MIDDLEWARE IM VERGLEICH 5

Gegen Ende dieser Arbeit wird dieses Kapitel einer Gegenüberstellung der vorgestellten Middleware-Konzepte gewidmet und damit deren Gemeinsamkeiten und Unterschiede zur Sprache gebracht.
Zu Beginn dieses Kapitels werden im ersten Abschnitt die vorgestellten Middleware-Technologien nach deren wichtigsten Eigenschaften, die ersten Unterschiede bieten, untereinander verglichen. Um einen Überblick über ihre Zusammenarbeit mit den darunter liegenden Kommunikationsschichten zu vermitteln, wird im darauf folgenden Abschnitt ihre Eignung mit zurzeit aktuellen drahtlosen Kommunikationsprotokollen diskutiert. Im letzten Abschnitt werden sie anhand ihrer wichtigsten Kriterien erläutert. Anhand dieser Merkmale werden sie anschließend in einer Tabelle zusammenfassend dargestellt und beschrieben.

5.1 Vergleich der Middleware untereinander

- **Jini vs. UPnP:** Obwohl die Technologien Jini und UPnP grundsätzlich das gleiche Ziel verfolgen, gibt es wesentliche Unterschiede. Der größte Unterschied liegt bei der Implementation. Während UPnP dem Entwickler darin freie Hand lässt, welche Programmiersprache er verwenden will, baut Jini auf Java RMI auf. Ein Jini System ist also immer in Java geschrieben.
  Somit kommt man zu einem weiteren großen Unterschied zwischen Jini und UPnP: Beide Technologien sind zwar plattformunabhängig. Jini erreicht dies aber dadurch, dass es auf Java basiert. Java bringt die Eigenschaft des ‚plattformunabhängigen' von sich aus mit. Jini arbeitet also auf einem Level weit über dem Betriebssystem und den Verbindungsprotokollen. Im Gegensatz dazu muss UPnP aufgrund der offenen Programmiersprache hier einen anderen Weg beschreiten. Es wird direkt auf Protokollebene gearbeitet [Microsoft-UPnP 2003] (siehe Kapitel 4 UPnP-Abschnitt). Da UPnP auf den etablierten und weit verbreiteten TCP/IP Standard aufbaut, der heute auf allen Systemen zur Verfügung steht, erreicht es die gewünschte Flexibilität.

  Jini soll den Entwicklungsaufwand deutlich reduzieren. Allerdings wird dieser Vorteil durch den größeren Bedarf an Ressourcen erkauft. Bei UPnP dagegen soll der Bedarf an Ressourcen deutlich geringer sein, da die Kommunikation bei dieser Technologie hauptsächlich auf dem Verschicken von XML-Dateien basiert.

  Ein sehr wichtiger Punkt ist auch die Voraussetzung an die Netzwerke, welche die Technologien verlangen. Während Jini voraussetzt, dass das Netzwerk konfiguriert ist, das heißt, dass es einen DHCP-Server gibt, wählt UPnP einen anderen Ansatz. UPnP benutzt im Idealfall auch einen DHCP-Server, kann jedoch, falls kein solcher vorhanden ist, mittels eines AutoIP genannten Verfahrens das Problem der IP-Adressen auch selbst lösen [UPnP 2003].

Zusammengefasst stehen die Technologien von UPnP und Jini trotz gewisser Gemeinsamkeiten für unterschiedliche Philosophien:

- UPnP stellt den PC in das Zentrum eines jeden Zukunftshaushalts. Der PC kontrolliert alles, vom Stromverbrauch über den Anrufbeantworter bis hin zur Übertragung von Videodaten [UPnP2003].

- Jini verfolgt dagegen einen radikal dezentralen Ansatz, bei dem Kontrolle und "Intelligenz" über das ganze Netzwerk verteilt sind, explizit keine Zentrale existiert und man alle Geräte von überall kontrollieren kann.

