Gebäudemanagementsoftware auf Basis des OSGi-Standards

Ralf Vandenhouten, Thomas Kistel

Zusammenfassung

Der OSGi-Standard wurde für die Entwicklung komponentenbasierter Software in Java spezifiziert. In diesem Artikel wird eine Architektur für Gebäudemanagementsoftware vorgestellt, die auf Basis von OSGi entwickelt wurde. Grundlage der Client-Server-Architektur des entwickelten Systems ist das Eclipse-Framework, das mit Equinox eine Implementierung des OSGi-Standards liefert. Die verwendeten OSGi-Technologien ermöglichen eine modulare Integration unterschiedlicher Gebäudemanagement-Geräte verschiedener Hersteller in die Softwareplattform und dessen einheitliche Steuerung und Visualisierung. Die Client-Anwendung des Systems profitiert dabei von der komfortablen Benutzeroberfläche der Eclipse Rich-Client-Platform. Das Ergebnis ist eine flexibel einsetzbare Softwarelösung für ein breites Anwendungsspektrum, das von der Überwachung von Industrie- und Bürogebäuden bis hin zu Privathäusern mit Touch-Display-Bedienung reicht. Wesentliche Vorteile der Lösung sind die kontextsensitive Informationsbereitstellung sowie die Unterstützung und Automatisierung der Prozesse im Gebäudemanagement.

Der Artikel geht auf die technologischen Hintergründe des Software-Systems ein und stellt die betriebswirtschaftlichen Anwendungsfälle vor.

Abstract

The OSGi-Standard was designed for the component-based development of software applications in Java. The architecture of a facility management software which was developed based on OSGi is presented in this article. Basis of the client-server architecture of the developed system is the Eclipse framework which provides an implementation of the OSGi standard with Equinox. The OSGi technologies used allow for a modular integration of different facility management devices of different manufacturers into the software platform, and enable a consistent control and visualization. The client application of the system benefits from the comfortable user interface of the Eclipse Rich-Client-Platform. The result is a flexible software solution for a wide application range that reaches from the supervision of industry and office buildings to private houses and touch display operation. Context-sensitive provision of information as well as the assistance and automation of the processes in the facility management are the essential advantages of the solution.

In this article the reader will be introduced into the technological backgrounds of the software system. Furthermore the business use-cases of the software application will be explained.

1 Einleitung

Der Begriff Facility Management wird aus dem Englischen oft direkt als »Gebäudemanagement« übersetzt. Facility Management ist jedoch umfassender und beschreibt nach [1] eine Managementdisziplin zur ergebnisorientierten Handhabung von Facilities und Services. Ziele der Facility Prozesse sind u. a. die Befriedigung von Grundbedürfnissen von Menschen am Arbeitsplatz und die Unterstützung der Kernprozesse von Unternehmen. Facility Management betrachtet dabei den gesamten Lebenszyklus von Facilities: vom Planen, Erstellen, Gebäudemanagement bis zum Abriss der Liegenschaft. Gebäudemanagement ist somit nur ein Teil von Facility Management, wenngleich aber der komplexeste. Gebäudemanagement lässt sich in folgende drei Kernbereiche untergliedern:

- technisches Gebäudemanagement
- kaufmännisches Gebäudemanagement
- infrastrukturelles Gebäudemanagement

Das technische Gebäudemanagement wurde in den letzten Jahren in erster Linie durch die informationstechnischen Entwicklungen der Gebäudetechnik und Gebäudeleittechnik geprägt. Am Markt existiert eine Reihe von Softwarewerkzeugen für das Gebäudemanagement. Der Großteil dieser Softwarewerkzeuge ist

herstellerspezifisch oder auf bestimmte Lösungszwecke zugeschnitten. An der Technischen Fachhochschule Wildau wurde zusammen mit der ixellence GmbH und der Gemtec GmbH die Gebäudemanagementsoftware *Wotan* entwickelt, die die Kontrolle und Verwaltung unterschiedlicher Gebäudeanlagen ermöglicht. Hierzu gehören unter anderem

- Einbruchmeldeanlagen
- Brandmeldeanlagen
- Überwachungskameras
- Schaltsysteme
- Telefonanlagen

Zusammen ergibt dies ein flexibles Gesamtsystem, welches sich in sehr unterschiedlichen Bereichen, wie in Einkaufscentern, Banken, Bürogebäuden, Bundeswehrkasernen, aber auch Privathäusern einsetzen lässt.

