Mobile Klienten: Ortsübergreifender Zugang zu Diensten in offenen verteilten Informationssystemen

Michael Merz, Kay Müller-Jones, Winfried Lamersdorf

Universität Hamburg, Fachbereich Informatik,
Datenbanken und Informationssysteme
Vogt-Kölln-Str. 30, D-22527 Hamburg
eMail: [merz|lamersd|kmueller] @ informatik.uni-hamburg.de

Zusammenfassung

Die zunehmende Verbreitung und Leistungsfähigkeit globaler Kommunikationsinfrastrukturen führt zu einer entsprechend unübersichtlichen Anzahl von Diensten in offenen verteilten Informationssystemen. Der Mechanismus "Markt" erscheint hierbei zunehmend bedeutend, wenn die Zielsetzung in der effizienten Koordinierung von Nachfragern und Anbietern dieser Dienste liegt. Ziel der Projekte COSM und TRADE ist daher insbesondere die Schaffung einer verteilten Kommunikationsinfrastruktur, welche die flexible Koordination bei der Dienstnutzung unterstützt.

Die Verwendung aktueller Technologiestandards, insbesondere des *Dynamic Invocation Interface* (DII) aus der *Common Object Request Broker Architecture* (CORBA) sowie darauf aufbauende Instrumente zur Dienstbeschreibung stehen bei der Realisierung der erforderlichen Systeminfrastruktur im Vordergrund.

1. Motivation[1]

Untereinander verbundene Rechnernetze realisieren inzwischen als weit verbreitete und leistungsfähige Kommunikationsmedien Zugangsmöglichkeiten zu einer Vielzahl von Informationsdiensten an fast beliebigen Orten in der Welt. Dadurch ist bereits heute die technische Grundlage für weiträumig verteilte elektronische Märkte realisiert, die auch weiterhin - z. B. bzgl. Knoten- und Anwenderzahl, aber vor allem hinsichtlich der Vielfalt der nutzbaren Informationsdienste - ein anhaltendes Wachstum erwarten lassen. Somit entstehen aber auch Orientierungsprobleme für Nutzer von Diensten in dermaßen vielfältigen und offenen Dienstemärkten. Um diesen Problemen zu begegnen, ist zunächst - soweit möglich - eine weitgehende Vereinheitlichung (Standardisierung) von Diensten und deren Schnittstellenbeschreibung erforderlich sowie zudem auch adäquat mächtige systemtechnische Unterstützungsmechanismen, die den Zugang zu Diensten in diesen elektronischen Märkten erleichtern könnten. Ob aber dem Nutzer derartiger Dienste - durch Vereinheitlichung des Zugangs zu Netzdiensten - bereits ausreichende *Systemunterstützung* zur Verfügung steht, ist jedoch stark zu bezweifeln. Systemtechnisch sollte zudem sowohl die Interaktion mit entfernten Diensten im Verlaufe einer Sitzung von einem, aber auch von wechselnden Standorten aus, unterstützt werden. Dadurch können dann auch *ortsmobile* Benutzer oder Benutzergruppen durch entsprechend wechselnde Zugangsmöglichkeiten zu den von ihnen genutzten Diensten stets ”begleitet” werden und so Zugang zu Netzdiensten *von überallher*, auf *einheitliche* Weise und mit *geringstmöglichem Aufwand* erhalten.

---

[1] Appeared in: GI/SI Jahrestagung, Zürich, Sept. 1995

Das in diesem Beitrag vorgestellte Konzept der *mobilen Klienten* und seine systemtechnische Realisierung unterstützt eine ortsübergreifende Nutzung von Netzdiensten, d.h. für solche Klienten, die von *wechselnden* Orten und Nutzern aus mit einem Server kommunizieren.

