# Vergleich von Universal Plug and Play und Bonjour im Kontext ubiquitärer Systeme

Christoph Pohl und Bastian Wollschlaeger

Technische Universität Dresden
Lehrstuhl für Multimediatechnik
01062 Dresden
{christoph.pohl,bastian.wollschlaeger}@mailbox.tu-dresden.de

**Zusammenfassung** Immer mehr Geräte aus dem Haushalt werden mit dem Heimnetzwerk verbunden und erhalten Zugang zum Internet. Eine Herausforderung dabei besteht darin die unterschiedlichsten Geräte miteinander kommunizieren zu lassen, um dem Anwender möglichst viel Arbeit zu erleichtern oder gar ganz abzunehmen. Damit dies möglich werden kann, stehen verschiedene Technologien zur Verfügung, die Geräten automatisch Zugang zum Heimnetzwerk eröffnen. Darunter befinden sich mit UPnP und Bonjour zwei Technologien mit unterschiedlichen Vorgehensweisen, die Geräte ohne Konfigurationsaufwand in ein Netzwerk einbinden. Dieser Artikel vergleicht beide Technologien miteinander und zeigt deren Vor- und Nachteile auf. Des Weiteren wird der Fragestellung nachgegangen, wie es möglich ist verschiedene Standards der dynamischen Geräte- und Serviceerkennung zu koppeln.

## 1 Einleitung und Motivation

Durch die zunehmende Verbreitung von vernetzten Geräten sowohl im Haushalt als auch im öffentlichen Raum steigt die Interaktionsmöglichkeit der Nutzer mit einer heterogenen Ansammlung von Geräten. Zu diesen Geräten zählen sowohl Smartphones, Fernseher und Musikanlagen, als auch Systeme zur Steuerung von beispielsweise Licht und Heizung. Somit nimmt auch der Bedarf nach einer für den Anwender transparenten Konfiguration der Geräte zu, wobei hier die Probleme der dynamischen Geräte- und der Serviceerkennung gelöst werden müssen. Umgebungen, in denen eine Vielzahl von technischen Geräten Informationen austauschen, nennt man ubiquitäre[1] Systeme. Dabei lassen sich öffentliche Netze von den sogenannten Heimnetzen unterscheiden. Dieser Artikel fokussiert sich ausschließlich auf letzteres. Der Kern eines solchen Heimnetzes ist dabei das *Residential Gateway*, welches verschiedene Netzwerkgeräte miteinander verbindet.

Eine Vielzahl von aktuellen Forschungsprojekten befasst sich mit der Entwicklung von sogenannten „Smart Homes“, welche neue Möglichkeiten eröffnen, um das Leben von Menschen mit der Technologie der ubiquitären Systeme zu bereichern. Eine große Herausforderung in solchen Umgebungen ist die dynamische

[1] allgegenwärtig, überall verbreitet

Erkennung von Geräten, da sowohl mobile als auch stationäre Apparate nicht dauerhaft mit dem vorhanden Netzwerk verbunden sein müssen. Smartphones zum Beispiel befinden sich meistens in einer Tasche des jeweiligen Nutzers und verlieren somit spätestens beim Verlassen des Smart Homes ihre Verbindung zum Selbigen. Ein Fernseher hingegen besitzt einen festen Standort, wobei auch hier, da er nicht dauerhaft in Betrieb ist, keine ständige Verbindung zum Netzwerk besteht. Somit wird eine Technologie benötigt, die es ermöglicht Geräte und deren Services dynamisch in ein Netzwerk einzubinden um somit einen Informationsaustausch zu gewährleisten.

Im folgenden Beispiel wird der geschilderte Sachverhalt verdeutlicht: Das Intelligent Bulletin System in [3] ist Teil eines ubiquitären Systems auf einem Universitätscampus und soll dem Nutzer zur richtigen Zeit und am richtigen Ort Zugang zu wichtigen Informationen bieten können. Dabei wird zuerst die Identität des Nutzers festgestellt. Anschließend besteht die Möglichkeit ihm Neuigkeiten zu präsentieren oder ihn über andere wichtige Informationen, die ausschließlich ihn betreffen, zu informieren.

Diese Arbeit stellt eine Vielzahl von wichtigen Technologien kurz vor und fokussiert sich anschließend im Hauptteil auf den Vergleich von UPnP[2] und Bonjour[3]. Dabei werden die genannten Standards detailliert vorgestellt und in Hinsicht auf Vorzüge aber auch Grenzen deren Anwendung verglichen. Abschließend wird ein Ausblick auf die zukünftige Implementierung dieser beiden Technologien in ubiquitäre Systeme gegeben und aufgezeigt, wie eine generelle Zusammenarbeit über Technologiegrenzen realisiert werden kann.

## 2 Ansätze zur dynamischen Geräte- und Service-Erkennung

Dieses Kapitel schafft einen Überblick über aktuelle Technologien zur Gerätebzw. Service-Erkennung. Im Folgenden wird dabei auf die Technologien UPnP, Bonjour, Bluetooth, Jini, SLP und OSGi eingegangen.

### 2.1 Aktuelle Technologien im Überblick

*Universal Plug and Play* – UPnP – ist eine Technologie, die von einem Zusammenschluss von fast 1000 Firmen, darunter Microsoft, Samsung, Sony und Intel, entwickelt wurde. Das UPnP Forum hat sich 1999 gegründet und entwickelte den UPnP-Standard, der 2008 die Version 1.1 erreichte. Das Ziel von UPnP ist es, PCs aller Art sowie intelligente und kabellose Geräte konfigurationslos zu verbinden und zu verknüpfen [3]. Bei Geräten im Zusammenhang mit Audio- und Videoinhalten ist die *Digital Living Network Alliance* (DLNA) für die Entwicklung, Standardisierung und den Test von entsprechenden Geräten zuständig [**?**]. Dabei nutzt UPnP bestehende Internettechnologien wie XML,

[2] Universal Plug and Play
[3] eine Implementierung des Zeroconf-Systems von Apple

HTTP, TCP/IP etc. und eigene Protokolle [1]. So wird zum Suchen von Diensten und Ressourcen sowie zur Bekanntmachung der Präsenz eines Gerätes das *Simple Service Discovery Protocol* (SSDP) verwendet. Das UPnP-Gerätemodell ist hierarchisch aufgebaut und erlaubt es einem Client (*Control Point* genannt), einzelne Subgeräte unabhängig voneinander anzusprechen, nachdem das Wurzelgerät (*Root Device*) gefunden wurde. Die gerätespezifischen Details werden vor der Anwendung verborgen und ermöglichen eine einheitliche Behandlung verschiedenster Gerätetypen [9]. Dieser Prozess ist in Abb. 1 illustriert. In der Spezifikation werden außerdem die Bereiche „Geräte-Adressierung“, „Dienst-Bekanntmachung und Entdeckung“, „Geräte-Kontrolle“, „Ereignisverarbeitung“ und „Darstellung“ behandelt [9]. Weitere Details zu dieser Technologie werden im Punkt 2.2.1 erläutert. In [**?**] wird außerdem auf ein Debian-Paket namens *gupnp-tools* verwiesen, mit dem sich ein beispielhaftes UPnP-Netzwerk unter Linux untersuchen lässt. So kann beispielsweise eine Glühlampe ein- und ausgeschaltet und gedimmt werden.

