based protocols will probably lead to some level of harmonization. However, a building network will continue to be a very heterogeneous environment for many years to come.

From an end user perspective, one finds that there is no common way to control all these devices. Today's user interfaces are specific (domain, manufacturer, device), inflexible and often proprietary. There are proprietary solutions that harmonize control for selected high-end users in the home automation segment, but these Smart Homes are "hand-crafted". Bringing the heterogeneous zoo of devices together comes at (engineering-)costs a provider of commercial building applications strictly wants to avoid. A harmonized, intuitive and still cost-effective solution could bring real value to all stakeholders: End users are able to fulfil their tasks more efficiently using adaptive, context-sensitive, responsive interfaces. For instance, one could think of applications involving building technicians or caretakers doing maintenance work, security officers getting detailed alarms and building status, secretaries preparing meeting rooms or any person who needs to hold a presentation in a new multimedia room. They can use their own familiar control device (e.g. smartphone), the user interface adapts to the level of their knowledge and every multimedia system looks the same providing only the services needed to fulfil their specific goal. Devices from various vendors can be combined and integrated ad hoc with a minimum of engineering efforts, even in fast changing environments. So the system provides flexibility also for the solution provider and the owner of the building.

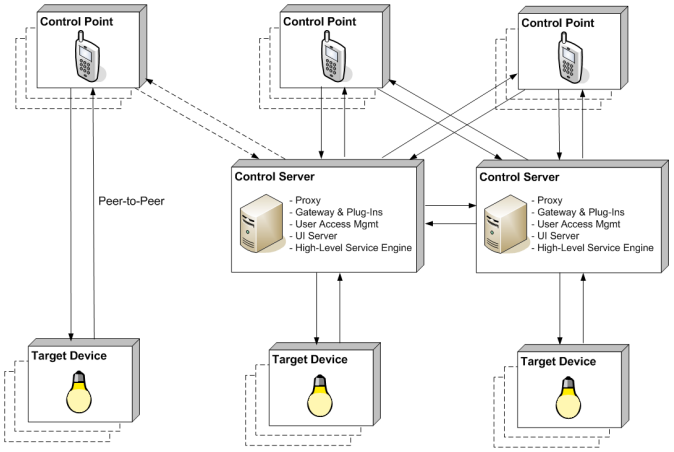

Fig. 1. CARUSO Scenario: Two synchronized Control Servers acting as Proxy, Gateways, User Access Servers, UI Servers and High-Level Service providers. One Control Point in Peer-to-Peer mode (fall back)

II. Architecture and Implementation

A. Network Topology and Infrastructure

We believe that IP will play an increasing role in the next generation of building networks, including the building automation field level. So to start with, we assume that every communication partner in the network is able to exchange data over IPv4 and UDP, either directly or over a gateway. Further, all communication partners must gain access to the IP network. Secure network connection and IP configuration issues are not addressed.

The network topology affects many of the characteristics of the system. For sure, the network must integrate the devices to be controlled, the **Target Devices (TD)** and connect them to the controlling devices, the **Control Points (CP)**. Two modes of operation are proposed: A **peer-to-peer mode** where the

control points directly communicate to the target devices and a **client-server mode**, which introduces new server elements. Depending on the building environment, the functionality of these server elements may be concentrated on one physical node or be distributed and multiplied over the network to provide scalability, network segmentation and improve the performance. The servers provide the following functionality:

- **Proxy**: Builds the bridge between CPs and the rest of the building network. It discovers the target devices, collects their descriptions and makes them accessible to CPs. The proxy hides the addresses of the underlying target devices and speeds up discovery of low-power devices.
- **User based device access**: The CPs build up secure server sessions through user authentication. Depending on user access levels, the server decides about the services and actions exposed to the CPs. Example: Predefined standard users have access to ”public” building functions available from the local room without login. Multiple requests to TDs are prioritized and synchronized.
- **Gateway**: A plug-in architecture allows the server to act as a gateway. A gateway (unlike a proxy) actually translates field protocols to integrate IP/UPnP incompatible devices. Runtime installation of plug-ins introduces new protocols and adds plug and play functionality.
- **UI Server**: The TDs must be presented to the user on the CPs somehow. The user interfaces (UIs) shall neither be statically stored on the CPs nor come from the TDs. The CPs build up their user interfaces on the fly using UI descriptions dynamically generated on a UI server.
- **Services Server** An optional component holds high-level services and supplementary resources (icons, UI skins, help texts ...). Other server components can discover these shared services and provide them to the user.

For several reasons, OSGi has been chosen as technology for the server components. First, OSGi is modular and provides a sophisticated life cycle management that is well suited for distributed embedded devices with dynamic software migration. Second, it is service oriented and provides many standard services ”out of the box” (logging, user admin...).

Under normal operating conditions, the CP can rely on the services of these servers. However, if no server can be located, the CP has the capabilities to discover and control TDs within its physical range on its own (UPnP peer-to-peer mode). In this fall back scenario, it may use previously cached server information or provide only a restricted functional UI based on UPnP descriptions. No high-level services are available. If TDs do not implement any security features, they can be directly controlled. The P-t-P scenario, UI considerations, local storage and the current state of mobile browser technology led to the decision to place a dedicated control application on the CPs (initially zero-deployment was the goal).

B. Plug and Play Functionality

UPnP (Universal Plug and Play) is the natural choice to implement plug and play functionality in CARUSO. It’s very powerful and seems to be a widely accepted protocol for this purpose. The cross-industry organization DLNA [4] has included UPnP as an integral part into their ”guidelines based on open industry standards to complete the cross industry digital convergence”. Developed by Microsoft, UPnP initially targeted PC peripherals and home automation. But case studies and prototypes have indicated that it can also be adopted for commercial building automation [5] [6].

UPnP defines protocols and procedures involved in IP addressing, device and service discovery, device and service description, control, eventing and presentation [1]. Presentation is limited to an URL of an optional presentation page (mostly HTML). Besides this presentation link, UPnP device and service descriptions do not contain any UI related information. To selectively find devices and services, UPnP may post search messages to the network (e.g. ”all colour printers”, ...).

On top of that, the UPnP Forum has developed specifications for device classes, so-called ”Device Control Protocols” (DCPs). Each DCP defines a common interface for a class of TDs and its services. Control over any UPnP DCP compliant device becomes easy with the knowledge of its well-defined interface and the ability to detect it. DCPs have been defined for many TD classes such as Media Servers, Printers, HVAC devices, lighting equipment, UI servers/clients and others.

UPnP solves a lot of problems. Two things it cannot do: UPnP is inherently device-centric and it has poor support for presenting user interfaces. For high-level services involving more than one device and for the user interface presentation, other solutions need to be found.

C. User Tasks and Services

Users approach the system with certain tasks in their mind. These may be as simple as ”turn on the light” or ”set the room temperature to 20 degrees centigrade”. But they can also become more complex, ”watch a DVD” or ”prepare Room 302 for a presentation meeting”. The system places services at the user’s disposal to achieve her goals. The granularity and characteristics of these services greatly affects the usability.

On one end, an ideal system knows a minimal number of complex, **intelligent high-level services**, in which each service exactly maps to a user task. A minimum of interaction steps is required to finish a task. However, the chance is high that a one-to-one mapping cannot be achieved and thus the system becomes inflexible. A service that almost does what the user intends is probably more annoying than helpful. On the other end stands a system with many small and simple low-level services that could, in an intelligent orchestration, achieve any user task. Here, the user needs to know a number of these services and apply them in the correct order, which reduces the ease of use considerably.