2 Anforderungen

Die Anforderungen an ein solches Softwaresystem sind vielfältig. Eines der Kernkriterien ist die Integration unterschiedlicher Gebäudetechniken von verschiedenen Herstellern. Diese müssen gemeinsam auf einer grafischen Oberfläche einer PC-Anwendung (Client) dargestellt werden, sodass der Nutzer schnell einen Überblick über alle technischen Anlagen im Gebäude bekommt. Weiterhin muss die Möglichkeit bestehen, mehrere Clients starten zu können. Die Konfiguration aller Anlagen soll aber dennoch zentral erfolgen. Gleichzeitig besteht die Einschränkung, dass nicht alle Gebäudeanlagen direkt an einen Server angeschlossen werden können, da sich die Anlagen teilweise in verschiedenen Gebäuden befinden. Aus diesem Grund ist auch eine Benutzerverwaltung erforderlich, da nicht jeder Client alle Anlagen aus allen Gebäuden verwalten darf.

Eine Basisfunktion des Softwaresystems ist die Darstellung der Zustände der einzelnen Gebäudeanlagen auf der grafischen Oberfläche der jeweiligen PC-Anwendung. Weiterhin sollen diese Zustände ggf. verändert (geschaltet bzw. gesteuert) werden können. Als anschauliches Beispiel kann hier eine Einbruchmeldeanlage (EMA) genannt werden. Die Zustände der einzelnen Melder und Sensoren werden in einer grafischen Ansicht dargestellt. Außerdem soll in der PC-Anwendung ersichtlich sein, ob die EMA scharf (löst bei Einbruch einen Alarm aus) oder unscharf (löst keinen Alarm aus) ist. In Bürogebäuden werden Alarmanlagen oftmals während der Betriebszeiten unscharf gestellt, um Fehlalarme zu vermeiden. Außerhalb der Betriebszeiten (insbesondere am Wochenende) werden diese scharf geschaltet, um bei Einbruch einen Alarm zu signalisieren. Scharf und unscharf Schalten einer Einbruchmeldeanlage sind somit ein täglich wiederkehrender Prozess, der automatisiert erfolgen kann. Aus diesem Grund müssen bestimmte Steuerungsaktionen des Softwaresystems auch zeitgesteuert ausgeführt werden können.

3 Lösungskonzept

Das Softwaresystem Wotan wurde auf Basis der *Eclipse Rich-Client-Platform (RCP)* [2] entwickelt. *Eclipse* war ursprünglich nur eine Entwicklungsumgebung für die Programmiersprachen Smalltalk und Java und firmierte unter dem Produktnamen *Visual Age* von IBM. Heute wird das Eclipse-Projekt unter *eclipse.org*, einer Nonprofit-Organisation, betrieben. Eclipse wurde auf Basis einer offenen und erweiterbaren Plugin-Struktur entwickelt, von der man schnell erkannte, dass diese auch für die generische Entwicklung anderer Anwendungen genutzt werden kann. Da in Eclipse alle Funktionen in unterschiedlichen Plugins realisiert sind, ist es auch möglich, Eclipse durch eigene Plugins zu erweitern oder als Grundlage für eine eigene Applikation zu verwenden (Eclipse Rich-Client Anwendung).

Mit der Version 3.0 wurde Eclipse vollständig restrukturiert und die Ablaufplattform auf Basis von *OSGi (Open Service Gateway Initiative)* [3] gestellt, die das Kernstück von Eclipse bildet. Vom OSGi-Standard existieren unterschiedliche kommerzielle und nicht-kommerzielle (Open Source) Implementierungen. Im Eclipse-Projekt wird die OSGi-Implementierung unter dem Namen *Equinox* betrieben. Bei OSGi-basierten Anwendungen werden die Funktionen in einzelnen *Bundles* gekapselt. Das OSGi-Framework besteht dabei selbst aus ein oder mehreren *System-Bundles* (vgl. Abbildung 1).

