

# **ChariTime – Ein Agenten-orientiertes Softwaresystem zur Terminplanung im Krankenhaus**

Dipl. Inform. Ines Münch, Dipl. Inform. Marcel Gnoth

## **1 Einleitung**

Im Oktober 1998 wurde zwischen der Medizinischen Klinik und Poliklinik mit Schwerpunkt Kardiologie, Angiologie und Pulmologie des Universitätsklinikums Charité, dem Lehr- und Forschungsgebiet Künstliche Intelligenz des Instituts für Informatik der Humboldt-Universität zu Berlin und der GMD - Forschungszentrum Informationstechnik GmbH eine Kooperationsvereinbarung getroffen. Die generelle Erwartung an dieses Forschungsprojekt ist die Verbesserung der Koordination zwischen den Stationen, der Poliklinik und den Funktionsbereichen der kardiologischen Klinik (Klinik I) durch den Einsatz eines Softwaresystems. Es soll eine bessere Ressourcenauslastung der diagnostischen Bereiche, eine Minimierung der Patientenwartezeiten und eine bessere Informiertheit aller Beteiligten erreicht werden. Da das geschaffene System zum täglichen Einsatz kommen soll, ist die professionelle Weiterentwicklung und Betreuung über den Rahmen eines Forschungsprototyps hinaus notwendig. Für diesen Aufgabenteil zeichnet der industrielle Kooperationspartner, die NTeam GmbH verantwortlich.

Im Rahmen dieser Kooperation ist ChariTime, ein Agenten-orientiertes Softwaresystem, entstanden. Seit Dezember 2000 befindet sich dieses System in seiner Einführungsphase in der Klinik I der Charité. ChariTime wird als Produkt von der NTeam GmbH kommerziell vermarktet. In Zusammenarbeit mit dem zentralen Rechenzentrum der Charité soll das Produkt so erweitert werden, dass es in allen Kliniken des Universitätskrankenhauses eingesetzt werden kann.

In der kardiologischen Klinik der Charité wird eine Vielzahl komplexer medizinischer Beratungs-, Diagnose- und Therapieleistungen erbracht. Bedingt durch den qualitativen und quantitativen Umfang dieser Leistungen und die räumliche Aufgliederung der Klinik ist eine funktionale Aufteilung der stationären, poliklinischen und diagnostischen Abteilungen entstanden, die gravierende Koordinierungsprobleme mit sich brachte.

In der Klinik stellen die einzelnen Bereiche autonom agierende Einheiten dar, die ihre Termine selbstständig koordinieren und das dazu notwendige Wissen lokal verwalten. Eine Abbildung dieser Abteilungen auf Agenten, die als ihre Interessenvertreter an der Terminkoordination teilnehmen, liegt deshalb nahe. Auch die Patienten werden in ChariTime durch Agenten vertreten. An diesem Punkt wird im Softwaresystem von der Realität abgewichen, insofern dort die Stationen bzw. Sprechstunden die Interessen der Patienten vertreten. Die Patienteninteressen fließen jetzt explizit beim Softwareentwurf durch die Modellierung des Patienten-Agenten in die Terminkoordination ein. So soll ein solcher Agent beispielsweise versuchen, zusammenhängende und kurzfristige Termine zu beschaffen und die Einhaltung vereinbarter Termine zu überwachen.

## 2 Systemaufbau

### 2.1 Grundlegende Systemanforderungen

Die an die Software gestellten Anforderungen machen die Entwicklung eines offenen und verteilten Systems zur Terminkoordination notwendig. Oberflächenkomponenten müssen an verschiedensten (heterogenen) Arbeitsplätzen zur Verfügung stehen und das zur Terminkoordination notwendige Wissen sollte (zur Vermeidung von Redundanzen etc.) in der Obhut der entsprechenden Bereiche (bzw. ihrer Agenten) verbleiben, die dann für die Pflege ihrer Daten zuständig sind.