Fact is that, in general, the user is not interested what devices are involved, he just wants to get the job done. Most of today’s remote controls pose device-centric views on the user. Such a view maybe helpful in a specific situation, like a building engineer reading a parameter out of a device. However, most of the time, users would prefer to see the system as a collection of convenient services that closely resemble their needs.

CARUSO takes both views into account. The selection criterion is the role of a person in the building (she can

have more than one role, but not at the same time). For this reason, roles can be defined (standard user, advanced user, fitter, engineer, administrator). The role of the user defines the view and the security level.

Besides of low-level (UPnP) services, CARUSO provides high-level services to its users. They come in the form of dynamically loadable server plug-ins. A high-level service is an executable (OSGi bundle) that runs on a server, triggered over a CP (or from another server). This engineered or partly dynamically generated service executes low-level services to achieve a certain goal. It's left up to the implementer of the service how intelligent the low-level services are chosen and aggregated. A CARUSO UPnP device - the "Services Server" - publishes these high-level services over the UPnP network. In addition, the CP itself provides a facility for the user to record macros or scenes, store them locally and replay them later on. These are simple sequential execution blocks for low-level functions on different devices without the ability to take influence on the timing or the execution flow.

D. User Interface

The user interface represents the system to the user and thus is naturally a very important factor for the acceptance of any control system. Although many possible human machine interaction schemes exist [7], CARUSO sticks to a 2D graphical user interface (WIMP) that can be shown on mobile clients. The user interface shall be able to adapt to: System status, display size, display resolution, input modality (keypad, pen/touch screen, mouse/touchpad/keyboard), user role, user language, computing performance and network bandwidth. The conclusion was made that dynamic UI generation mechanisms best meet our requirements.

Two main criteria influence the decision on how to implement these UI mechanisms: (1) Where the generation process takes place (2) when it does so. Other factors are the proper separation of control application/UI presentation/UI logic and an appealing presentation scheme that goes behind textual UIs or fixed standard widgets.

In the chosen solution, the CP gets a link to a file generated in a **User Interface Description Language (UIDL)**. Its a declarative description of the user interface. The CP parses the file - which is coming from the UI server - and generates and renders the final UI out of it. Numerous such languages have been created already (the authors counted 25 up till now). Besides most of them are in XML, they vary greatly in what they can do and how they do it (e.g. level of abstraction). The user interface description builds the bridge between the purely functional service description and the final UI. It must provide features like grouping of user interface elements, element hierarchies, internationalization etc. that are needed to build a sophisticated UI. Statements made should be on a higher level than "place a button with height/width to position x/y". The decisions on what widgets to take and where to place them is left up to the CP. It best knows about its own possibilities to render the UI. The .NET compact framework has been evaluated as technology for the control application and the rendering. Using this "thick client" technology on the mobile

Fig. 2. Upper row, left: UPnP Control Point Prototype, right: Generated SVG Prototype for service "Watch DVD". Lower row: Design Studies

device was also influenced by an evaluation of other state-of-the-art UI technologies. During this evaluation, an SVG prototype has been implemented (Fig. 2). It produced quite usable interfaces. However, we finally found that SVG is more an enhanced textual vector graphics format than an UIDL and decided upon XAML, introduced by Microsoft with the new Windows Presentation Foundation (WPF) of Vista.

E. Implementation

A demonstrator of the control point with a functional tree based UI has been built (Fig. 2). The system was tested in peer-to-peer mode and with an implementation of the UPnP Proxy running as an OSGi Java bundle on an industrial PC (PIII, 800MHz) and Ubuntu V6.06 Linux. Tests with around 30 PC simulated UPnP devices (dimming lights, media servers and players) have shown that the proxy approach is feasible and fast enough for lighting and jalousie applications (reaction time below 200ms).

Currently, we work on a modular, user-friendly UI design and the mechanisms to dynamically generate and render it using XAML. Further, a gateway plug-in integrating wireless ZigBee devices into CARUSO is under development. Together with the project industry partner, we investigate on how CARUSO servers could be integrated into existing IP capable building controllers including re-usage of device access information and management tools. Additional configuration and hardware costs shall be kept to a minimum.