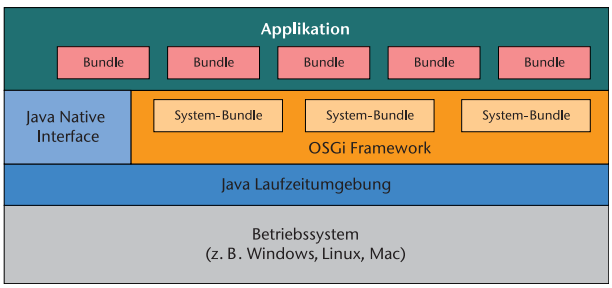

Abb. 1: OSGi-Architektur

Der OSGi-Standard bietet daher ein solides Rahmenwerk für die Modularisierung von Software-Applikationen. Ein großer Vorteil von OSGi ist, dass die Bundles dynamisch zur Laufzeit gestartet und gestoppt werden können, ohne dabei die Anwendung neu starten zu müssen. Dies ist insbesondere bei Serveranwendungen ein großer Vorteil, da dadurch ein unterbrechungsfreier Betrieb gewährleistet wird. Weiterhin können auch Aktualisierungen (Updates) für einzelne Bundles während der Laufzeit durchgeführt werden. Neben der Bundle-Technologie bilden die OSGi-Services eine weitere wichtige Komponente dieser Architektur. Jedes Bundle kann eigene Services exportieren, die von anderen Bundles dynamisch geladen und verwendet werden können.

Bei Eclipse Rich-Client-Anwendungen entspricht jedes Plugin immer einem OSGi-Bundle. Eclipse-Plugins besitzen somit immer den Funktionsumfang, den der OSGi-Standard definiert. Darüber hinaus gibt es bei Eclipse-Plugins einen Extension-Point-Mechanismus, mit dem es möglich ist, ein Plugin durch andere Plugins zu erweitern. Dieser mächtige Mechanismus ist einer der Hauptgründe für die große Verbreitung der Eclipse Rich-Client-Platform bei der Entwicklung eigener Anwendungen. Die populärsten Anwendungen, die auf Basis von Eclipse RCP entwickelt wurden, sind sicherlich die Jet Propulsion Laboratory Tools der NASA [4], die u. a. bei Steuerungsfunktionen in Mars-Missionen eingesetzt werden, und IBM's Lotus Notes, welches seit der Version 8 auf Eclipse RCP basiert.

4 Nutzungsmöglichkeiten von Eclipse und OSGi

Zur Realisierung der genannten Anforderungen wurde eine Client-Server-Architektur gewählt, wie sie Abbildung 2 zeigt. Die Gebäudeanlagen sind dabei jeweils an einem Anlagenserver angeschlossen, von dem ein oder mehrere vorhanden sein können. Die Konfigurationen der einzelnen Anlagenserver sind auf einem zentralen Konfigurationsserver hinterlegt. Die grafische Benutzeroberfläche des Gebäudemanagementsoftware *Wotan* ist auf dem Client installiert, der ebenfalls Konfigurationsdaten, wie Benutzerdaten, vom Konfigurationsserver lädt und sich zu verschiedenen Anlagenservern verbinden kann.

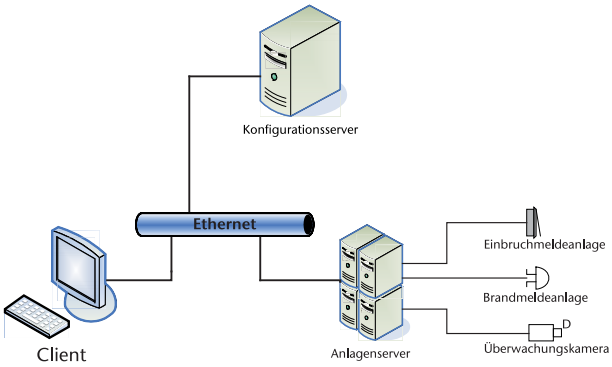

Abb. 2: Client-Server-Architektur des Wotan-Systems

Es existieren also drei verschiedenen Anwendungen, die das *Wotan-System* bilden:

- Client: grafische Benutzeroberfläche und Interaktionen
- Anlagenserver: Verbindungsschnittstelle zu den Gebäudeanlagen
- Konfigurationen: zentrales Laden und Speichern von Konfigurationsdaten

Die Kommunikation der Anwendungen untereinander erfolgt TCP-basiert und meist über Ethernet. Möglich ist auch die Anbindung der Anlagenserver an das System, z. B. über ISDN-Leitungen.