❖ **CORBA vs. Jini:** In mancher Hinsicht versucht Jini im Vergleich zu CORBA und den CORBA-Services, das Rad neu zu erfinden. Jini und CORBA sind unterschiedliche Architekturen. Die eine basiert auf dem P2P-Prinzip und die andere auf dem C/S-Prinzip. Aber am wichtigsten ist, dass CORBA das spezifische Protokoll GIOP definiert [OMG 2003]. Im Gegensatz zu Jini ist die Kommunikation der Proxies mit ihrem Backend nicht festgelegt. Es könnten Verfahren wie Sockets, RMI oder CORBA selbst verwendet werden [Sun 2003]. Ein großer Unterschied zwischen den beiden Technologien ist auch: CORBA naming und trading service liefert eine Referenz auf das entfernte Objekt, das die Methoden beinhaltet, und Jini-LUS liefert ein Proxy-Objekt by value, das ohne Server-Unterstützung die Methoden ausführt. Außerdem wird Objekt-Interface bei CORBA durch IDL definiert, die in verschiedenen Programmiersprachen geschrieben werden kann. Im Vergleich zu Jini kommt nur Java als Programmiersprache in Frage, genauso wie bei JacORB. Weiterhin werden bei CORBA nur primitive Objekt-Typen und bei Jini Object by value unterstützt.

❖ **ViaXML vs. Jini:** Bei beiden Systemen nutzen die Clients die Dienste, um ihre Aufgabe zu erfüllen. Wie in der realen Welt, muss ein Kunde zuerst in Erfahrung bringen, wo er den geeigneten Dienstleister findet. Jini nutzt dazu den LUS und ViaXML den XML-RPC oder SOAP, das den UDDI-Server[72] beansprucht. Die wesentlichen Unterschiede bestehen darin, dass Jini auf Java Interfaces und RMI aufsetzt und ViaXML hingegen das Internet benötigt, eine C/S-Technologie ist, während Jini eine P2P-Technologie ist.

❖ **ViaXML und andere Middleware:** Im Vergleich zum Transport binärer Daten bei CORBA/JacORB führt die Verwendung von XML-RPC bei ViaXML dazu, dass die übertragene Datenmenge größer ist. Bei geringerer Bandbreite könnte sich dies negativ auf die Übertragungszeit auswirken. Weiterhin müssen auf der Senderseite die XML-Dateien generiert und auf der Empfängerseite wieder geparst werden. Dieses trägt am meisten zu den Performance-Verlusten gegenüber der Verwendung von

[72] UDDI (*Universal Description, Discovery and Integration)* ist eine Industrie-Initiative von u. a. Ariba, IBM und Microsoft. Sie definiert plattformunabhänige, offene Standards für die Beschreibung von Services.

CORBA/JacORB bei. Bei ViaXML bringt die Benutzug der XML beim Daten-Transport zwei gewichtige Vorteile. Erstens sind die übertragenen Nachrichten von Entwicklern lesbar, was in der Entwicklungsphase und in Fehlerfällen günstig ist. Zweitens ermöglicht die Verwendung von XML im Gegensatz zu einem festen Nachrichten- und Typsystem flexible Erweiterungen.
Ein direkter Vergleich ist schwierig, da sich JacORB/ORB, Jini/RMI und UPnP/SSDP auf ein verteiltes Objektmodell konzentrieren, ViaXML/SOAP hingegen grundsätzlich den Austausch von Nachrichten zwischen Clients und Servern ermöglicht.

5.2 Eignung der Middleware und der drahtlosen Protokolle

Alle in dieser Arbeit vorgestellten Middleware-Technologien (Jini, UPnP, ViaXML, JacORB) können in Zusammenarbeit mit den drahtgebundenen, in Kapitel 2 vorgestellten, Protokollen wie Ethernet, ATM usw. eingesetzt werden. Da die HAW voraussichtlich ein mobiles Labor aufbaut, könnte die Zusammenarbeit der Middleware-Technologien mit den drahtlosen Kommunikationsprotokollen besonders interessant sein. Diese Middleware-Technologien können mit jedem drahtlosen Protokoll (siehe hierzu Kapitel 2 Abschnitt 2.2.3 Zugriffsverfahren und Kommunikationsprotokolle), das ein IP Stack zur Verfügung stellt, eingesetzt werden.

- **Jini:** Jini kann mit den drahtlosen Technologien z. B. IrDA oder WLAN, die den Netzwerk-Layer bereitstellen, zusammenarbeiten. Durch die relativ hohen Hardwareanforderungen kann Jini nur auf entsprechend leistungsfähigen Systemen eingesetzt werden. Da Jini unabhängig von der Netzwerktechnik ist [Sun 2003], kann das Konzept grundsätzlich auch in Bluetooth-basierten Netzen eingesetzt werden. Durch die Bluetooth-Einführung wird z. B. die Reichweite in einem Jini-System eingeschränkt.
  Jini ist völlig plattformunabhängig, vorausgesetzt es existiert eine entsprechende JVM (In diesem Zusammenhang soll auf Kapitel 4, Jini-Abschnitt, verwiesen werden). Jini setzt wie erwähnt eine funktionierende Netzwerk-Ebene voraus, z. B. einen initialisierten IP-Protokollstack und kümmert sich nicht selbst um die Netzwerk-Ebene. WLAN stellt eine solche Umgebung auf Basis von TCP/IP zur Verfügung. Laut Sun Microsystems sind die beiden Technologien Bluetooth und WLAN kompatibel [Sun 2003].