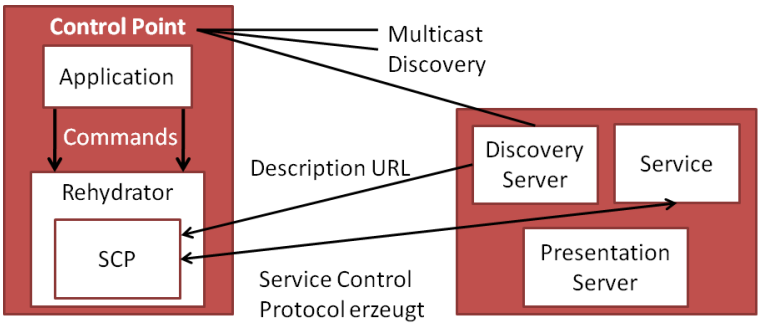


**Abbildung 1.** Interaktion zwischen Client und Service in UPnP, nach [9]

*Bonjour* ist eine Zeroconf-Netzwerkarchitektur und wurde von Apple entwickelt, um Services von Geräten in einem IP-Netzwerk freizugeben und zu entdecken. Es implementiert Multicast DNS (mDNS), DNS-Based Service Discovery (DNS-SD) sowie das Internet Protocol Version 4 (IPv4) [7]. mDNS und DNS-SD werden genutzt um Namen und IP-Adressen ohne separaten DNS-Server zu übersetzen sowie einen Mechanismus bereitzustellen, der das automatische Finden und Bereitstellen von Netzwerkdiensten ermöglicht. mDNS dient dabei als eine Beschreibung für Clients, wie DNS-Anfragen an Multicast-Adressen verschickt werden müssen und DNS-SD ist für die Netzwerkdienste zuständig. In mDNS ist festgelegt, dass die DNS-Top-Level-Domain „.local.“ linklokal ist. Dies bedeutet, dass Anfragen und Antworten, die mit „.local.“ versehen sind, ähnlich wie linklokale IP-Adressen nur im lokalen Netz verarbeitet werden können. Seit Mac OS X 10.2 (August 2002) ist Bonjour fest in das Betriebssystem integriert und findet hauptsächlich Anwendung in dem Browser Safari, einigen Adobe-Programmen, der Musikverwaltungssoftware iTunes sowie in netzwerkfähiger

Hardware wie beispielsweise Druckern. Dabei werden Hard- und Software komplett ohne oder mit sehr wenig Konfiguration erkannt und miteinander verbunden. Bonjour ist auch für Windows erhältlich und ermöglicht damit auch unter diesem Betriebssystem und in heterogenen Netzen eine einfache Vernetzung von bonjour-fähiger Soft- und Hardware. Weitere Details zu dieser Technologie werden im Punkt 2.2.2 erläutert.

*Bluetooth* wurde 1998 von der Special Interest Group (SIG) gemäß des IEEE 802.15.1 Industriestandards entwickelt. Dabei wird es hauptsächlich für die Funkübertragung zwischen Geräten (Punkt-zu-Punkt oder Piconetze) und über kurze Distanzen genutzt. Seit der Version 1.0 gab es eine Vielzahl von Verbesserungen, vor allem im Bereich Geschwindigkeit, Übertragungsentfernung und Sicherheit. Mit der aktuellen Version 4.0 werden drei verschiedene Technologien miteinander vereint. Neben der klassischen Bluetooth-Technik, welche vorrangig für kurze Distanzen genutzt wird, gibt es „Bluetooth High Speed“ und die „Low Energy Wireless Technology“. Ersteres bedient sich der WLAN-Technik und erreicht höhere Datenraten bis theoretisch 24 MBit/s. „Low Energy“ hingegen erreicht nur 1 MBit/s kann aber in Geräten zum Einsatz kommen, welche von einer Knopfzellenbatterie angetrieben werden und in der Regel Monate oder gar Jahre ohne Batteriewechsel laufen können. Bluetooth beinhaltet ein Service Discovery Protocol (SDP) zum Lokalisieren verfügbarer Dienste und zum Feststellen deren Eigenschaften. SDP stellt dabei nur grundlegende Methoden zum Suchen und Erkennen bereit und identifiziert die Dienste über die UUID[4]. Dabei werden Verbindungen zwischen einem Server, welcher eine Liste mit den Eigenschaften der von ihm angebotenen Dienste enthält, und einem Client ermöglicht.

*Jini* wurde im Jahr 1999 durch Sun Microsystems eingeführt und zielt auf eine einfache und konfigurationslose Zusammenarbeit verschiedenster Gerätearten ab. Dabei wird sich der Java-inhärenten Plattformunabhängigkeit bedient. Jini definiert dafür die Begriffe *Service*, *Lookup Server* und *Client*. Lookup Server verwalten hierbei verfügbare Services, die wiederum von Clients genutzt werden. Es ist auch möglich, dass ein Service als Client agiert – als Beispiel sei hier ein Teleskop genannt, welches Bilder für einen PC bereitstellt, selbst aber einen Drucker und seinen Druckservice nutzt [9]. Zusätzlich kann Jini zum Überbrücken von verschiedenen Technologien verwendet werden, indem es mit der OSGi-Technologie kombiniert wird. Diese Einsatzmöglichkeit wird im Punkt 4.2 detaillierter erklärt. Konkrete Implementierungen nutzen die Protokolle UDP und TCP, allerdings ist es auch denkbar andere Transport- und Vermittlungsprotokolle zu verwenden. Die Aktualität dieser Technologie ist jedoch in Frage zu stellen, da das letzte Update im Source-Code Repository mehr als zwei Jahre zurückliegt.