Während die Client-Applikation eine grafische Benutzeroberfläche (GUI) auf Basis von Eclipse RCP besitzt, kommen die beiden Server-Applikationen ohne GUI aus. Die Server-Applikationen können als reine OSGi-Anwendungen betrachtet werden, obgleich sie den Extension-Point-Mechanismus von Eclipse verwenden. Die unterschiedlichen Anlagen, die an den Anlagenserver angeschlossen werden können, sind softwaretechnisch in separaten OSGi-Bundles realisiert. Neue Geräte können somit einfach durch neue Bundles hinzugefügt werden. Die Geräte-Bundles sind selbst dafür verantwortlich, die Verbindungsdaten zu dem jeweiligen Gerät auszulesen und sich mit der Anlage zu verbinden. Bei erfolgreicher Verbindung wird das Gerät an einer Geräteplattform angemeldet, die dann im Wotan-System zur Verfügung steht. Die Geräteplattform ist für die Verwaltung aller an einem Anlagenserver angeschlossenen Geräte verantwortlich. Die Registrierung der Geräte bei der Geräteplattform erfolgte bislang über den Extension-Point-Mechanismus. In Zukunft soll dieser Mechanismus durch OSGi-Services realisiert werden, da dieses Vorgehensmodell noch mehr Flexibilität liefert [5].

Bei der Clientanwendung werden ebenfalls Extension-Points verwendet, so zum Beispiel für die Integration von Wizards, Dialogen und Ansichten in die grafische Benutzeroberfläche.

Jedes Plugin der Clientanwendung steuert damit seine eigenen grafischen Komponenten zur Gesamtanwendung bei. Beispielsweise existiert in der Clientapplikation ein Plugin, welches für die Erstellung und das Betrachten von Gebäudelageplänen verantwortlich ist. Dieses Plugin integriert einen Wizard zum Erstellen, einen Editor zum Bearbeiten und eine Ansicht zum Anzeigen der Lagepläne. Ist das Plugin nicht installiert, stehen diese Funktionen nicht mehr zur Verfügung. Dadurch sind die einzelnen Komponenten der Anwendung lose aneinander gekoppelt, sodass keine Abhängigkeiten zwischen den einzelnen Komponenten bestehen. Dies erhöht im hohen Maße die Wartbarkeit und Skalierbarkeit des komplexen Gesamtsystems.

Ein weiteres Beispiel für die Modularisierung des Gesamtsystems zeigt sich bei der Benutzerverwaltung. Hier wurde ein *Core-Plugin* entwickelt, das die Programmlogik des Rechtesystems (Laden/Speichern von Benutzerdaten, Überprüfung von Benutzerrechten) enthält. Die grafischen Komponenten zur Einstellung der Benutzerrechte oder der Login-Dialog beim Client wurden in einem separaten *UI-Plugin* realisiert, das eine Abhängigkeit auf das *Core-Plugin* besitzt. Das UI-Plugin ist nur beim Client installiert, während das Core-Plugin zusätzlich auch beim Konfigurationsserver installiert ist, da dort ebenfalls Benutzerrechte überprüft werden müssen. Damit werden Programmlogik und grafische Benutzeroberfläche in separaten Plugins getrennt, die eine bessere Wiederverwendbarkeit gewährleisten. Eine ähnlich flexible Lösung wird auch bei anderen Funktionsmodulen des Systems, wie den Timern zum zeitgesteuerten Auslösen von bestimmten Aktionen, verwendet. Die Timer werden beim Client über einen Dialog konfiguriert und je nach Einstellung beim Client (z. B. Anzeige eines Dokuments) oder Anlagenserver (z. B. Scharfschalten einer Einbruchmeldeanlage) ausgeführt.