III. Related Work

Many have contributed valuable ideas, concepts and technologies that inspired CARUSO. A bunch of quite sophisticated universal remote controls for homes can already be bought and there are research projects that come close to what this proposal suggests. Apart from common universal remote controls and remote control software for PDAs, the most notable products on the market for home environments are the **Philips Pronto Series**, **Logitech's Harmony**, **AMX R4 ZigBee** and products from **Xabler** and **Nevo**.

The **PECo (Personal Environment Controller)** [8] comes with an intuitive and novel interaction metaphor that provides 3D-sights of the rooms controlled on a PocketPC. The creators of PECo have also proposed a generic UPnP architecture [9] and mention EIB. They have implemented their concept in an ambient intelligence meeting room. PECo needs a detailed model of the physical environment. **PUC (Personal Universal Controller)** [10] [11] is a peer-to-peer approach to control devices over various standard clients. The UIs are generated on the client with help of a proprietary user interface description in an XML dialect developed for this project. A prototype has proved the concept on a Palm device and a smartphone. The new **URC (Universal Remote Console)** [12] standard proposes a "Protocol to facilitate operation of information and electronic products through remote and alternative interface and intelligent agents". The first attempt to really standardize remote UIs. The creators of the standard have proposed an **URH (Universal Control Hub)** architecture based on UPnP using their concept of UI Sockets and task based user interfaces that is close to CARUSO [13]. A lot of work has been done concerning smart services, intelligent device ensembles and how to get to them (e.g. using pattern matching or distributed intelligent agents) [14]- [17]. Model-driven user interface techniques have delivered many ideas on how to dynamically generate user interfaces. Interesting work has been done concerning multi-modal [18] and multi-target user interfaces [3] also especially related to mobile devices [19]. **iCrafter** [20] provides a service framework for ubiquitous computing environments that supports UI selection, generation and adaptation. It also includes a service concept with patterns for on the fly aggregation of services.

IV. Conclusion

The paper sketched an architecture for unified supervision and control of devices in a commercial building environment. The proposed IP network provides distributed server components for tasks such as access control, UIs, field protocol plug-ins and high-level services but also includes restricted peer-to-peer scenarios. UPnP introduces the required plug and play capabilities and the architecture assures that standard based target devices can be integrated into the system without modification. Dynamically generated, bi-directional user interfaces and a service based approach shall provide the flexibility and user friendliness critical for the acceptance of such a system. We think that the proposed pragmatic concept enables a new generation of control systems for buildings that can be realized not too far from now.

References

[1] UPnP Forum, "UPnP Device Architecture", Version 1.0, June 2000, *[ONLINE] http://www.upnp.org/download/UPnPDA10_20000613.htm*

[2] OSGi Alliance, "The Dynamic Module System for Java", *[ONLINE] http://www.osgi.org/*

[3] G. Calvary, J. Coutaz, D. Thevenin, Q. Limbourg, L. Bouillon, J. Vanderdonckt, "A unifying reference framework for multi-target user interfaces", *Interacting with Computers, Volume 15, Issue 3, June 2003, Pages 289-309*

[4] DLNA - Digital Living Network Alliance, "DLNA Overview and Vision Whitepaper 2006", *[ONLINE] http://www.dlna.org/en/industry/about/dlna_white_paper_2006.pdf*

[5] W. Kastner and H. Scheichelbauer, "UPnP Connectivity for Home and Building Automation", *Parallel and Distributed Computing and Networks, 2004*

[6] A. Klapproth, D. Käslin and T. Peter, (Apr. 2005), *Lowcost Wireless Webserveer*, *ZigBee Entwicklerforum 2005 Munich, [ONLINE] http://www.ceesar.ch/cms/upload/pdf/Paper%20Wireless%20Webserver%20HTA%20Luzern.pdf*