1.1 Dienste in elektronischen Märkten

Der Zugang zu Diensten in elektronischen Dienstemärkten kann u. a. danach unterschieden werden, auf welche ”Art” von Diensten zugegriffen werden soll: z. B. können solche Dienste *organisationsintern* oder *kommerziell* genutzt werden, es können prinzipiell entweder *standardisierte Dienste* verwendet werden oder jeder Diensterbringer kann eine unterschiedliche *individuelle Semantik* realisieren. Ist in derartigen Szenarien ein *Diensttyp* (d.h. eine festgelegte Schnittstelle und Semantik eines Dienstes) vorab bekannt, kann anhand von *Diensteigenschaften* (wie z. B. Auflösung eines Druckers oder seine aktuelle Auslastung) einem Nachfrager genau der Dienst vermittelt werden, welcher bzgl. der nachgefragten Eigenschaften als ”am besten geeignet” erscheint [ISO-ODP].

Wichtigste Voraussetzung für eine solche, *system*unterstützte Dienstvermittlung ist also eine einheitliche Beschreibung derartiger *klassifizierter* Dienste [MML95] und ihrer jeweiliger Diensteigenschaften. Auf der Basis dieser Dienstbeschreibungen kann dann ein im Rahmen der Standardisierungsbemühung der ISO im Bereich ODP (Open Distributed Processing) entworfener *Trader* in vielen Fällen auch eine weitgehend *automatisierte* Vermittlung zwischen Klientennachfrage und Server-Angebot durchführen. Aufgrund der dafür notwendigen vorherigen Vereinheitlichung von Diensttypen bietet sich der Trader jedoch zunächst nur für den Einsatz innerhalb *geschlossener* Organisationen an, da dort - im umgekehrten Verhältnis zur Organisationsgröße - der für Diensttypen erforderliche Normierungsprozeß am effizientesten durchgeführt werden kann.

Bei individuell entworfenen, z.B. kommerziell angebotenen Diensten kann deren Diensttyp nur schwer in einheitlicher und effizienter Weise standardisiert werden, da bei einer einheitlichen Dienstsemantik auch alle individuellen Ausprägungen, ”Gepflogenheiten” und Vorteile einzelner Diensterbringer berücksichtigt werden müssen. Eine systemtechnische Unterstützung erscheint (hier im Gegensatz zu organisationsintern genutzen Diensten) als erheblich komplexere und nur teilweise automatisch zu lösende Aufgabe. Zudem zeigt z.B. die Entwicklung des Internet-Dienstes *World Wide Web* (WWW), daß auch ohne standardisierte Dienstsemantik individuelle Server mit begrenztem Aufwand für Nachfrager erfolgreich sichtbar und nutzbar gemacht werden können. In dieser Kategorie der (kommerziell angebotenen) *unklassifizierten*, meist nur interaktiv vom Endanwender genutzten Dienste mit individueller Semantik *(one-of-a-kind services)*, besteht sogar die Forderung nach Freiraum zur individuellen Ausgestaltung von Schnittstelle, Semantik und Diensteigenschaften. Eine Normierung besteht hier nur in Form einer Festlegung von Schnittstellen- oder Dokumentenbeschreibungen, nicht jedoch der durch Dienste erbrachten Inhalte. Aufgrund einer derartigen Beschränkung der Normierung auf die *Repräsentationsebene* (vgl. z. B. die Sprache HTML und die damit korrespondierende, einheitliche Präsentation beim WWW) kann dann eine einheitliche Interpretation beliebiger Dienstschnittstellen durch geeignete *generische* Software-Komponenten gewährleistet und systemtechnisch unterstützt werden. In ähnlicher Weise beschränken sich z.B. bei der *Telescript*-Technik [Whit94] verteilte Systemkomponenten auf die *Interpretation* einer Skriptsprache. Syntax und Befehlsumfang dieser Sprache sind allen Interpretern bekannt und somit normiert.

1.2. Klassifizierte und unklassifizierte Dienste

Folgende Beobachtungen lassen sich zunächst direkt aus der dargestellten Einteilung von Diensten in ”klassifizierte” und ”unklassifizierte” ableiten:

1. Klassifizierte Dienste sind a-priori in ihrer Entwurfsautonomie eingeschränkt, dafür jedoch effizienter für *diensttypspezifische* Klienten nutzbar. Bei unklassifizierten Diensten entscheidet jeder individuelle Diensterbringer selbst über den Grad seiner Autonomie (vgl. z. B. Abschnitt 4.4).
2. Im Falle der klassifizierten Dienste erstreckt sich die Standardisierung auch auf die Dienstsemantik, während sie sich bei unklassifizierten Diensten generell nur auf die Repräsentationsebene beschränkt.
3. Klassifizierte Dienste sind für *diensttypspezifische* Software-Komponenten nutzbar, welche sie in der Rolle des Klienten nutzen, während unklassifizierte Dienste mit individueller Semantik nur für den menschlichen Benutzer bzw. *diensterbringerspezifische* Software-Komponenten zugreifbar sind.
4. Klassifizierte Dienste eignen sich zunächst für *organisationsinterne* Kontexte mit der Möglichkeit der *organisationsübergreifenden* Standardisierung, während unklassifizierte Dienste aufgrund des geringen Konfigurationsaufwandes eher im kommerziellen Umfeld zu finden sind.

Bei der Nutzung einer Vielzahl von unterschiedlichen Diensten in weltweiten elektronischen Märkten sind auf der Seite der Klienten in zunehmendem Maße *Systemkomponenten* erforderlich, die dem Benutzer helfen, z.B. konform zum spezifischen Protokoll und Zustand eines entfernten Servers dessen Operationen korrekt aufzurufen. Grundlage für die Realisierung derartiger Software-Komponenten ist eine gemeinsame *Dienstrepräsentation,* wie sie hier im 3. Kapitel skizziert wird. Darauf aufbauend kann ein sogenannter *generischer Klient* [MML94b] realisiert werden, der als generisches systemtechnisches Werkzeug den Zugang auch zu unklassifizierten Diensten in Netzen unterstützt. Schließlich bilden Dienstrepräsentation und generischer Klient die Basis für die in Kapitel 4 beschriebene Systemunterstützung der dynamischen Dienstnutzung durch *mobile Klienten,* die auch *ortsübergreifende* Zugangsmöglichkeiten zu Diensten realisiert.

2. Technische Voraussetzungen

Eine geeignete Infrastruktur, welche die Vermittlung und den Zugriff auf entfernte, unklassifizierte Dienste unterstützt, sollte in hohem Maße dem dynamischen Charakter von Klient/Server-Bindungen auf dem Dienstemarkt gerecht werden. Da aus diesem Grunde zur Laufzeit Bindungen zu unklassifizierten Diensten eingegangen werden müssen, deren Diensttyp a-priori unbekannt ist, sind statisch typisierte Mechanismen wie die Generierung von *Stubs* beim ONC RPC von Sun oder beim *Distributed Computing Environment* der OSF (DCE) [Schi93] für eine unmittelbare Nutzung ungeeignet. Für das Szenario des offenen verteilten Dienstemarktes soll jedoch angenommen werden, daß eine Klientenkomponente für den Zugriff auf unklassifizierte Dienste nicht neu kompiliert und gebunden werden muß. Wie auch im 4. Kapitel dargestellt, zeigt es sich, daß das *Dynamic Invocation Interface* (DII), welches ein Bestandteil der *Common Object Request Broker Architecture* (CORBA) [OMG91] ist, eine geeignete Grundlage für den Entwurf heterogener, verteilter Klient/Server-Anwendungen darstellt, wenn dynamisch typisierte Prozeduraufrufe unterstützt werden sollen. CORBA-DII bietet eine standardisierte Schnittstelle zur Definition von Repräsentationen für Operationsaufrufe und Parameterlisten sowie für das entfernte Aufrufen der beschriebenen Operation. Allerdings sichert der Einsatz des DII keine Typkonformität zwischen Klient und Server mehr zu - vielmehr muß zur Laufzeit aufgrund einer Spezifikation, welche Informationen über die Server-Schnittstelle liefert, das Erzeugen von Aufrufrepräsentationen so automatisiert werden, daß Typfehler vermieden werden. Träger dieser Information ist die nachfolgend eingeführte Dienstrepräsentation.