Das *Service Location Protocol* (SLP) ist ein Protokoll zum Auffinden von Diensten in einem TCP/IP Netzwerk. Es wurde von der IETF[5] entwickelt um die Konfiguration vernetzter Clients innerhalb eines lokalen Netzwerks zu verein-

[4] Unique Universal Identifier
[5] Internet Engineering Task Force

fachen. Dabei teilt SLP allen Clients die Verfügbarkeit eines bestimmten Dienstes mit, wobei Anwendungen mit SLP-Unterstützung diese Informationen verarbeiten und somit automatisch konfiguriert werden können. Das Service Location Protocol baut auf ein dreistufiges Konzept auf und teilt die Hosts in folgende Klassen ein: User Agent (UA), Service Agent (SA) und Directory Agent (DA). Um einen Dienst in einem Netz anbieten zu können, muss der Host über einen Service Agent verfügen, welcher den Dienst bei dem Directory Agent registriert. Beide Agents kommunizieren in frei wählbaren Zeitintervallen miteinander, um die Verfügbarkeit und Adressänderungen auf einem aktuellen Stand zu halten. Der Client kann jetzt mit Hilfe des User Agents beim Directory Agent anfragen, ob ein gewünschter Dienst verfügbar ist und gegebenenfalls abfragen, unter welcher Adresse dieser Dienst erreicht werden kann.

Eine weitere Technologie wurde von der *Open Services Gateway Initiative* (OSGi) entwickelt. Die im Jahr 1999 gegründete Organisation entwarf ein Gateway, das der zentrale Einstiegspunkt für ein Heimnetzwerk und dessen Verwaltung sein kann, indem es heterogene Technologien überbrückt. OSGi spezifizierte außerdem ein Java-basiertes, dienst-orientiertes Framework, welches das dynamische Laden und Verwalten von Softwarekomponenten unterstützt. Diese Komponenten, sogenannte *Software Bundles*, stellen im Framework die einzelnen Services dar und sind dort über „BundleContext"-Objekte integriert. Sie veröffentlichen ihren Service, indem sie ein „Service"-Objekt im Framework registrieren. Diese Objekte können nun von anderen Komponenten benutzt werden, um den gewünschten Dienst auszuführen. Dadurch, dass nur die API[6] spezifiziert wurde, konnte sowohl Plattform- als auch Anwendungsunabhängigkeit gewahrt werden [4]. Das OSGi-Gateway arbeitet nach dem Client-Server-Prinzip, wobei in einem Smart Home das Residential Gateway den Server darstellt [11].

### 2.2 Detaillierte Beschreibung von UPnP und Bonjour

Nachdem im letzten Abschnitt eine große Vielfalt an Technologien präsentiert wurde, werden nun detailliert die Technologien UPnP und Bonjour betrachtet und schließlich verglichen.

**2.2.1 Universal Plug-and-Play** UPnP erweitert das Plug-and-Play Konzept zu einem Netzwerkkonzept basierend auf Standard IP [8]. Geräte, die UPnP unterstützen, können nach [1,3] dynamisch einem Netzwerk beitreten, eine IP-Adresse bekommen, ihren Namen verkünden, verfügbare Services bei Anfragen mitteilen, selbst nach Services und Geräten suchen und das Netzwerk ohne Konfiguration verlassen. Diese UPnP Netzwerke bestehen insgesamt aus Geräten, Diensten und Control Points. Geräte sind dabei Objekte, welche die Dienste bereitstellen und weitere verschachtelte Geräte enthalten können. Sie können wiederum von Control Points (Clients) gefunden und genutzt werden [6].

Das Erstellen eines UPnP-Netzes erfolgt schrittweise durch das Hinzufügen von neuen Gräten. Im Folgenden werden die Schritte des UPnP Frameworks

[6] Application Programming Interface

vom Hinzufügen eines Gerätes bis zum Ausführen von dessen Services erläutert. In [3,6,10] finden sich darüber hinausgehende Informationen.

1. **Addressing and Discovery** - Dies ist der erste Schritt im UPnP-Netzwerk. Das Gerät macht seine Anwesenheit den Control Points durch periodische SSDP-Nachrichten bekannt. Für die Adressierung wird eine eindeutige IP-Adresse benötigt. Um diese zu erhalten, wird nach einem DHCP-Server gesucht oder, wenn dieser nicht verfügbar ist, AutoIP (dynamische IP-Adressen-Zuweisung) verwendet. Das Gerät bekommt somit eine IP-Adresse leihweise zugeordnet. Bei Verwendung von AutoIP befindet sich diese im Bereich von 192.168.1.0 bis 192.168.254.255 und wird durch ARP[7]-Probes[8] ermittelt. Eine durch AutoIP erhaltene IP-Adresse kann allerdings nur im lokalen Netzwerk verwendet werden, wodurch das gemeinsame Nutzen eines Dienstes in mehreren Netzen in diesem Fall nicht möglich ist. Wenn nötig kann die Leihfrist der Adresse durch das Gerät auch verlängert werden. Zusätzlich wird von den Geräten verlangt, dass sie, wenn sie mit AutoIP eine Adresse bezogen haben, periodisch nach einem DHCP-Server suchen. Im Erfolgsfall kann ein Gerät durch diesen Server eine neue IP-Adresse beziehen.
   Unter Nutzung der ihm zugewiesenen Adresse kann ein Gerät nun mithilfe des UPnP Discovery Protokolls SSDP (siehe unten) den Control Points seine Dienste verkünden [1]. Gleiches ist möglich, wenn ein Control Point dem Netzwerk hinzugefügt wird – dieser kann nun nach bestimmten Geräten suchen. In beiden Fällen werden die wichtigsten Daten eines Gerätes in einer „Discovery Message“ ausgetauscht. Diese Nachricht enthält beispielsweise den Gerätetyp, einen UUID, einen Verweis auf detailliertere Geräteinformationen (üblicherweise eine URL[9]) und optionale Zustandsvariablen des Geräts – bei einem MP3-Player sind hier unter anderem Liedtitel, Interpret, Genre und Laufzeit denkbar.
2. **Description** - Um mehr über ein Gerät zu erfahren, muss ein Control Point nach der Discovery die Beschreibung des UPnP-Devices laden. Dies geschieht unter Nutzung der URL aus der Discovery Message. Die Informationen sind in XML[10] strukturiert und nach Gerätebeschreibung und Dienstbeschreibung sortiert. Sie enthalten z. B. Herstellerinformationen, Seriennummer, herstellerspezifische URLs, gerätespezifische Zustandsvariablen sowie Beschreibung und Parameter für die verfügbaren Aktionen. Für jeden dieser Dienste werden der Servicetyp, sein Name und URLs für die Beschreibung, Ausführung und Ereignisbehandlung gespeichert.
3. **Control** - Durch die Kenntnis eines Gerätes und seiner Dienste kann ein Control Point mittels dieser Dienste Aktionen aufrufen lassen oder die Werte von Zustandsvariablen des Gerätes durch Polling ermitteln. Der Auf-

[7] Address Resolution Protocol
[8] Ein versuchsweises Ansprechen eines Gerätes mit dieser IP. Antwortet kein Gerät, so ist diese Adresse noch unbesetzt.
[9] Uniform Resource Locator
[10] eXtensible Markup Language

ruf ist hierbei einem RPC[11] ähnlich und wird durch SOAP (siehe unten) ermöglicht. Die Resultate des Dienstes werden danach an den Control Point zurückgeliefert.