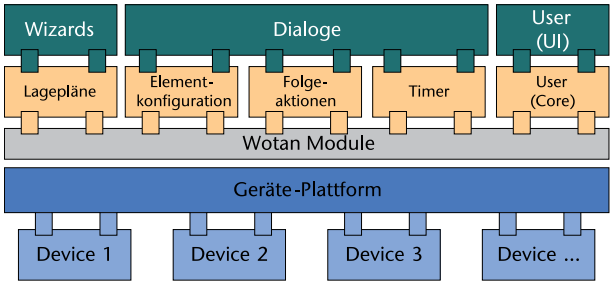

Abb. 3: Pluginstruktur des Wotan-Systems

Ein Großteil der Flexibilität der OSGi-Infrastruktur bzw. der Eclipse-Plugins besteht darin, dass Bundles der eigenen Anwendung andere Bundles dynamisch erweitern können. Dies kann zum einen über die OSGi-Services erfolgen, zum anderen aber auch über den Eclipse-spezifischen Extension-Point-Mechanismus. Letzterer wurde im Wotan-System verwendet, um die einzelnen Plugins erweiterbar zu gestalten. Die Paradigmen zur Bereitstellung eigener Erweiterungen sind in [6] sehr gut beschrieben. Im Wotan-System wurden beispielsweise die Konfigurationsdialoge mit Erweiterungspunkten ausgestattet, da *a priori* nicht klar ist, welche zusätzlichen Konfigurationen durch das System später noch realisiert werden müssen. Dadurch bleibt die Anwendung leicht erweiterbar, ohne die Basisarchitektur verändern zu müssen.

5 Anwendungsbeispiele

Die Anwendungsmöglichkeiten des Wotan-Systems sind äußerst vielfältig. Grundsätzlich werden in der Clientapplikation zwei Betriebsmodi unterschieden:

- Designmodus: In diesem Modus können alle Konfigurationseinstellungen am System vorgenommen werden.
- Nachrichtenmodus: Für den Überwachungsbetrieb der Gebäudemanagementsoftware

Abbildung 4 zeigt die Clientapplikation im Designmodus. Hier stehen für die Konfiguration eine Reihe von Wizards, Dialoge und Editoren zur Verfügung.

Abb. 4: Clientanwendung im Designmodus

Durch den bereits genannten Extension-Point-Mechanismus können die GUI-Elemente auch in Zukunft durch zusätzliche Funktionen erweitert werden. Die wesentlichen Ansichten und Dialoge des Designmodus sind:

- Die Lageplanübersicht stellt alle mit dem Wizard erstellten Lagepläne in einer Baumansicht dar.

- Die Modulansicht (vgl. Abbildung 5) visualisiert alle angeschlossenen Geräte der jeweiligen Anlagenserver.
- Im Dialog für die *Element-Konfiguration* (vgl. Abbildung 6) können alle Einstellungen für die Geräte, Maßnahmepläne, Folgeaktionen und Benutzerrechte vorgenommen werden.
- Im *Timer*-Dialog werden zeitgesteuerte Aktionen konfiguriert.