3. Eigenschaft und Verwendung der Dienstrepräsentation

Funktion der *Dienstrepräsentation* (DR) ist eine im wesentlichen semiformale Spezifikation von Schnittstelle und Semantik beliebiger entfernter Dienste. Als Komponenten gehören zur DR neben der Spezifikation von Syntax und (so weit wie möglich) Semantik der aufrufbaren *Operationen* eines Dienstes sowie seines spezifischen Aufruf*protokolls* auch *lokale Zustandsvariable*, welche interaktiv durch den Benutzer oder als Ergebnis entfernter Prozeduraufrufe gelesen und modifiziert werden können. Sie repräsentieren u. a. spezifische Eigenschaften einer speziellen Klient-Server-Sitzung (bzw. ihres aktuellen Zustandes), die im Netz an beliebigen Orten persistent gehalten werden kann. Auch über Rechnergrenzen hinweg kann eine DR kommuniziert und für die Nutzung von Netzdiensten von ganz unterschiedlichen Orten aus eingesetzt werden.

3.1. Strukturelle Eigenschaften der Dienstrepräsentation

Zusammengefaßt können ganz unterschiedliche Aspekte der Dienstspezifikation jeweils durch dedizierte, in die DR eingebettete Beschreibungskomponenten ausgedrückt werden: Ausgehend von der erforderlichen Spezifikation der *operationalen Schnittstelle* des Dienstes über eine dienstspezifische *Protokollspezifikation* bis hin zu *natürlichsprachlichen Erläuterungen* für den Anwender, sind prinzipiell auch noch weitere *Detailspezifikationen* des Dienstes in die DR einbettbar. Geeignete systemtechnische *Werkzeuge* helfen, jeweils spezielle Spezifikationskomponenten innerhalb der DR zu erkennen und zu interpretieren. Als Beispiel dafür sei zunächst der *generische Klient* (siehe Abschnitt 4.1) genannt, welcher über die Interpretation einer Benutzerschnittstellendefinition und das Generieren eines dienstspezifischen Formulars den Anwender bei der Interaktion mit einem beliebigen entfernten Dienst unterstützt.

Entsprechend der geschilderten und der in Abschnitt 3.2 aufgeführten Anforderung an die Zustandsrepräsentation wird jede DR als dynamische Datenstruktur realisiert, welche zur Programmlaufzeit von ganz unterschiedlichen Systemkomponenten geladen, inspiziert und modifiziert werden kann. Damit ist sie im Prinzip einer Schnittstellenspezifikationen im CORBA-Schnittstellen-Repository [OMG91] vergleichbar, die ebenso inspiziert werden kann, jedoch nicht gleichzeitig auch der *Zustands*kapselung dient, d. h. keine modifizierenden Operationen auf Schnittstellenbeschreibungen erlaubt.

Neben den exemplarisch aufgeführten *spezialisierten Anwendungen* (wie z.B. dem generischen Klienten) sind aber auch *generische Anwendungen* erforderlich, welche den Inhalt einzelner Spezifikationskomponenten in einer typgerechten Weise darstellen, jedoch nicht interpretieren. Als Beispiel dafür seien hier sogenannte Dienst-”Browser” oder Schnittstellen-”Repositories” genannt, welche - anhand von Typinformationen - in der Lage sind, Spezifikationskomponenten von Diensten zu interpretieren und somit deren Inhalt entweder anzuzeigen (*Browser*) oder persistent zu speichern (*Repository*). Zu diesem Zweck sind ebenfalls zur Laufzeit Typinformationen als Bestandteil der DR erforderlich, welche die Struktur von Spezifikationskomponenten festlegen. Diese *Typkomponenten* sind ihrerseits konstruiert aus normierten *Typkennungen*, so daß generische Anwendungen über die Interpretation von Typkomponenten den Inhalt einer kompletten DR lesen können.

Generische Anwendungen, wie z. B. Repositories, können somit aufgrund des dynamischen Schemas innerhalb der DR Spezifikationskomponenten auch dann verwalten, wenn im Laufe der Entwicklung neuartige Komponententypen eingeführt werden. Ein daraus aufgebautes, dynamisch erweiterbares Repository-Schema ist dadurch in der Lage, eine große Menge an unterschiedlichen Dienstrepräsentationen mit jeweils verschiedenartigen Komponententypen zu verwalten und Anfragen an Komponententypen und deren Extensionen zu verarbeiten.