4. **Event Notification** - Hier können sich Control Points bei Änderungen von Variablen in Geräten benachrichtigen lassen. Dafür muss die betreffende Variable entsprechend gekennzeichnet sein. Dann kann sie vom Control Point abonniert werden, wodurch dieser vom Dienst benachrichtigt wird, wenn sich der Wert der Variablen ändert. Die Änderungsmeldungen sind in XML strukturiert und können wahlweise per Unicast oder Multicast gesendet werden.
   Als anschauliches Beispiel für Eventing findet sich in [**?**] ein Netzwerk aus Musiksammlung, MP3-Abspieler und Fernbedienung. Der MP3-Abspieler sucht nach im Netzwerk vorhandenen Musikquellen und gibt diese wieder. Um nun von der Fernbedienung gesteuert werden zu können, benötigt die Fernbedienung Informationen über den aktuellen Musiktitel wie Künstler, Album, Titel, Laufzeit etc. Diese Informationen sind im MP3-Abspieler in Zustandsvariablen hinterlegt, welche die Fernbedienung abonnieren kann. Dann erhält sie eine Nachricht vom Abspieler, wenn sich eine der Variablen geändert hat – zum Beispiel weil der Nutzer ein anderes Lied gestartet hat.
5. **Presentation** - Der Präsentationsschritt ermöglicht die Überwachung und Manipulation eines Gerätes durch eine HTML-basierte Nutzungsschnittstelle. Durch die optionale URL in der Discovery Message kann eine Webseite geladen werden, die je nach ihrer Gestalt verschiedene Funktionen des Gerätes zugänglich macht.

Die Vielzahl an verschiedenen Aufgaben für ein UPnP-fähiges Gerät werden durch unterschiedliche Protokolle bewältigt. Der dabei entstehende UPnP-Protokollstack ist in Abb. 2 dargestellt. Die im Protokollstack beteiligten Protokolle werden anschließend näher erläutert.

- **SOAP** (Simple Object Access Protocol) – Dieses Protokoll ist im Rahmen von Webservices weit verbreitet. UPnP nutzt es für RPCs per HTTP und XML in der „Control“-Phase.
- **SSDP** (Simple Service Discovery Protocol) – SSDP beschreibt die Suche nach Services und definiert, wie Control Points benötigte Ressourcen suchen und Geräte ihre Präsenz den Control Points bekanntmachen
- **GENA** (General Event Notification Architecture) – Durch GENA wird das Senden von Benachrichtigungen (Notifications) mit HTTP über TCP/IP und Multicast UDP definiert und somit der Schritt „Event Notification“ unterstützt.

**2.2.2 Bonjour** Bei *Bonjour* handelt es sich um eine Zeroconf-Technologie, bei der möglichst wenig Konfigurationsaufwand betrieben werden soll, bevor sich Geräte gegenseitig erkennen und ihre Services bereitstellen können. Nutzer

[11] Remote Procedure Call

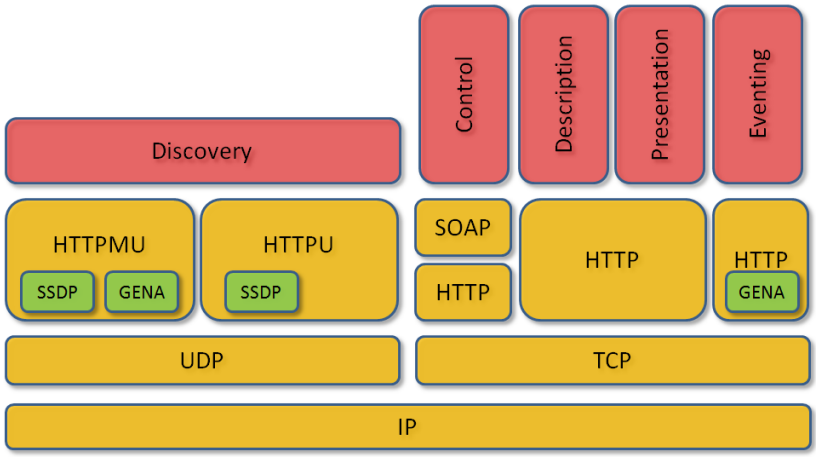


**Abbildung 2.** Der Protokollstack von UPnP, nach [3] und [10]

in einem Netzwerk sollen nicht länger IP-Adressen und Host-Namen manuell vergeben oder durch explizites Angeben eines Namens nach Services suchen müssen. Es soll ermöglicht werden, dass durch eine simple Anfrage eine Auflistung der zur Verfügung stehenden Services präsentiert werden. Um dies zu gewährleisten hat die ZEROCONF Working Group, welche Teil der IETF ist, Voraussetzungen festgelegt, welche sich in drei Hauptaufgaben unterteilen lässt:

- Adressierung (Zuweisung der IP-Adressen zu deren Hosts)
- Benennung (Vergabe von Namen an die Hosts anstelle der IP-Adressen)
- Serviceermittlung (automatisches Auffinden von Services im Netzwerk)

Für jeden dieser drei Schritte besitzt Bonjour eine vollautomatische Lösung, die in den folgenden Abschnitten näher beschrieben wird.

Bei der **Adressierung** wird eine zufällige Netzwerkadresse ausgewählt und mit Hilfe eines ARP-Requests geprüft, ob diese in Benutzung oder frei ist. Dieses Verfahren nennt sich „self-assigned link-local addressing“ und ist sowohl in IPv4 als auch in IPv6 implementiert. Dabei ist hinzuzufügen, dass beide Hosts in dem selben lokalen Netzwerk sind, sobald alle Nutzdaten der Sicherungsschicht unverändert an das gewünschte Ziel kommen. Dies hat zur Folge, dass kein IP-Router mit dem Datenaustausch zweier Hosts in Berührung kommt.