Der Aufbau des Agenten-orientierten Systems orientiert sich teilweise an den Vorschlägen der FIPA in [FIPA98]. Unter Agenten werden in ChariTime gekapselte Softwareeinheiten verstanden, die über einen eigenen Zustand, eigenes Verhalten und eigene Rechenzeit verfügen. Diese Agenten können mit anderen Einheiten wie Agenten bzw. menschlichen Anwendern interagieren, um ihre Ziele zu erreichen. Ein Agent unterscheidet sich hier von einer passiven Komponente dadurch, dass er die Möglichkeit hat, Nachrichten zu verschicken und zu empfangen, um semantisch motivierte Aufgaben zu erfüllen. Rein passive Komponenten nehmen nicht am Nachrichtenaustausch teil und erledigen innerhalb des Systems nur technisch relevante Funktionen im Rahmen der Aufgaben der Agenten.

Ein Weg, ein solches Agenten-orientiertes System zu konstituieren, ist die Umsetzung von Agenten, die miteinander direkt kommunizieren können. Für die Realisierung der einzelnen Aspekte eines solchen Systems müssen konkrete Bedingungen erfüllt sein:

- Die Agenten müssen über Aufgaben bzw. daraus resultierende Ziele verfügen.
- Um eine aufgabenspezifische Kommunikation zu ermöglichen, muss ein gemeinsames Verständnis der Interaktionspartner (Agenten) über ihre Begriffswelt existieren (Ontologie).
- Sollen die Agenten die Interessen realer Organisationseinheiten vertreten, muss eine entsprechende Zuständigkeit ihrerseits für diese definiert werden.
- Für die Kommunikation benötigen sie ein gemeinsam verwendbares Kommunikationssystem, innerhalb dessen es eine eindeutige Adressverwaltung geben muss, deren Erreichbarkeit jedem Agenten bekannt ist.

Alle Agenten müssen Basisinformationen (in erster Linie die Adressen der Nachrichtenwarteschlangen) der anderen Agenten abfragen können. Es müssen Aufgaben der administrativen Systemverwaltung von zentralen Komponenten erledigt werden, wobei das aufgrund der Interessenvertretung durch die Agenten notwendige Zuständigkeitsmanagement gesondert berücksichtigt wird. Diese Dienste werden von DirectoryFacilitator (DF) und SystemManagementAgent (SMA) übernommen. Der DF übernimmt alle Standardaufgaben des täglich anfallenden, größtenteils fachlich geprägten Managements. Der SMA hingegen ist hauptsächlich für die technische Verwaltung der KnowledgeAgents (s.u.) und ihrer „Lebensräume“ (Workspaces) verantwortlich und kontrolliert das Hinzufügen bzw. Löschen aller am System beteiligten Agenten.

Die Agenten kommunizieren ausschließlich über asynchrone Nachrichten miteinander. Dies ist Voraussetzung für die Integration weiterer Agenten zu jedem beliebigen Zeitpunkt. Diese Form der Kommunikation verhindert außerdem, dass Agenten Aufgaben nicht erledigen können, wenn die Erreichbarkeit einzelner Interaktionspartner nicht gegeben ist. (Es können in diesem Fall beispielsweise andere Interaktionspartner gesucht werden.) Die Kommunikation der Agenten wird durch passive Komponenten unterstützt, die diverse Basisfunktionen zur Verfügung stellen.

## 2.2 Agentenmodell

Im Agentenmodell von ChariTime gibt es zwei Agententypen, die über ihre aufgabenbezogenen Verantwortlichkeiten bezüglich der zu vertretenen Personen bzw. Personengruppen (im weiteren Text auch Organisationseinheiten genannt) im System definiert sind.

Die Informationen zu den einzelnen Organisationseinheiten werden von **KnowledgeAgents** verwaltet. Diese leisten die inhaltliche anwendungsbereichsbezogene Arbeit, vertreten also die Interessen der beteiligten Personen bzw. Abteilungen bei der Terminkoordination. Sie arbeiten dabei mit dem Fachwissen über die einzelnen Organisationseinheiten. Ein KnowledgeAgent kann für mehrere Organisationseinheiten zuständig sein, während es für eine Organisationseinheit immer genau einen zuständigen KnowledgeAgent gibt.