3.2. Dynamische Aspekte der Dienstrepräsentation

Neben der Kapselung von unterschiedlichen Spezifikationskomponenten besteht die zweite wesentliche Aufgabe der DR in der Kapselung des *Kommunikationszustandes*: Anwendungsspezifische Datenstrukturen, Parameterwerte, Netzzustände bzw. Datenwerte der Spezifikationskomponenten repräsentieren dabei jeweils einen aktuellen DR-Zustand, welcher sich aus der bisherigen Ausführung von Server-Operationen innerhalb einer speziellen Klient/Server-Sitzung ergeben haben kann. Dieser sitzungsspezifische Gesamtzustand kann - je nach Umfang der Spezifikationskomponenten - von Anwendungen wie dem generischen Klienten dazu verwendet werden, z. B. zulässige Folgeoperationen zu bestimmen, entfernte Operationen durch den Benutzer ausführen zu lassen oder Datenstrukturen der DR aufgrund von Operationsresultaten zu modifizieren.

Die Fähigkeit zur Kapselung von Dienstspezifikation und Sitzungsszustand in Verbindung mit der Migrationsfähigkeit von in einer DR spezifizierten Klientenzuständen erlaubt so also beliebigen Benutzern und Benutzergruppen die *ortsübergreifende* Nutzung von Diensten (z.B. am Arbeitsplatz, von zuhause aus, auf Reisen etc.). Basis dafür ist die Flexibilität der hier zugrundegelegten DR - insbesondere ihre Erweiterbarkeit um zusätzliche Spezifikationkomponenten. Damit können Standardisierungsbemühungen in Dienstemärkten zunächst auf die Dienstbeschreibungsebene beschränkt werden. Sie ermöglicht Diensterbringern (insbesonderen den innovativen) in einem elektronischen Markt einen raschen und flexiblen Marktzugang durch die autonome Festlegung von Spezifikationskomponenten. Damit können Dienstrepräsentationen mit zunehmendem Spezifikationsumfang flexibel auch für die im folgenden Kapitel skizzierten Bereiche systemtechnischer Unterstützung komplexer verteilter Anwendungen in offenen, weltweit verteilten Rechnernetzen eingesetzt werden.

4. Exemplarische Anwendungsbereiche für Dienstrepräsentationen

Nachdem Funktion und Struktur der DR im vorangegangenen Kapitel skizziert wurden, sollen nun einige verschiedenartige Anwendungbeispiele in schrittweiser Erweiterung aufgeführt werden. Im ersten Szenario erfolgt die Dienstnutzung durch einen einzelnen Benutzer, welcher anhand eines generischen Klienten entfernte Prozeduraufrufe ausführt. Den *Engines* der Telescript-Technik entsprechend wird dargestellt, wie Dienstrepräsentationen als Kapselung des Evaluationszustandes von Suchagenten eingesetzt werden.

4.1. Der generische Klient

Der generische Klient fungiert als Bindeglied zwischen dem Benutzer und beliebigen entfernten Anwendungsdiensten, auf die der Benutzer ohne weitere Kenntnis ihrer Struktur zugreift. Entsprechend Abb. 1 wird zum Bindungszeitpunkt dazu vom referenzierten Server an den Benutzer eine DR übertragen und von GK interpretiert. Anhand der Spezifikationskomponenten der DR kann nun der GK *automatisch* Dialogfenster generieren, welche z.B. Editoren zur Modifikation von DR-Datenwerten und zum Aufruf entfernter Operationen bereitstellen. Anhand dieser Editoren können insbesondere Parameter- und Resultatwerte vor und nach Operationsaufrufen modifiziert werden. Eine entfernte Server-Operation kann der Benutzer schließlich durch das Betätigen einer - ebenfalls automatisch, anhand der DR generierten - Schaltfläche ausführen. Für diesen im Rahmen der Projekte COSM und TRADE [MML94] implementierten Spezifikationsumfang (siehe Kapitel 5) sind im einzelnen die folgenden Komponententypen der DR erforderlich:

- Spezifikation der Server-Schnittstelle (Operations- und Parameterkomponenten),
- Spezifikation der Benutzerschnittstelle (Dialogboxen, Editoren, Schaltflächen),

- Information über den Server-Knoten (Netzadresse, Bezeichnung des Dienstes etc.) und
- anwendungsspezifische Datenstrukturen (Parameterwerte, Zustandsinformationen).