Bei der **Benennung** wird Multicast DNS (mDNS) verwendet und um einen mDNSResponder[12] erweitert. Bei mDNS werden DNS-Anfragen mittels IP-Multicast an das lokale Netzwerk versendet. Diese werden dann von jedem Service oder Gerät überprüft und bei einer Übereinstimmung wird eine DNS-Response mit der eigenen Adresse zurückgeschickt. Aufgrund der Tatsache, dass die DNS-Anfragen an eine Multicast-Adresse geschickt werden, ist kein DNS-Server zur Beantwortung nötig. Der anfangs erwähnte mDNSResponder

[12] interpretiert und beantwortet mDNS-Anfragen

entlastet die Anwendung um die Notwendigkeit der Interpretation und Beantwortung von mDNS-Nachrichten. Sobald ein Service im mDNSResponder registriert ist, werden zukünftige Anfragen nur noch an das bestimmte Gerät gesendet. Diese Registrierung erfolgt über die Bonjour APIs. Diese sind für die Services verfügbar, welche beispielsweise auf einem Host-Computer laufen. Geräte wie Drucker, welche ebenfalls Services bereitstellen können, müssen den mDNSResponder implementiert haben um Anfragen behandeln zu können. Damit die Zuweisung eines Namens zu einer IP-Adresse ohne Konflikte abläuft, muss der Name eineindeutig sein. Im Gegensatz zu herkömmlichen DNS-Host-Namen hat der lokale Name ausschließlich im LAN eine Bedeutung. Ähnlich wie bei Adressierung kann ein Name frei gewählt werden. Ist dieser schon vorhanden, muss ein neuer gesucht werden. Bei Geräten ist es üblich, an den Namen eine Zahl anzufügen, sobald der eigene Name schon im Netzwerk vorhanden ist. Bei Services benennt Bonjour selbst den Namen um, wenn es zu einem Konflikt kommen sollte.

Ein weiteres Element von Bonjour ist die **Serviceermittlung**. Mit Hilfe dieser wird es Anwendungen ermöglicht alle zur Verfügung stehenden Instanzen einer bestimmten Art von Dienstleistung zu finden und eine Liste der benannten Services aufrecht zu halten. Der Anwendung steht dann sofort IP-Adresse und Port-Nummer des gewählten Services zur Verfügung. Dabei werden die Liste der zur Verfügung stehenden Services und die dazugehörigen Adressen getrennt voneinander verwaltet, sodass die Services dynamisch verlagert werden können ohne viel Netzwerkverkehr zu verursachen. Um eine bestimmte Art Dienstleistung zu erhalten, wird eine mDNS-Anfrage verschickt und jedes Gerät, welches diesen Service bereitstellen kann, antwortet mit seinem Namen. Das Resultat ist eine Liste von zur Verfügung stehenden Services. Dieses Verfahren nennt sich „Browsing“ und unterscheidet sich von dem herkömmlichen gerätezentrierten Ansatz. Anstelle jedes einzelne Gerät zu fragen, welche Services es bereitstellt, wird eine lokale Anfrage gestellt, welche Geräte einen speziellen Service zur Verfügung stellen. Dies spart Zeit und Ressourcen, denn der servicezentrierte Ansatz sendet nur eine Anfrage und generiert nur relevante Antworten. Des Weiteren werden ausschließlich die Service-Namen gespeichert, sodass Anwendungen bei einer Änderung der IP-Adresse oder Port-Nummer des Services trotzdem auf ihn zugreifen können.

Um den Overhead der Zeroconf-Technologie bzw. die Belastung des Netzwerks zu minimieren nutzt Bonjour Mechanismen, die im Folgenden kurz vorgestellt werden: Beim Caching werden die gesammelten Informationen über verfügbare Services abgespeichert und müssen bei einem weiteren Aufruf nicht noch einmal komplett neu gesucht werden. Dabei sind die mDNSResponder für die Aktualität dieser Liste bezüglich der Adressen verantwortlich. Ein weiterer Mechanismus ist das Unterdrücken von schon erhaltenen DNS-Antworten. Dabei wird bei einer DNS-Anfrage eine Liste von schon erhaltenen Antworten mitgeschickt, sodass Geräte wissen, dass sie nicht erneut antworten müssen.

## 3 Vergleichende Betrachtung von UPnP und Bonjour

Sowohl UPnP als auch Bonjour wurden entwickelt, um Geräte und deren Services dynamisch zu erkennen. Trotzdem unterscheiden sich beide in einigen Punkten. In diesem Abschnitt werden die Technologien unter folgenden Kriterien verglichen:

**Zuweisung von Adressen** – Nach welchen Mechanismen funktioniert die Adresszuweisung? Können vorhandene DHCP-Server verwendet werden?

**Art der Dienstregistrierung** – Werden alle verfügbaren Dienste eines Netzes zentral in einem Dienstverzeichnis gespeichert oder muss jeder Nutzer selbst nach passenden Diensten suchen?

**Form der Dienstbeschreibung** – Können die von einem Gerät angebotenen Dienste einheitlich beschrieben werden?

**Kommunikation zwischen Geräten** – Auf welche Art und Weise kommunizieren einzelne Geräte miteinander? Werden standardisierte Spezialprotokolle oder generische Protokolle verwendet?

**Sicherheitsmechanismen** Existieren Sicherheitsmechanismen in den vorgestellten Standards/Technologien? (Sicherheit ist hier im Sinne von Security zu verstehen)

**Reichweite** – In welchem Rahmen stehen Services zur Verfügung? Ist die Nutzung auf das lokale Netz beschränkt oder können auch entfernte Services genutzt werden?

**Verwendete Protokolle** – Welche Protokolle verwendet die Technologie?

**Vorwiegende Verbreitung** – Auf welchen Plattformen und in welchen Branchen wird die Technologie am häufigsten eingesetzt?

Tabelle 1 gibt einen Überblick über die Vergleichskriterien und wird im Folgenden erläutert.

Bei der Adressierung von Geräten werden bei beiden Technologien an sich keine DHCP-Server benötigt, um freie IP-Adressen zu vergeben. UPnP kann jedoch vorhandene DHCP / DNS-Server optional nutzen. Im Gegenzug bietet Bonjour zusätzlich den Vorteil Service-Namen anstelle von IP-Adressen zu vergeben, wohingegen UPnP nur den im Gerät vorhandenen *Friendly Name* als Alternative zu IP-Adressen nutzen kann.

Die Dienstregistrierung wird von beide Technologien unterschiedlich durchgeführt. Die verfügbaren Dienste eines UPnP-Netzes werden nicht zentral, sondern durch jedes Gerät einzeln gespeichert. Bei Bonjour wird eine Registrierung der Services auf einem Host-Computer vorgenommen. Services, welche auf anderen Geräten wie beispielsweise einem Drucker laufen, müssen den mDNS-Responder implementiert haben, welcher auf mDNS-Anfragen reagiert. Dies ist ganz klar ein Vorteil für UPnP, da somit jederzeit eine Kommunikation zwischen den Geräten möglich ist.