- **UPnP:** Auch UPnP setzt eine funktionierende Netzwerk-Ebene voraus, fordert dafür allerdings wesentlich geringere Hardwareanforderungen. UPnP setzt einen initialisierten TCP/IP-Stack voraus [UPnP 2003] (siehe Kapitel 4 UPnP-Abschnitt). Da Bluetooth und WLAN mit TCP/IP umgehen können, soll UPnP auch mit diesen beiden Technologien, genauso wie mit IrDA funktionieren.

- **ViaXML:** Die Kommunikation bei ViaXML findet durch XML-RPC oder SOAP mit HTTP/HTTPs Unterstützung statt. Man kann auch in diesem Fall Bluetooth, IrDA oder WLAN als Kommunikationsprotokolle verwenden, wenn dabei ein IP Stack vorhanden ist. ViaXML kann bei der nächsten Generation Microsoft Tools (Visual Studio .NET 2003) benutzt werden. Dabei können die Applikationen in VB.NET, C#, oder C++ geschrieben werden. Zusätzlich kann ViaXML auch vom Pocket Internet Explorer und mit allen aktuellen Microsoft development Tools benutzt werden [Odysseysoftware 2003].

- **CORBA/JacORB:** CORBA definiert ein Framework zur Implementierung von Interoperabilitätsprotokollen unter dem Namen GIOP (General Inter ORB Protocol). Eine bestimmte Ausprägung von GIOP ist IIOP (Internet Inter ORB Protocol), welches auf TCP/IP aufsetzt [OMG 2003]. Man kann ein Bluetooth Inter ORB Protocol oder WLAN Inter ORB Protocol entwerfen, die an Stelle von TCP/IP die entsprechenden Bluetooth-Protokolle bzw. WLAN-Protokolle verwenden, und auf Basis der Open-Source CORBA-Plattform JacORB implementieren. JacORB 1.4.1 (und ältere Versionen) verwendet(en) reguläre TCP/IP-Kommunikation mit Standard-Java-Sockets. D. h., überall wo diese Voraussetzungen gegeben sind, kann JacORB eingesetzt werden. Ab der kommenden Version 2.0 soll laut Nicolas Noffke (Autor des JacORB-Projekts) eine ETF (Extensible Transport Framework) Implementierung verfügbar sein, mit der beliebige eigene Transports über diese standardisierte Schnittstelle in JacORB "eingestöpselt" werden können.

5.3 Direkter Vergleich anhand ausgewählter Kriterien

Die vorgestellten Middleware-Technologien werden in diesem Abschnitt detailliert anhand deren wichtigsten und spezifischen Merkmalen diskutiert. Hierzu zählen die Diensterbringung, die Abstraktionsmechanismen, die Technologiereife bzw. der Middleware-Markt, die Interoperabilität mit anderen Technologien, sowie die Sicherheitsfunktionen und die Plattformunabhängigkeit. Um Bedeutungsmissverständnisse dieser Merkmale zu vermeiden, werden sie im Folgenden erst kurz vorgestellt:

- Diensterbringung: Die Diensterbringung beschreibt die wichtigsten Funktionen, die von der jeweiligen Middleware zur Verfügung gestellt werden.

- Abstraktionsmechanismen: In diesem Punkt werden die Mechanismen vorgestellt, die zur Abstraktion von Technologien bzw. der Kommunikationsmechanismen implementiert sind.

- Technologiereife / Middleware-Markt: Unter diesem Punkt wird diskutiert, ob eine Technologie als Konzept verfügbar, verbreitet, ausgereift oder bereits am Markt zu kaufen ist.

- Interoperabilität mit anderen Technologien: Hierbei wird gezeigt, welche konkreten Technologien sich durch eine Middleware integrieren lassen, so dass eine Interoperation möglich wird.

- Sicherheitsfunktionen: In diesem Teil wird erörtert, welche middlewarespezifischen Sicherheitsfunktionen von der jeweils betrachteten Technologie realisiert sind bzw. werden.

- Plattformunabhängigkeit: Hier wird berichtet, welche Middleware-Technologien von der Plattform und der Programmiersprache unabhängig sind und wodurch sie diese Unabhängigkeit erreichen.