Da dem GK aufgrund der dynamischen Bindung an einen Server Typinformationen über Dienstschnittstellen erst zur Laufzeit bekannt sind, kann kein statisch typisierter RPC-Mechanismus (wie z. B. bei OSF DCE [Schi93] oder beim Sun ONC) verwendet werden. Generischer Klient und Server-Anwendung nutzen aus diesem Grunde das CORBA-DII [OMG91] als Mechanismus der Interprozeßkommunikation. Hierbei werden zur Laufzeit Parameterrepräsentationen in Anlehnung an die in der DR abgelegten Schnittstellenspezifikationen konstruiert, an den Server übertragen und dort die betreffenden Operationen mit dieser *Named Value List* aufgerufen.

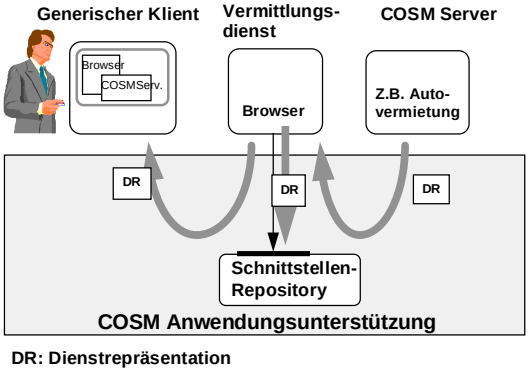

Bild 1: Der generische Klient zur Unterstützung des Dienstzugangs

Dienstrepräsentationen werden gemeinsam in Repositories verwaltet, so daß dieses Medium für Suchanfragen nach gewünschten Diensten, zum Blättern durch Spezifikationskomponenten (Browsing) aber auch zur privaten Ablage durch Server und generische Klienten verwendet werden kann (Abb. 1). Soll nun eine Server-Sitzung zwischen Benutzern transferiert werden, kann dies entweder direkt zwischen zwei zeitgleich aktiven GKs erfolgen oder indirekt über die Ablage der DR im Repository.

4.2. Steuerung von Suchagenten durch Dienstrepräsentationen

Zugriffe auf entfernte Dienste müssen nicht notwendigerweise ausschließlich über generische Klienten erfolgen, wenn Semantik und Schnittstelle des Dienstes bei der Implementierung eines Klienten bekannt sind. Im Gegensatz zum *generischen* ist daher in diesem Zusammenhang von einem *spezifischen Klienten* die Rede. Aufgabe eines spezifischen Klienten kann z. B. die Kombination existierender Dienste zu einem neuen sein, d. h., ein derartiger *Mehrwertdienst* sorgt aufgrund einer zusätzlichen Abstraktion für Transparenz bezüglich der von ihm genutzen *Ressourcendienste*.

Innerhalb dieses Spektrums zwischen generischen und spezifischen Klienten sind unterschiedliche Grade an Spezialisierung der Klienten bezüglich der Server-Semantik denkbar. Ein Beispiel stellen die aus dem Bereich der Datenbankrecherche bekannten *Suchagenten* dar, welche im Auftrage eines Klienten Suchanfragen an Datenbanken, Dateisysteme, Volltextrecherchesysteme etc. stellen. Die besondere Abstraktion liegt bei Suchagenten in der Kapselung datenbankspezifischer Recherche-Techniken, während sie selbst eine einheitliche Schnittstelle bereitstellen. Über diese Schnittstelle werden Suchagenten bezüglich Schlüsselwörter, Suchstrategie, zu konsultierender Knoten etc. konfiguriert. Eine aktuelle Entwicklung ist die *Telescript*-Technik des Unternehmens *General Magic* [Whit94], welche im Gegensatz zur DR eine Skriptsprache zur Steuerung autonomer Suchagenten verwendet.