Bei UPnP können sowohl Dienste als auch Geräte einheitlich als XML-Dokument beschrieben werden. Somit ist gewährleistet, dass die Beschreibungen strukturiert und sowohl menschen- als auch maschinenlesbar sind. Auf das Beschreibungsdokument wird in der Discovery Message eines Gerätes per URL

**Tabelle 1.** Gegenüberstellung von UPnP und Bonjour

| **Kriterium** | **UPnP** | **Bonjour** |
|---|---|---|
| Zuweisung von Adressen | AutoIP oder mittels DHCP / DNS (wenn verfügbar) | self-assigned link-local addressing |
| Art der Dienstregistrierung | dezentral | zentral |
| Form der Dienstbeschreibung | XML | keine |
| Kommunikation zwischen Geräten | standardisierte *Device Control Protocols* (DCPs) | standardisierte Netzwerkprotokolle |
| Reichweite | lokales Netz (LAN) | lokales Netz (LAN) |
| Verwendete Protokolle | HTTPMU, HTTPU, SOAP, SSDP, GENA, UDP, TCP, IP | IPv4, IPv6, mDNS, DNS-SD |
| Vorwiegende Verbreitung | Microsoft Windows, Unterhaltungselektronik | Mac OS, teilweise Linux |

verwiesen. Bonjour dagegen stellt lediglich eine Liste der zur Verfügung stehenden Services, sowie die dazugehörigen IP-Adressen und Port-Nummern bereit. Somit ist es im Gegensatz zu UPnP nur eingeschränkt möglich die gewonnen Informationen bei Bedarf weiterzuverarbeiten.

UPnP-fähige Geräte kommunizieren mit standardisierten *Device Control Protocols* (DCPs), die – ähnlich wie internetbasierte Kommunikation – deklarativ sind, in XML ausgedrückt und per HTTP übertragen werden [1]. Diese werden von speziellen technischen Arbeitsgruppen erarbeitet und müssen den UPnP-Standardisierungsprozess durchlaufen. Bonjour hingegen verwendet standardisierte, IP-basierte Netzwerkprotokolle. Dies hat zur Folge, dass neu erscheinende Geräteklassen mittels Bonjour sofort erkannt werden, aber nicht zwangsläufig miteinander kommunizieren können. Bei der Verwendung von UPnP hingegen wird sowohl die Erkennung, als auch die Kommunikation gewährleistet, allerdings muss mit einer Verzögerung der Unterstützung gerechnet werden, da die Protokolle für die Geräteklasse wie oben beschrieben zuvor einen Standardisierungsprozess durchlaufen müssen.

Wenn ein UPnP-Gerät durch AutoIP dynamisch seine IP-Adresse zugewiesen bekommt, so ist diese nur für das lokale Netz (LAN) gültig. Dies gilt auch für Bonjour. Somit kann ein Dienst nicht ohne Weiteres in mehreren Netzen genutzt werden.

Wie in der Tabelle ersichtlich setzen beide Technologien vor allem auf die Protokolle der im Internet üblichen TCP/IP-Kommunikation. Universal Plug and Play verwendet zusätzlich Protokolle für die Unterstützung von Webservices.

UPnP ist bei Windows dominierend, hat allerdings bei Mac OS X und Linux durch die Verbreitung von Bonjour kaum Marktanteile. Ansonsten ist UPnP vor allem in der Unterhaltungselektronik weit verbreitet, was die Existenz der Unterarten UPnP/AV (Audio/Video) und DLNA belegt und berechtigt. Bonjour

dagegen ist derzeit hauptsächlich in netzwerkfähigen Druckern implementiert und in den von Apple selbst entwickelten Produkten wie iTunes, iPhone, AppleTV und Airport-Express, welche das Streamen von Bildern, Videos und Musik untereinander ermöglichen, verbreitet.

# 4 Kopplung verschiedener Standards

## 4.1 Überbrücken von verschiedenen Technologien durch OSGi

In Punkt 2 wurde dargelegt, dass eine Vielzahl von verschiedenen Technologien existiert, um Geräte in einem Heimnetzwerk zu verbinden. Da sich zum jetzigen Zeitpunkt keine Technologie entscheidend durchgesetzt hat, ist es nötig die verschiedenen Technologien geeignet zu verbinden, um ein heterogenes Gesamtnetzwerk aufzubauen. In Abb. 3 wird ein mögliches Beispielszenario dargestellt, welches aus [4] stammt. Ein Haus besitzt hier mehrere Subnetze, beispielsweise Jini-, UPnP-, USB- und Ethernet-Netze. Die Autoren schlagen vor, mit der OSGi-Technologie eine Brücke für andere Transportmechanismen zu kreieren. Dabei werden die einzelnen Techniken auf OSGi gemappt und somit vereinigt. Konkret wird in der Arbeit auf UPnP und Jini eingegangen. Für jede zu unterstützende Technologie muss ein Treiber geschrieben werden, der eine X-to-OSGi Transformation und eine OSGi-to-X Transformation beinhaltet. Diese Transformationen stellen ein Import-/Export-Modell dar, in dem native Services in das OSGi-Netz importiert und für andere Technologien wieder exportiert werden. Somit ist das zentrale OSGi-Netz eine Art Proxy für alle spezifischen Subnetze.

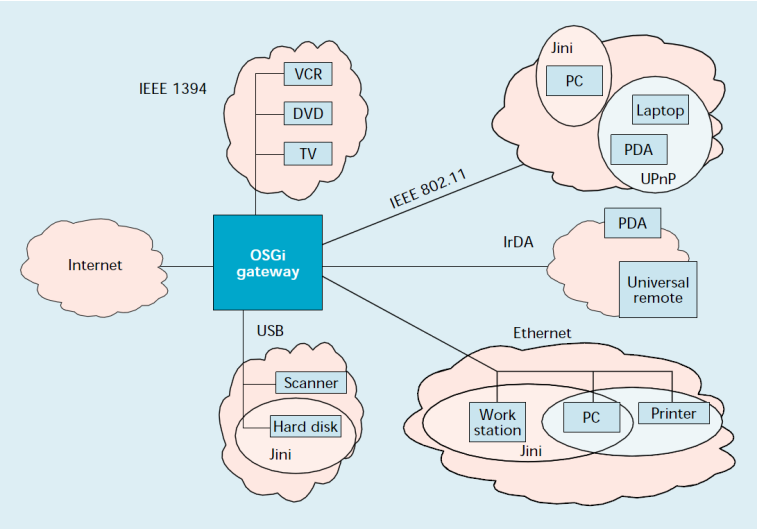


**Abbildung 3.** Beispiel einer OSGi-Heimnetzarchitektur mit heterogenen Technologien, entnommen aus [4]

### 4.2 Verbinden von einzelnen OSGi-Subnetzen

Wie oben erläutert, kann die OSGi-Technologie benutzt werden, um verschiedene Verbindungsstandards in Heimnetzwerken miteinander zu verknüpfen. Allerdings sorgt der zentralisierte Mechanismus von OSGi dafür, dass die verwalteten Dienste nur lokal, also nicht von anderen, entfernten Frameworks zugänglich sind. Im Folgenden werden zwei Ansätze vorgestellt, um getrennte OSGi-Netzwerke zu verbinden. Zuerst wird dabei die bereits in Abschnitt 2 erklärte Jini-Technologie verwendet und schließlich eine Variante unter Nutzung von *Devices Profile for Web Services* (DPWS) vorgestellt.