Diese Merkmale werden im Zusammenhang mit der Middleware in den nachfolgenden Abschnitten in Anlehnung an [BSI 2003] und an vom Hersteller zur Verfügung gestellten Informationen [Jini 2003], [UPnP 2003], [Odysseysoftware 2003], [JacORB 2003] näher betrachtet und u. a. in der *Tabelle 5.1* zusammenfassend dargestellt.

5.3.1 Diensterbringung

- Jini bietet eine Reihe von Features zur Verwaltung und Nutzung der Dienste und Ressourcen an. Es handelt sich dabei vorwiegend um Infrastrukturdienste. Diese umfassen Vermittlungsdienste (LUS), Java RMI Dienste [73], Sicherheitsdienste, Leasing-Dienste, Transaktionsdienste sowie Dienste zur Verwaltung von Ereignissen.

- Die Dienste, die von UPnP erbracht werden, sollen von der Adressierung von Diensten über die Erkennung, die Beschreibung und das Management von Diensten bis hin zur Ereignissteuerung, sowie der Darstellung von Diensten reichen. Die Adressierung in UPnP erfolgt auf Basis von TCP/IP und darauf aufbauend auf DHCP oder AutoIP. Es ist ebenso möglich, Geräte über Namen anstelle von numerischen Adressen anzusprechen (DNS). Die Erkennung von Geräten erfolgt über das Protokoll SSDP. Dabei kann ein neu integriertes Gerät alle von ihm angebotenen Dienste über die Control Points veröffentlichen. Ebenso können über die Control Points andere Geräte und deren Dienste gesucht werden. Die Nachrichten, die zusammen mit Ereignissen verschickt werden, werden in XML formatiert und beschrieben und mittels GENA zwischen HTTP-Ressourcen versandt.

- ViaXML ist ein Konzept, das auf XML-RPC aufbaut und zur Entwicklung von Anwendungen zwischen mobilen Endgeräten und Rechner geeignet ist. Die Dienstleistung findet über das Internet statt. Dieses verwendet XML-RPC mit HTTP/HTTPs Unterstützung für den Datentransport.

[73] Die Dienste der Java RMI umfassen die Suche, die Aktivierung von Objektgruppen, sowie die Speicherverwaltung und Speicherbereinigung [Jini 2003].

- Den Kern von CORBA/JacORB bildet die Vermittlung von Objekten durch den ORB, der als Telefonzentrale in einem verteilten System fungiert. Er stellt Dienste für die Initialisierung und Auffindung von Zielobjekten und für die Übermittlung von Methodenaufrufen (requests) und deren Resultaten zwischen Clients und Zielobjekten zur Verfügung. Als Kommunikationsprotokoll wird, wie inzwischen allgemein üblich, das IIOP verwendet. Beim Start eines Servers gibt dieser von sich eine Inter Object Reference (IOR) aus, die, bei dem als CORBA Object Service implementierten Namensdienst, registriert wird und von aufrufenden Clients abgerufen werden kann. Mit Hilfe dieser IOR kann sich der Client direkt mit dem Server verbinden. Als Entwicklungskomponente dient ein IDL Compiler, der wie üblich Java Stubs und Skeletons erzeugt.

5.3.2 Abstraktionsmechanismen

- Dadurch, dass der Jini-Einsatz eine lauffähige JVM, sowie eine Unterstützung der Kommunikation über Java RMI erfordert, ist Jini plattformunabhängig. Da Jini auf die Programmiersprache Java zugeschnitten ist, kann die Architektur nicht für beliebige Programmiersprachen verwendet werden. Darüber hinaus ist Jini unabhängig von der zugrunde liegenden Netztechnologie. Jini bietet Unterstützung für die Geräte oder Dienste, für die keine JVM zur Verfügung steht und die nicht über Jini-Technologie mit einem Jini-System verbunden sind.

- UPnP ist dadurch charakterisiert, dass es auf bestehende Technologien und Standards aufsetzt und eine klare Schichtenarchitektur aufbaut. Dadurch wird eine normierte Abstraktion von der Geräteschicht bis auf die Dienstschicht möglich. So erfolgt die Abstraktion der Kommunikation über IP, TCP, UDP sowie HTTP. Service Discovery erfolgt über SSDP, Ereignisse werden über GENA verwaltet und die Steuerung der Dienste wird über SOAP implementiert. Zudem werden zur Dienst- und Namensverwaltung LDAP[74] und DNS eingesetzt.