Die **UserAgents** bilden die Schnittstellen zum Anwender und nehmen ihre Handlungsanweisungen auf bzw. leiten diese weiter. UserAgents fordern von den KnowledgeAgents Wissen an, um es dem Nutzer über eine grafische Oberfläche anzeigen zu lassen. Ein UserAgent kann das Wissen mehrerer KnowledgeAgents an den Anwender weitergeben. (Dabei regelt eine Rechteverwaltung die Verfügbarkeit von Informationen für die UserAgents.)

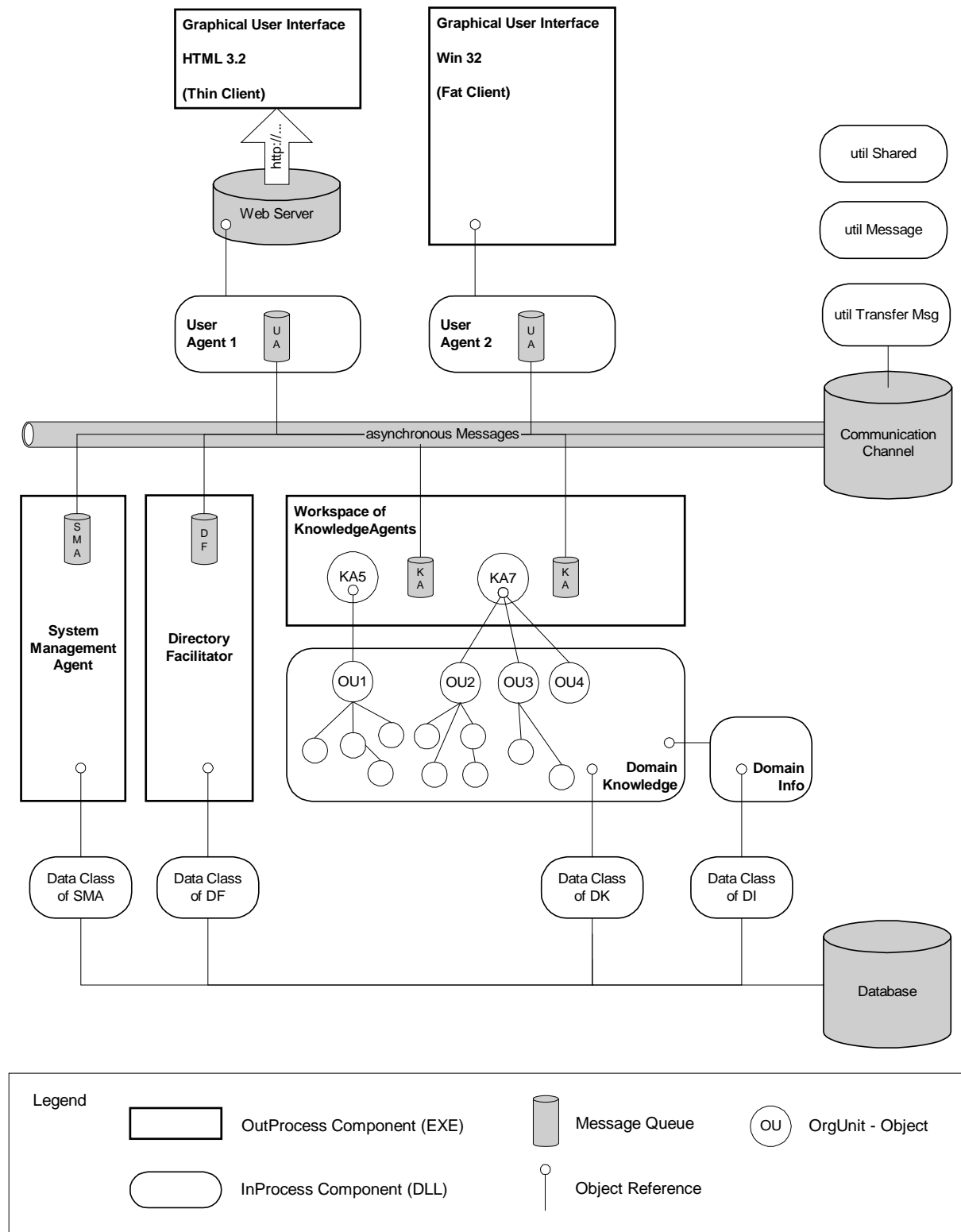
## 2.3 Systemarchitektur

ChariTime ist als komponentenbasiertes Softwaresystem implementiert. Die einzelnen Komponenten sind objektorientiert aufgebaut. Das System besteht aus aktiven Komponenten (Agenten), die über die Möglichkeit verfügen, Nachrichten zu verschicken und passive Komponenten, die dies nicht können.

Die aktiv am System beteiligten Komponenten (UserAgents und KnowledgeAgents, DirectoryFacilitator und SystemManagementAgent) können Nachrichten erhalten und verschicken. Dies ist durch eine „Nachrichtentonne“ in der Grafik verdeutlicht. Der DirectoryFacilitator und der SystemManagementAgent sind als zentrale Komponenten in einer Instanz im System vertreten. Sie legen Informationen mit Hilfe ihrer Datenklassen in einer Datenbank ab. Die Datenspeicher von ChariTime sind der Nachrichten- und der Datenbankserver.

Die grafische Oberfläche ist derzeit sowohl in Form eines Win32-Clients als auch in Form eines wesentlich „schlankeren“ HTML-Clients implementiert bzw. entworfen worden. Die beim Anwender zu installierende Oberfläche richtet sich letztendlich nach den dort angetroffenen technischen Gegebenheiten. Grundsätzlich sind plattformunabhängig