[7] Ali A. Nazari Shirehjini, "Klassifikation der Human-Environment-Interaction in intelligenten Umgebungen.", *Informatik 2006. Informatik für Menschen. Band 2 : Beiträge zur 36. Jahrestagung der Gesellschaft für Informatik e.V., Bonn : Gesellschaft für Informatik, p382-389, 2006*

[8] Ali A. Nazari Shirehjini, "A novel interaction metaphor for personal environment control: Direct manipulation of physical environment based on 3d visualization", *Computers & Graphics, Special Issue on Pervasive Computing and Ambient Intelligence, Volume 28., Elsevier Science, p667-675, 2004*

[9] Ali A. Nazari Shirehjini, "A Generic UPnP Architecture for Ambient Intelligence Meeting Rooms and a Control Point allowing for integrated 2D and 3D Interaction.", *Smart Objects and Ambient Intelligence. SOC-EUSAI 2005. Proceedings : A Joint Conference SOC-EUSAI. Innovative Context-Aware Services : Usages and Technologies. 2005, pp.207-212*

[10] J. Nichols, B. A. Myers, M. Higgins, J. Hughes, T. K. Harris, R. Rosenfeld, M. Pignol, "Generating remote control interfaces for complex appliances", *Proceedings of the 15th annual ACM symposium on User interface software and technology, p161-170, 2002*

[11] J. Nichols and B. A. Myers, "Controlling Home and Office Appliances with Smartphones", *IEEE Pervasive Computing, special issue on Smart-Phones, Vol. 5, No. 3, July-Sept, 2006, pp. 60-67, 2006*

[12] G. Zimmermann, T. Nixon, M. Beard, E. Sitnik, B. LaPlant, S. Trewin, S. Laskowski, & G. Vanderheiden, "Toward a unified universal remote console standard.", *Extended Abstracts on Human Factors in Computing Systems, CHI2003 Conference on Human Factors in Computing Systems, p874-875, 2003*.

[13] G. Zimmermann, G. Vanderheiden, C. Rich, "Universal Control Hub & Task-Based User Interfaces", 2006, *[ONLINE] http://myurc.org/publications/2006-Univ-Ctrl-Hub.php*

[14] M. Hellenschmidt, T. Kirste, "SodaPop: A Software Infrastructure Supporting Self-Organization in Intelligent Environments", *IEEE International Conference on Industrial Informatics 2004. Proceedings : Collaborative Automation - One Key for Intelligent Industrial Environments., p479-486, 2004*

[15] M. Valle, F. Ramparany, L. Vercouter, "Flexible composition of smart device services.", *The 2005 International Conference on Pervasive Systems and Computing(PSC-05), 2005*

[16] M. Valle, F. Ramparany, L. Vercouter, "Dynamic service composition in ambient intelligence environments: a multi-agent approach." *Proceeding of the First European Young Researcher Workshop on Service-Oriented Computing, 2005*

[17] M. Vukovic, P. Robinson, "Adaptive, planning-based, Web service composition for context awareness", *Second International Conference on Pervasive Computing, Vienna, April 2004*

[18] P. Shroff, JM. Winters, "Generation of Multi-Modal Interfaces for Hand-Held Devices Based on User Preferences and Abilities", *IEEE D2H2 Distributed Diagnosis / Home Healthcare, 2006*

[19] J. Eisenstein, J. Vanderdonckt, A. Puerta, "Applying model-based techniques to the development of UIs for mobile computers", *Proceedings of the 6th international conference on Intelligent user interfaces, p69-76, 2001*

[20] S. R. Ponnekanti, B. Lee, A. Fox, P. Hanrahan, T. Winograd, "ICrafter: A Service Framework for Ubiquitous Computing Environments", *Proceedings of the 3rd international conference on Ubiquitous Computing, p. 56-75, 2001*