- Da ViaXML auf den freien XML-Standards basiert und Standard-HTTP/HTTPs Internet-Protokolle verwendet, ist sie plattformunabhängig. ViaXML abstrahiert von den Kommunikationstechnologien, die durch die Verwendung standardisierter Protokolle wie SOAP oder XML-RPC und HTTP integriert werden, sowie die konsequente Nutzung von XML, wodurch eine standardisierte Beschreibungsmöglichkeit unterschiedlicher Dokumente und Dokumentformate ermöglicht wird.

- Ein bedeutender Aspekt bei CORBA ist die Programmiersprachenunabhängigkeit der Objektimplementierungen. Diese Unabhängigkeit von speziellen Programmier-

[74] *Lightweight Directory Access Protocol.*

sprachen wird über die IDL[75] erreicht. Es existieren Schnittstellen zu nahezu allen verfügbaren, objektorientierten Programmiersprachen. Die in IDL-Spezifikationen definierten Datentypen und Sprachkonstrukte werden über GIOP in eine Menge von Sprachprimitiven übersetzt, die zwischen ORBs transportiert werden können. Diese Sprachprimitive werden ihrerseits mittels IIOP wiederum in Nachrichten transformiert, die dann über TCP/IP übertragen werden können. Neben der Abstraktion der Programmiersprache durch die IDL, die bei JacORB in Java implementiert ist, stehen zwei weitere Abstraktionsmechanismen zur Verfügung: Das Protokoll IIOP, das von der Übertragungstechnologie abstrahiert und einen Kommunikationsmechanismus definiert, sowie das Protokoll GIOP, das vom Kommunikationsmechanismus abstrahiert und die Spezifikation von komplexen Sprachkonstrukten ermöglicht. Da sämtliche Komponenten in Java geschrieben wurden, kann JacORB auf allen Plattformen eingesetzt werden, für die es eine Java-Unterstützung gibt.

5.3.3 Technologiereife / Middleware-Markt

- Was die erforderliche Jini-Technologie angeht, so erfordert ein sinnvoller Einsatz von Jini bisher noch eine Reihe von Ressourcen wie Prozessorleistung, Speicher oder Bandbreite der Netze. Auch die Skalierbarkeit ist nicht gegeben und kann bereits bei Verwendung von eher lokal ausgerichteten Netztechnologien wie Bluetooth zu einem Flaschenhals werden. Probleme technischer, vertrieblicher und lizenzrechtlicher Art haben bislang die Entstehung einer engagierten Entwicklergemeinde verhindert [Muller 2001]. Aus den genannten Gründen kann diese Technologie lediglich ‚als in Teilen reif' eingestuft werden. Sie ist trotzdem schon als Konzept auf der Sun-Homepage verfügbar.

- UPnP wird heute von einer Vielzahl von Unternehmen implementiert. Hierzu zählen Microsoft, Linksys, DLink, Intel, Netgear sowie Buffalo Technology und Arescom. Zudem ist UPnP in zahlreichen Produkten bereits verbreitet wie beispielsweise Windows XP. UPnP ist ebenfalls in Geräten unterschiedlicher Industrien implementiert, angefangen vom PC bis hin zu Geräten im Bereich Consumer Electronics oder Gebäudeautomation. In Folge der weiten Verbreitung dieser Middleware-Technologie und der Verfügbarkeit dieses Standards (Windows ME und XP verfügen schon über UPnP-Technologie) kann UPnP als ausgereift angesehen werden.

- ViaXML ist eine noch sehr junge Technologie, die insgesamt noch nicht als ausgereift bzw. vermarktet betrachtet werden kann. Dies relativiert sich jedoch, da auf zahlreiche

[75] Die OMG stellt eine Schnittstellenbeschreibungssprache für die Programmiersprachenunabhängigkeit bereit: Die IDL (*Interface Definition Language*). Mit ihr werden die verteilten Objekte an ihren äußerlich sichtbaren Eigenschaften in einer standardisierten und implementierungsunabhängigen Weise beschrieben.

inzwischen ausgereifte Technologien wie SOAP oder XML-RPC und HTTP zurückgegriffen wird. Es gibt inzwischen viele Organisationen, die ViaXML in ihren Betrieb eingesetzt haben, wie PepsiAmericas, Microsoft, Daimler-Chrysler, die US Army, die US AirForce, Canon etc. [Odysseysoftware 2003]. Die ViaXML-Technologie ist bereits auf der Odyssey Software Homepage verfügbar.