Das gewählte Realisierungskonzept der DR bietet hinreichend flexible Datenstrukturen, um sowohl als Konfiguration für Suchagenten als auch als Träger für Rechercheergebnisse zu dienen: Eine DR kann zwischen *Engines* kommuniziert werden und - als Kapselung eines Evaluationszustandes - auszuführende Folgeoperationen determinieren.

Zusammenfassend dargestellt, erlaubt die DR generell den Transfer eines Ausführungskontexts als *Agent* zwischen räumlich verteilten *Engines*. Ein Agent ist somit eine generische, dienstunabhängige Instanz, welche lediglich die standardisierten Operationen implementiert. Für alle weiteren Operationen werden im Falle des Aufrufes anhand des CORBA-DII dynamische Parameterinstanzen erzeugt und versendet. Entsprechend der einleitend geforderten Fähigkeit der Systeminfrastruktur, eine dynamische Fortentwicklung des Gesamtsystems in Richtung einsatzspezifischer Teilstandards, kann der Einsatz von Agenten und die dafür erforderlichen standardisierten Operationen ohne Einwirkung auf den Repräsentationsstandard und ohne Seiteneffekt bzgl. anderer Interpretationen erfolgen.

4.3. Notariats- und Clearing-Dienste

Kurz erwähnt seien schließlich noch die im Rahmen des COSM-Projektes realisierten Beispiele von ”Notariats”- und ”Clearing”-Diensten (vgl. [MML94]): Da eine GK-Server-Bindung anonym erfolgt, gleichzeitig aber Dienste kommerziell angeboten werden und somit der Dienstnutzung ein Zahlungsstrom zugeordnet sein kann, besteht die Anforderung zum einen in der "Beurkundung" eines zustandegekommenen "Vertrages" zwischen Klient und Server, welche die Identität beider Vertragspartner sowie den Inhalts des Vertrages selbst festhält. In diesem Kontext wird die Bereitstellung einer Dienstrepräsentation als Vertrags*angebot* und der Operationsaufruf nach erfolgter Bindung als *Annahme* verstanden. Gemäß der Erweiterbarkeit der Dienstrepräsentation ist hierbei vorgesehen, auch Kosteninformationen einzubetten, welche anderen Spezifikationskomponenten (z.B. Operationen) zugeordnet sind. Wird nun zum Bindungszeitpunkt je eine vom Server versandte und eine vom generischen Klienten erhaltene Dienstrepräsentation an den Notariatsdienst zur Protokollierung gesendet, besteht im Falle eines juristischen Konflikts die Möglichkeit der Berufung auf die abgelegte Dienstrepräsentation, um z.B. ggf. nicht erbrachte Leistungen einzufordern. Ein zweites Beispiel für einen derartigen *trusted third-party service* sind ”Clearing”-Dienste, welche z.B. für den Transfer elektronischer Währungsrepräsentationen [MN93] im anonymen Zahlungsverkehr erforderlich sind. Hier werden in verschlüsselter Form Münzrepräsentationen vom Klienten an den Server gesendet. Im Kontext des Dienstemarktes läßt sich eine derartige Zahlungseinheit auf einfache Weise in die bestehende Parameterliste einbetten - auch an dieser Stelle bietet das CORBA-DII im Rahmen des COSM-Prototyps eine geeignete Implementierungsgrundlage [MML94].

5. Stand der Prototypimplementation

Verschiedene prototypische Anwendungen wurden mit Hilfe des CORBA-DII und der Dienstrepräsentation implementiert. Hierbei stellt das DII eine Eigenentwicklung dar, welche wahlweise in Verbindung mit dem Sun-RPC oder dem DCE RPC eingesetzt werden kann. Der generische Klient wurde basierend auf der portablen GUI-Bibliothek *StarView* entwickelt. Entwicklungsplattform ist ein IBM RS/6000-Cluster, jedoch sind bereits Portierungen von Teilen der COSM-Infrastuktur auf Sun/Solaris bzw. IBM OS/2 durchgeführt worden. Im Sommer '94 wurde der generische Client in seiner aktuellen Version fertiggestellt, so daß bereits einige Erfahrungen im Umgang mit Dienstrepräsentationen, DR-Repositories und verschiedenen Anwendungsdiensten gesammelt werden konnten [MML94b]. Ebenfalls wurde der Notariatsdienst [Drag95] in einer prototypischen Version fertiggestellt. Im aktuellen Stadium der Implementierung befindet sich u. a. die Unterstützung von autonomen Suchagenten.