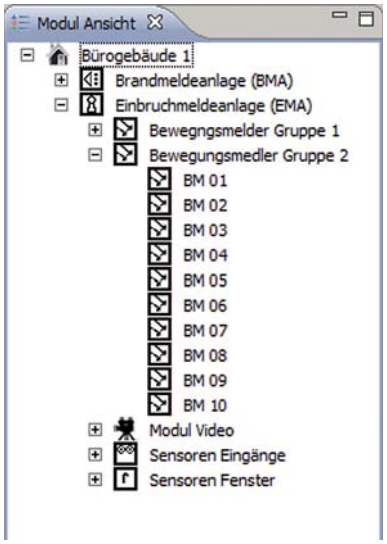

Abb. 5: Modulansicht

Die im Dialog *Element-Konfiguration* erstellten Maßnahmepläne werden im Nachrichtenmodus angezeigt, wenn ein bestimmter Melder in einen definierten Zustand wechselt. So kann dem Nutzer ein Maßnahmenkatalog angezeigt werden, wenn ein bestimmter Melder (z. B. Rauchmelder) ein Alarmsignal sendet. Ähnlich sind auch die *Folgeaktionen* zu betrachten, die im Gegensatz zu den Maßnahmeplänen keine direkten Handlungsanweisungen für den Nutzer darstellen, sondern Aktionen, die durch das Wotan-System automatisch ausgeführt werden. *Folgeaktionen* sind dabei beliebige Aktionen, die durch das Wotan-System ausgeführt werden sollen. Dies können Schaltbefehle von angeschlossenen Anlagen, das Öffnen oder Drucken von bestimmten Dokumenten oder aber auch das Öffnen von Kamerabildern sein.

Eine mögliche Folgeaktion für das Alarmsignal eines Rauchmelders kann z. B. das Ausdrucken einer Feuerwehrlaufkarte sein, die der eintreffenden Feuerwehr den Weg zum Brandherd zeigt.

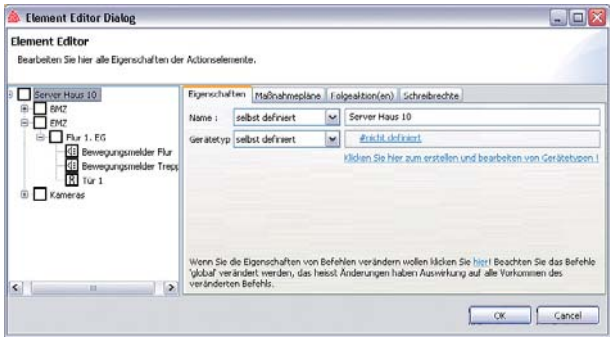

Abb. 6: Konfigurationsdialog für Wotan-Elemente

Der Nachrichtenmodus besitzt die Möglichkeit, auch in einem *Vollbildmodus* zu starten. In diesem Modus werden alle GUI-Elemente, wie Menüleiste oder Statusleiste, ausgeblendet, sodass nur die Ansicht der Lagepläne auf dem Bildschirm sichtbar ist. Durch spezielle *Funktions-Buttons,* die auf den Lageplänen positioniert werden können, ist es auch möglich, die Anwendung als Touch-Screen-Applikation zu nutzen. Abbildung 7 zeigt die Anwendung im Vollbild Nachrichtenmodus. Dargestellt ist dort eine Lageplanübersicht des Erdgeschosses eines Gebäudes mit den verschiedenen Meldern. Über die *Funktions-Buttons* kann zwischen den einzelnen Lageplänen der Gebäudeetagen navigiert werden.

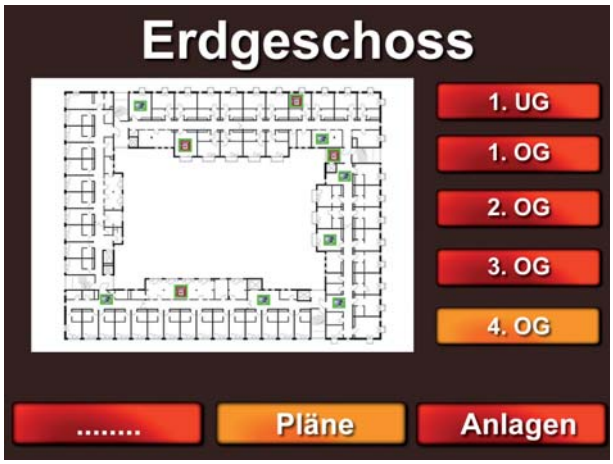

Abb. 7: Clientanwendung im Vollbildmodus

Der Vollbildmodus wird oft bei Terminals eingesetzt, bei dem der Benutzer nicht die zugrunde liegende Windows-Anwendung sehen soll, sondern nur ein Bedienpanel. Einen weiteren Use-Case für den Vollbildmodus stellen Installationen des Wotan-Systems in Privathäusern dar. Hier dienen die Funktions-Buttons nicht primär zum Navigieren zwischen verschiedenen Lageplänen, sondern zum Öffnen der einzelnen Kameradialoge