- In der CORBA-Welt gibt es eine Vielzahl an Implementierungen von ORBs. Hierzu zählen beispielsweise JacORB, MICO, SmalltalkBroker, OpenORB, Omniorb, VisiBroker etc.. CORBA ist eine, für interoperable Anwendungen, geeignete und verfügbare Technologie. Die Basisdienste (CORBA-Services) sind bereits implementiert. Höherwertige Dienste sind jedoch kaum standardisiert, bedingt durch die Zunahme an individuellen Anforderungen. Dennoch bietet CORBA eine sehr gute Infrastruktur, um über standardisierte Schnittstellen Anwendungen zu entwickeln und dabei auf sehr viel Grundfunktionalität in Form der Basisdienste zurückgreifen zu können. JacORB ist derzeit noch ein eingeschränkter Java ORB, der sich im Entwicklungsstadium befindet. Aus diesem Grund fehlen einige CORBA Komponenten (z. B. Security Service, Notification Service), die aber mit fortlaufender Entwicklung integriert werden sollen. Im Gegensatz zu den kommerziellen Konkurrenzprodukten sind alle Bestandteile in Java geschrieben, wodurch sich JacORB sehr flexibel einsetzen lässt. Zudem ist das Produkt kostenlos, auch der Source Code ist frei verfügbar. Ob das Produkt gegen die kommerzielle Konkurrenz und andere Technologien bestehen kann, bleibt abzuwarten; Die Ansätze sehen allerdings recht viel versprechend aus.

5.3.4 Interoperabilität mit anderen Technologien

- Jini soll in weiten Bereichen die Interoperabilität zwischen Endgeräten ermöglichen, auf Netzebene beispielsweise die Integration unterschiedlicher Betriebsysteme. Auf Protokollebene können ebenfalls Verbindungen geschaffen werden. Bei Jini bilden die kooperierende Geräte so genannte ‚*Communities*'. Diese sind in aller Regel durch die Netzwerktopologie begrenzt. Communities können zu einer beliebig großen ‚*federation*' zusammengeschlossen werden [Sun 2003]. Somit ist es zwar theoretisch möglich, Geräte mittels des Internets über große Distanzen zu teilen, aber dieses widerspricht den eigentlichen Intentionen von Jini.

- UPnP bietet selbst nicht die Möglichkeit zur Interoperabilität an. UPnP unterstützt sowohl Netztechnologien, als auch Computer-Ressourcen. Es werden alle drahtgebundenen und drahtlosen Netztechnologien unterstützt, auf die IP aufgesetzt werden kann. Zusätzlich werden beliebige Geräte wie PCs, Audio-/Videogeräte oder Drucker unterstützt, die netzfähig und analog der Gerätespezifikation über XML beschrieben sind.

- Die XML-RPC und SOAP Integration in ViaXML macht sie interoperabel mit jeder Technologie, die auch XML-RPC oder SOAP unterstützt. Diese Technologie kann in Java, Perl, PHP, Python, C++, ASP, ASP.Net geschrieben sein. ViaXML lässt dem Entwickler die freie Entscheidung für einen Web Server, der die unbegrenzte Stabilität, Fehlertoleranz und Erreichbarkeit bietet [Odysseysoftware 2003].

- CORBA bietet eine Reihe von Schnittstellen zu anderen Technologien. Auf Kommunikationsebene integriert CORBA Protokolle wie TCP/IP. Um ein Objekt zu implementieren, können verschiedene Programmiersprachen wie C++, Java sowie Smalltalk benutzt werden. Für diese Sprachen sind Language Mappings definiert, die angeben, wie die in IDL angegebenen Interfaces und Datentypen in Konstrukte der Programmiersprache umgesetzt werden. JacORB interoperiert mit jeder ORB-Implementierung über IIOP. In der Praxis wurde schon JacORB mit folgenden ORBs erfolgreich eingesetzt: MICO, TAO, Orbacus, OrbixWeb, VisiBroker, ORBit, omniORB, Vitria C++ und Java ORBs [JacORB 2003].

5.3.5 Sicherheitsfunktionen

- Der Grundgedanke von Jini ist, dass Dienste ein Netz betreten und verlassen können. Somit ist die Gefahr der Integration mit virenverseuchten Diensten in einem Jini-Netzwerk jederzeit gegeben. Um das zu vermeiden, verwendet Jini das gleiche Sicherheitskonzept wie alle Java-Anwendungen (JDK 2). Dieses Sicherheitsmodell kommt jedoch nur zum Einsatz, wenn in jeder Client- und jeder Server-Applikation ein Java SecurityManager installiert ist. Dieser überprüft, bevor sicherheitsrelevante Aktivitäten (z. B. Lesen, Ausführen von Dateien, Netzverbindungen zwischen den Diensten) gestartet werden, ob diese genehmigt wurden. Diese Informationen werden in Policy Dateien gespeichert; Hier werden alle Zugriffsrechte für alle User aufgelistet.