Ebenso wie OSGi setzt auch Jini auf die Java-Technologie auf. In [11] und [12] wird ein Ansatz diskutiert, mit Jini verteilte OSGi-Dienste zu überbrücken - der Jini-Bridge Service. Dieser Dienst ist ein üblicher OSGi-Service, der gleichzeitig der Jini-Spezifikation entspricht und dem OSGi-Framework als eine Art Plug-in hinzugefügt wird. So können andere OSGi-Services im Jini-Lookup-Service registriert und dadurch auch entfernten OSGi-Frameworks zugänglich gemacht werden. Die spezielle in der Arbeit verwendete Lösung nutzt für die Kommunikation zwischen lokalem OSGi-Service und dem Jini-Service-Proxy die *Remote Method Invocation* (RMI), welche als Java-Technologie auf einfache Art und Weise mit OSGi und Jini kombinierbar ist.

Eine andere Möglichkeit, OSGi-Netze zu verknüpfen, stellt DPWS dar. Diese Technologie wurde im Rahmen von *OASIS Web Services Discovery and Web Services Devices Profile* (WS-DD) im Mai 2004 veröffentlicht. DPWS wurde von UPnP inspiriert, zielt auf eingebettete Systeme mit begrenzten Ressourcen ab und definiert eine minimale Menge von Standards und Spezifikationen für eine Web-Service-basierte Kommunikation zwischen diesen Systemen. Dabei werden die Technologien SOAP-over-UDP und SOAP in Verbindung mit HTTP genutzt. Für verschiedene Aufgaben nutzen DPWS-Devices unterschiedliche Nachrichtentypen. Beispielsweise melden sich Geräte mit „Hello“ und „Bye“ im Netzwerk an bzw. ab, suchen mit „Probe“ nach passenden Diensten und rufen diese mit „Invocation“ auf. Weiterhin unterstützen Windows Vista und Windows 7 DPWS nativ, was diese Technologie zusätzlich attraktiver macht [4].

In [5] wurde ein Ansatz vorgestellt, um DPWS und OSGi zu kombinieren. Die Abbildungen 4 und 5 skizzieren die in dieser Arbeit verwendete Architektur. Aus Gründen der Übersichtlichkeit wurde die Übersichtsgrafik in diese beiden Teile geteilt. Die Terminologie unterscheidet dabei zwischen einem Client, welcher einen Dienst nutzen möchte, und einem Server, der diesen Dienst anbietet. Im Folgenden wird der Ablauf dargestellt, wie ein OSGi-Bundle B auf Clientseite des OSGi-Frameworks die Dienste eines entfernten Bundles A oder eines nativen DPWS-Gerätes A' durch eine OSGi-DPWS-Brücke nutzen kann. Das OSGi-Bundle A im Server wird durch einen Skeleton-Generator mit einem DPWS-Device A versehen und meldet sich damit im DPWS-Netzwerk an (Schritte 1 bis 4 in Abb. 5; Schritt 5 in Abb. 4). Ein Proxy-Generator im Client erstellt dann bei Bedarf einen DPWS-Proxy A und die für die Interaktion mit ihm benötigten Interfaces (Schritte 6 und 7). Somit kann das Bundle B im Client sowohl die OSGi-Funktionalität von A, als auch die Dienste eines nati-

ven DPWS-Gerätes A' nutzen (Schritt 8). Kernkomponenten der Brücke sind der Skeleton-Generator im Server und der Proxy-Generator im Client. Dabei dient der Skeleton-Generator dazu, einen OSGi-Service durch einen entsprechenden Skeleton-Service in das DPWS-Netzwerk einzufügen. Dieser Skeleton-Service analysiert einlaufende Anfragen und ruft den korrespondierenden OSGi-Service auf. Anschließend werden die Ergebnisse in eine DPWS-Antwort verpackt und verschickt. Der Proxy-Generator erzeugt auf Clientseite einen lokalen Proxy für die entfernten Services. Dieser Proxy ist ein OSGi-Service und bietet die gleichen Dienste an wie der entfernte OSGi-Service. Ähnlich dem Skeleton-Service ist auch der Proxy-Service für das Umwandeln eingehender Nachrichten zuständig, nur dass in diesem Fall DPWS-Nachrichten in OSGi-Aufrufe umgesetzt werden.

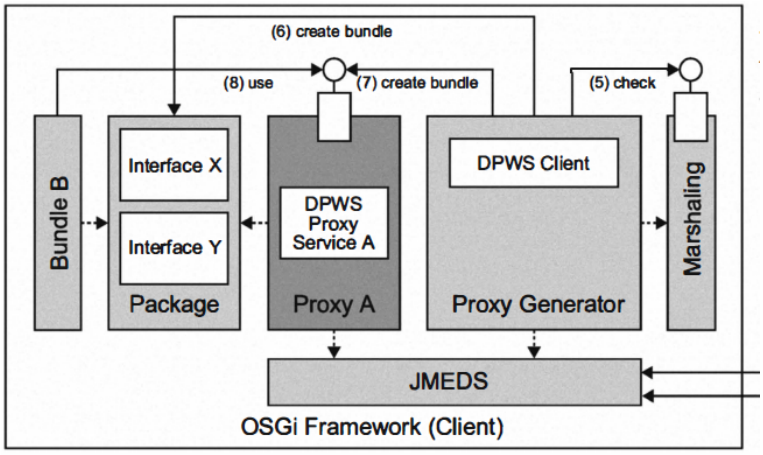


**Abbildung 4.** Modell der Kombination der DPWS- und der OSGi-Technologie (linke Seite), entnommen aus [5]

## 5 Fazit

Sowohl Universal Plug and Play als auch Bonjour verfolgen dasselbe Ziel – so einfach und schnell wie möglich ein Netzwerk zwischen Geräten aufzubauen. Bonjour ist eine geschlossene, Apple-eigene Erweiterung des offenen Standards Zeroconf und ermöglicht es ohne Umwege auch neuartige Geräte mit bestehenden zu verbinden. Bei UPnP hingegen wird bei der Erscheinung einer neuen Geräteklasse ein Forum ins Leben gerufen, um eigene geeignete Protokolle zu gewährleisten und somit die Konnektivität zu gewährleisten. Beide Wege haben ihre Vor- und Nachteile und auch in Zukunft wird der Wettkampf beider Technologien kein Ende finden. Bonjour zielt darauf ab, möglichst viele verschiedene IP-basierte Netzwerkprotokolle zu unterstützen, wohingegen UPnP dafür entwickelt wurde gerätespezifische Aufgaben zu lösen. Des Weiteren bietet UPnP