der Überwachungskameras des Gebäudes. Abbildung 8 zeigt den geöffneten Kamera-Dialog, der sich in ein Bild- und ein Steuerungsteil aufteilt. Der Bildteil des Kamera-Dialoges passt sich automatisch der Bildgröße der Überwachungskamera an. Über die Pfeilbuttons des Kamera-Dialoges kann die Kamera gesteuert werden. Möglich sind je nach Kameramodell auch Zoomfunktionen und Fokussierung. Einige Kameramodelle unterstützen auch voreingestellte Kamerapositionen (*Presets*). Da diese *Presets* im Wotan-System auch als Folgeaktionen gespeichert werden, ist es möglich, ein Kamerabild mit einer voreingestellten Zoom- und Schwenkeinstellung zu öffnen. Ein häufiger Use-Case dieser Funktion in Privathäusern ist z. B. das Öffnen des Bildes der Kamera des Vorderhauses, wenn an der Haustür geklingelt wird. Die Kamera schwenkt dann automatisch auf den Eingangsbereich der Haustür, sodass die klingelnde Person auf dem Display der Wotan-Anwendung dargestellt wird. Der Benutzer kann danach entscheiden, ob er die Tür über die Wotan-Anwendung öffnet oder den Klingelruf abweist.

Abb. 8: Kamera-Dialog

6 Fazit

Das vorgestellte Wotan-System ist eine Gebäudemanagementsoftware mit vielfältigen Einsatzmöglichkeiten in unterschiedlichen Bereichen. Die entwickelte Lösung ist herstellerneutral und hebt sich dadurch von den meisten am Markt verfügbaren Produkten ab. Ein großer Vorteil der Software ist die modulare Struktur auf Basis der Eclipse Rich-Client-Plattform und dem OSGi-Standard. Diese Architektur ermöglicht die flexible Erweiterbarkeit des Systems, ohne bestehende Komponenten anpassen zu müssen. Dadurch können bestehende Funktionen separat erweitert werden oder neue Module und Geräte in das Gesamtsystem integriert werden. Die erstellte Softwarelösung ist so konzipiert, dass sich durch entsprechende Konfigurationen sehr unterschiedliche Anwendungsfälle von Kunden realisieren lassen, die das Facility Management im Rahmen des Gebäudemanagements unterstützen.

Literaturverzeichnis

[1] Richtlinie GEFMA 100-1. http://www.gefma.de/definition.html.
[2] Eclipse Wiki. Eclipse Foundation. http://wiki.eclipse.org/index.php/Rich_Client_Platform.
[3] OSGi Homepage. OSGi Alliance. http://www.osgi.org.
[4] NASA Uses Eclipse for Interplanetary Operations. http://www.eclipse.org/community/casestudies/NASAfinal.pdf.
[5] Kistel, Th. (2007): Erfassung und Fernvisualisierung von Informationen bei Patientenmonitoren; Masterarbeit, TFH Wildau.
[6] Gamma, E./Beck, K. (2004): Eclipse erweitern. Prinzipien, Patterns und Plug-Ins, München: Addison-Wesley.
[7] Vandenhouten, R./Behrens, Th./Selz, M. (2006): Multi-Telemonitoring: Datenerfassung und Aufzeichnung von Vitalparametern und Primärsignalen auf dem PC oder PDA. In: Themenspecial Telematik im Gesundheitswesen, Competence Site/BITKOM.
[8] Vandenhouten, R./Behrens, Th. (2004): Ein Gatewaysystem für telematikbasiertes Gerätemonitoring (A gateway system for telematics-based device monitoring). In: Wissenschaftliche Beiträge der TFH Wildau, Heft 2004.
[9] Vandenhouten, R. (2003): Qualitäts- und Ressourcenmanagementsystem für die Produktion auf der Basis von WWW und WAP, Innovationskatalog 2003, BMWA.

Autoren

Prof. Dr. rer. nat. Ralf Vandenhouten
Technische Fachhochschule Wildau
Fachbereich Ingenieurwesen/Wirtschaftsingenieurwesen
Fachgebiet Telematik
Tel. +49 3375 508-359
ralf.vandenhouten@tfh-wildau.de

Thomas Kistel, M. Eng.
Tel. +49 3375 508-615
thomas.kistel@tfh-wildau.de