- Aufgrund des ursprünglichen Design-Ziels, eine nahtlose und einfache dynamische Integration und Vernetzung von Geräten und Diensten zu realisieren, bietet UPnP kein explizites Sicherheitskonzept an. UPnP sitzt das Vorhandensein von Sicherheitsmaßnahmen wie Subnetze (Gateways oder Firewalls), MAC-Schicht-Verschlüsselung (durch Netzwerkadapter), sowie physikalische Zugriffskontrolle voraus.

- Um einen sicheren Kanal zwischen einem Client und einem Server herzustellen, bietet ViaXML die Möglichkeit, über SSL empfindliche Benutzerdaten zu schützen. ViaXML verwendet die Authentisierung, die Autorisierung und die Verschlüsselungsservices von dem HTTP-Server (IIS oder Apache).

- Teil der CORBA-Dienste ist der Security Service. Dieser definiert Schnittstellen, um eine Vielzahl der in einem verteilten System relevanten Sicherheitsanforderungen zu

realisieren und zu verwalten. Mit dem Sicherheitsdienst lassen sich Sicherheitsfunktionen in den Bereichen Vertraulichkeit, Integrität, Verantwortlichkeit, sowie Verfügbarkeit implementieren [OMG Sec 2003]. CORBA realisiert die Zugriffskontrolle auf zwei Arten: Zum Einen können die Anwendungsobjekte selbst die Kontrolle vornehmen, zum Anderen kann die Kontrolle dem ORB überlassen werden. Der Security Service fehlt bei der JacORB-Implementierung, soll aber mit fortlaufender Entwicklung integriert werden. Dieser Dienst ist auch unter dem Namen CORBASEC bekannt.

5.3.6 Plattformunabhängigkeit

- Jini ist plattformunabhängig. Dies erreicht es dadurch, dass es auf Java basiert. Java bringt die Eigenschaft des ‚plattformunabhängigen' von sich aus mit. Jini arbeitet also auf einem Level weit über dem Betriebssystem und den Verbindungsprotokollen.

- UPnP ist auch plattformunabhängig. Diese gewünschte Flexibilität erreicht es dadurch, dass es auf dem etablierten, weit verbreiteten TCP/IP Standard, welcher heute auf allen Systemen zur Verfügung steht, aufbaut [Microsoft-UPnP 2003].

- ViaXML auf der Basis von SOAP und XML-RPC macht keinerlei Annahmen über das Programmiermodell und ist durch die weite Verbreitung von XML sehr leicht in allen relevanten Sprachen zu realisieren. Von daher ist sie auch plattformunabhängig.

- CORBA setzt auf das Konzept der objektorientierten Programmierung auf, ist aber nur mit Sprachen einsetzbar, für die eine Sprachbindung definiert wurde und eine ORB-Implementierung existiert. JacORB ist plattformunabhängig. Dies erreicht es dadurch, dass es wie Jini auf Java basiert und von ihr diese Plattformunabhängigkeit erbt.

5.4 Middleware im Vergleich: Zusammenfassung

Es sollen nun an dieser Stelle die wichtigsten Unterschiede der eingangs benannten Middleware-Technologien in Zusammenhang mit den im vorangegangenen Abschnitt (5.3 Direkter Vergleich anhand ausgewählter Kriterien) beschriebenen Technologiemerkmalen zusammenfassend tabellenartig dargestellt werden. Die nachfolgende *Tabelle 5.1* listet diese auf.