6. Zusammenfassung

Im vorliegenden Beitrag wurden die besonderen Eigenschaften eines *offenen Dienstemarktes* dargestellt und die Vorteile der dortigen Autonomie gegenüber der organisationsinternen Konfiguration verteilter Anwendungen motiviert. Der Dienstrepräsentation - allgemein eingesetzt als generischer Träger beliebiger Information, jedoch in diesem Beitrag speziell als Instrument der Dienstbeschreibung - kommt hierbei eine besondere Bedeutung zu. Die Integration der dargestellten Einsatzbereiche der Benutzerunterstützung bei der Dienstnutzung, der Benutzerkoordination, der Steuerung verteilter Agentensysteme oder des dynamischen Transaktionsmanagements kann nur aufgrund der Flexibilität der Dienstrepräsentation erreicht werden. Hierbei spielt insbesondere die bewußte Trennung von generischen *Standards der Repräsentation* und spezialisierenden *Standards für Inhalte* eine grundlegende Rolle. Gegenüber sprachbasierten Ansätzen (HTML, Telescript), welche nur bei globaler Gültigkeit neue Versionen einführen können, bietet der vorgestellte Ansatz die Möglichkeit, für spezialisierte Einsatzbereiche *dezentralisiert* und *autonom* Erweiterungen an der Struktur von Dienstrepräsentationen und darauf spezialisierter Anwendungen vorzunehmen.

Wenn auch die Projekte COSM und TRADE hierbei nicht für sich in Anspruch nehmen können, empirisch den Nachweis für die Angemessenheit der vorgestellten Architektur zur Unterstützung offener Dienstemärkte erbracht zu haben, so scheint doch auch aufgrund ähnlicher Entwicklungen (WWW bzw. Telescript-Technik) und deren Erfolgsfaktoren der prinzipielle Ansatz besonders vielversprechend, Koordinationsmechanismen realer Märkte auch für die Realisierung verteilter Anwendungen zu nutzen. Schließlich erbringt der Mechanismus des Marktes täglich den Nachweis einer dezentralisierten, effizienten und hinreichend reagiblen Koordinationsform für den Gütertausch realer Anbieter und Nachfrager.

7. Referenzen

[Drag95] Dragnev, K.: „Entwurf eines Notariatsdienstes zur Protokollierung von Client/Server-Bindungen in offenen Kommunikationsumgebungen“, Studienarbeit, Universität Hamburg, Fachbereich Informatik, 1995

[ISO-ODP] ISO/IEC JTC1 SC21 WG7: Trader, Working Document N7047, 1992

[Kräh94] Krähenmann, N.: ”Ökonomische Gestaltungsanforderungen für die Entwicklung elektronischer Märkte”, Diss., Nr. 1553, Hochschule St. Gallen, 1994

[MML94] Müller, K., Merz, M., Lamersdorf, W.: ”Trusted Third Party Services in COSM”, EM - Electronic Markets, 4(12), S. 7f, Sept. 1994

[MML94b] Müller, K., Merz, M., Lamersdorf, W.: ”Service trading and mediation in distributed computing systems”, in Proc. ICDCS ’94, IEEE Computer Society Press, 1994, S. 450ff

[MML95] Müller-Jones, K., Merz, M., Lamersdorf, W.: ”The TRADEr: Integrating Trading into DCE”, to appear in Proc. ICODP, Third intl. Conference on Open Distributed Processing, 1995

[OMG91] OMG: ”The Object Request Broker: Architecture and Specification”: OMG Document No. 91.12.1, Object Management Group, Framingham, MA, USA, 1991

[Schi93] Schill, A.: ”DCE: Das OSF Distributed Computing Environment: Grundlagen und Anwendungen”, Springer, New York, Berlin, Heidelberg, 1993

[Whit94] White, J. E.: "Telescript Technology: The Foundation for the Electronic Marketplace", White Paper, General Magic Inc., 1994