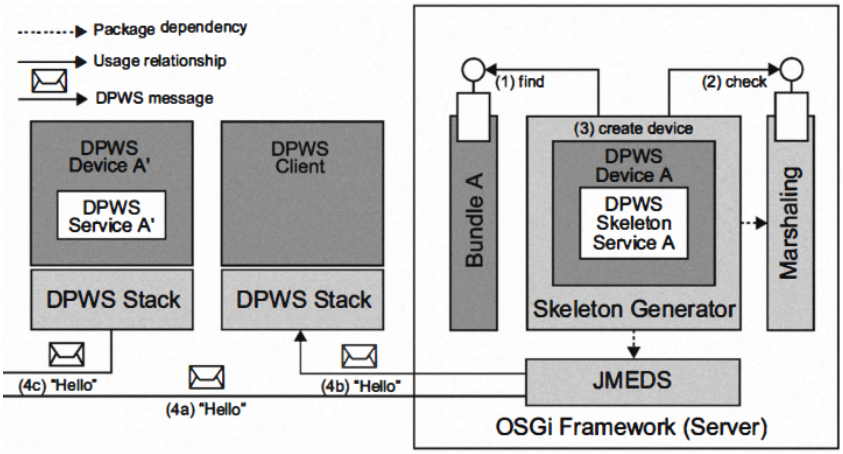


**Abbildung 5.** Modell der Kombination der DPWS- und der OSGi-Technologie (rechte Seite), entnommen aus [5]

nur eine eingeschränkte Lösung für alle IP-basierten Netzwerkprotokolle, was als Nachteil angesehen werden kann.

Aufgrund der Tatsache, dass große Unternehmen verschiedene Technologien bereitstellen und stets hinter ihrer eigenen stehen werden, wird es auch in Zukunft Rivalitäten zwischen Apple und der Allianz von über 550 Unternehmen geben. Beide Technologien haben ihre Stärken und werden wohl auch zukünftig ihre Daseinsberechtigung haben und gerade deshalb werden vielleicht in absehbarer Zeit Entwickler neuer Geräte sowohl UPnP und Bonjour in ihre Technik implementieren.

## 6 Ausblick

Dieser Artikel hat gezeigt, dass die steigende Verbreitung von kommunikationsfähigen Geräten mithilfe der vorgestellten Geräte- und Service-Erkennungsmethoden ubiquitäre Umgebungen ermöglichen. Durch die konfigurationslose Erkennung von dynamisch hinzugefügten Geräten können auch unausgebildete Personen ihr persönliches Heimnetzwerk zusammenstellen. Für die Wahl des verwendeten Netzes existiert bis heute kein einheitlicher Standard oder Marktführer. Aus diesem Grund wird es auch wichtig sein, die verschiedenen Technologien zu kombinieren. Für die Ebene eines Heimnetzes kann dies wie gezeigt mit OSGi gelingen. Wird nun der Fokus ausgeweitet und auch die Verknüpfung von mehreren Heimnetzwerken angestrebt, so müssen die OSGi-Netze ebenfalls überbrückt werden. Dazu wurden Möglichkeiten mit den Technologien Jini und DPWS beschrieben.

Eine weitere Herausforderung im Kontext der dynamischen Geräte- und Service-Erkennung ist die Erkennung der Semantik der Services, welche zur Zeit

nicht übertragen wird. Momentan werden die Entscheidungen nur auf Basis der Parametertypen getroffen – Abhilfe könnten hier zum Beispiel Ontologien schaffen. Ein weitere Ansatz, um dieses Problem zu beheben, wird in [2] beschrieben. Die Autoren entwickelten ein System, MobiEureka, um auf Basis von Nebenbedingungen kompatible Dienste nach ihrer Relevanz zu ordnen.

## Literatur


1. Upnp forum (2011), http://upnp.org/
2. Al-Masri, E., Mahmoud, Q.: Device-aware discovery and ranking of mobile services. In: Consumer Communications and Networking Conference, 2009. CCNC 2009. 6th IEEE. pp. 1 –5 (jan 2009)
3. Chen, W., Kuo, S.Y., Chao, H.C.: Service integration with upnp agent for an ubiquitous home environment. Information Systems Frontiers 11(5), 483–490 (Jun 2008), http://www.springerlink.com/index/10.1007/s10796-008-9122-3
4. Dobrev, P., Famolari, D., Kurzke, C., Miller, B.: Device and service discovery in home networks with osgi. IEEE Communications Magazine 40(8), 86–92 (Aug 2002), http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1024420
5. Dohndorf, O., Kruger, J., Krumm, H., Fiehe, C., Litvina, A., Luck, I., Stewing, F.J.: Towards the web of things: Using dpws to bridge isolated osgi platforms. In: Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on. pp. 720 –725 (29 2010-april 2 2010)
6. de Freitas, G.B., Teixeira, C.A.C.: Ubiquitous services in home networks offered through digital tv. In: Proceedings of the 2009 ACM symposium on Applied Computing. pp. 1834–1838. SAC '09, ACM, New York, NY, USA (2009), http://doi.acm.org/10.1145/1529282.1529691
7. Inc., A.: Bonjour overview. Tech. rep. (2006)
8. Rhee, S.H., Yang, S.K., Park, S.J., Chun, J.H., Park, J.A.: Upnp home networking-based ieee1394 digital home appliances control. In: Yu, J., Lin, X., Lu, H., Zhang, Y. (eds.) Advanced Web Technologies and Applications, Lecture Notes in Computer Science, vol. 3007, pp. 457–466. Springer Berlin / Heidelberg (2004), http://dx.doi.org/10.1007/978-3-540-24655-8_49, 10.1007/978-3-540-24655-8_49
9. Richard, G.: Service advertisement and discovery: enabling universal device cooperation. IEEE Internet Computing 4(5), 18–26 (2000), http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=877482
10. UPnP-Forum: Upnp device architecture 1.1. Tech. rep., UPnP-Forum (October 2008)
11. Yiqin, L., Xiaodong, Y., Chunhong, Y.: Service sharing between osgi-based residential gateways in distributed network environment. In: Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on. pp. 1 –4 (dec 2009)
12. Yiqin, L., Yao, Y., Yingkai, S., Xiaodong, Y.: An approach to service integration in the osgi architecture of home networks. In: Communication Systems, 2008. ICCS 2008. 11th IEEE Singapore International Conference on. pp. 756 –760 (nov 2008)