Middleware / Merkmale	Jini	UPnP	ViaXML	JacORB
Hersteller	Sun	Microsoft	Odyssey Software	FU Berlin / Xtradyne Technologies AG
Lizenz	Open-Source-Lizenzen (SCSL[76]).	Open-Source (für Mitglieder)	Kostenpflichtig	Open-Source
Abstraktion / Netzwerktransport	Unabhängig	TCP/IP (DHCP, DNS oder Auto IP)	XML/RPC SOAP	IIOP
Programmiersprache	Java	Unabhängig	Unabhängig	Java
OS/Plattform	Unabhängig	Unabhängig	Unabhängig	Unabhängig
Code Mobility	Ja über Java RMI	Nein	Nein	Nein
Ressourcenverbrauch	JVM, Proxy, hohe Anforderung an Speicher und CPU	Keine JVM Kein Proxy	Keine JVM Kein Proxy	JVM Kein Proxy
Interoperabilität	Ja	Nein	Ja	Ja
Der zentrale Punkt	LUS	Control Point	Internet	ORB
Diensterbringung	Lookup, Discovery, Join	Addressing, Discovery, Description, Control, Eventing	XML-RPC / SOAP, HTTP/HTTPs	Initialisierung, Auffindung, Übermittlung durch ORB.
Sicherheit	SecurityManager (principal und access control list)	Kein explizites Sicherheits-konzept	SSL	CORBASec / Security Service ist in Planung
Anwendungsgebiete	Ubiquitous Computing, spontane Vernetzungen	Heimnetzwerk, Kleine Netzwerke	C/S-Anwendungen über Internet	Objekt-orientierte Konzepte
Technologiereife	Teilweise reif	Ausgereift (beinhalten mehrere BS)	Nicht sehr verbreitet	Nicht ganz ausgereift

Tabelle 5.1: Middleware im Vergleich

[76] *Sun Community Source License*

Kapitel 5: Middleware im Vergleich

Wie aus der *Tabelle 5.1* ersichtlich, bestehen die wichtigsten Unterschiede zwischen den betrachteten Middleware-Technologien (Jini, UPnP, ViaXML, JacORB) in den folgenden Eigenschaften:

- Den Netzwerktransportart lässt Jini offen, wobei UPnP den TCP/IP, ViaXML den XML-RPC / SOAP und JacORB den IIOP voraussetzen.

- Jini und JacORB sind im Gegensatz zu ViaXML und UPnP Java-abhängig.

- Für den Einsatz in der Praxis spielen auch die erforderlichen Ressourcen eine wichtige Rolle. Betrachtet werden die Prozessorleistung, die Speichergröße und die Bandbreite des Netzwerkes. Da Jini auf einem System mit einer Java VM laufen muss, hat die zur Verfügung stehende Hardwareleistung besondere Bedeutung. Deswegen liegt Jini beim Ressourcenverbrauch (JDK, Proxy-Ausführung und durch hohe Anforderung an Speicher und CPU) ganz schlecht im Vergleich zu den restlichen Technologien, bei denen kein mobiler Code gebraucht wird.

- Alle Technologien außer UPnP sind interoperabel mit anderen Technologien.

- Der zentrale Punkt ist bei Jini der LUS, bei UPnP der Control Point, bei ViaXML das Internet und bei JacORB der ORB.

- Bei der Jini-Technologie sind die Protokolle (Lookup, Discovery, Join) für die Diensterbringung zuständig. Bei der UPnP-Technologie kommen die Protokolle (Addressing, Discovery, Description, Control, Eventing) zum Einsatz. Die ViaXML-Technologie verwendet XML-RPC / SOAP und HTTP/HTTPs. Bei der JacORB-Technologie ist der ORB für die Initialisierung, die Auffindung und die Übermittlung zuständig.

- Jini verwendet die Sicherheitskonzepte des Auftraggebers und der Zugangskontrollliste (SecurityManager). ViaXML basiert hingegen auf SSL. Bei JacORB ist Security Service in Planung und UPnP hat kein explizites Sicherheitskonzept.

- Jini ist für Ubiquitous Computing[77] und spontane Vernetzungen gut geeignet. UPnP ist für das Heimnetzwerk und kleine Netzwerke von Nutzen. ViaXML ist für C/S-Anwendungen über Internet von Interesse. JacORB hingegen lässt sich für objektorientierte Konzepte gut einsetzen.

- Die Jini-Technologie ist lediglich in Teilen als reif eingestuft. Ob sie sich in Zukunft bewähren wird, bleibt dahingestellt. UPnP ist schon in vielen Windows BS mitintegriert und ist ausgereift. Da Microsoft bemüht ist, Fehler zu korrigieren, sind

[77] Für mehr Information zum Thema Ubiquitous Computing ist [Sauerburger 2003] lesenswert.

die mangelnden Sicherheiten bzw. die Lücken, die durch die UPnP-Integration zu Stande kamen, in Windows XP beseitigt worden [Microsoft-UPnP 2003]. ViaXML ist nicht sehr verbreitet, wird aber schon von einigen Firmen wie PepsiAmericas, Microsoft, Daimler-Chrysler verwendet. JacORB ist nicht ganz ausgereift, da einige CORBA Komponenten (z. B. Security Service) fehlen, die aber mit fortlaufender Entwicklung integriert werden sollen.