alle Arten von Oberflächen denkbar, die mit der Schnittstelle des UserAgent interagieren können. An jedem Computer-Arbeitsplatz wird durch den Anwender zusammen mit der grafischen Oberfläche genau ein UserAgent gestartet. Dieser steuert die grafische Oberfläche zur Interaktion mit dem Anwender, kontrolliert die Nachrichtenwarteschlange und kommuniziert mit den anderen Agenten. Dies sind hauptsächlich die KnowledgeAgents, die den UserAgent mit dem erforderlichen Fachwissen über die Organisationseinheiten versorgen.

Die KnowledgeAgents „leben“ in Workspaces und sind zuständig für den Zugriff auf das Fachwissen ihrer *COrgUnit*-Objekte. Dafür existieren die beiden Komponenten *CTDomainKnowledge* (für terminierungsrelevante Angaben) und *CTInfoData* (für rein informative Angaben) mit ihren Datenklassen. Das Fachwissen aller KnowledgeAgents wird derzeit noch zentral in einer Datenbank gespeichert. Die Zugriffsrechte für das Fachwissen sind jedoch eindeutig (anhand der Zuständigkeit für Organisationseinheiten) unter den KnowledgeAgents aufgeteilt, so dass das Fachwissen bei Bedarf auch verteilt abgelegt werden kann.



**Bild 1** Systemarchitektur von ChariTime

Es gibt drei technische Hilfskomponenten, die von fast allen anderen Komponenten benötigt werden:

- *CTUtilShared* (enthält eine Reihe von Hilfsfunktionen, die alternativ auch redundant in den auf sie angewiesenen anderen Komponenten implementiert werden könnten)
- *CTUtilTransferMsg* (kapselt den Zugriff auf das verwendete Nachrichtensystem und ermöglicht seine prinzipielle Austauschbarkeit)

- *CTUtilMessage* (gewährleistet ein einheitliches Format der Nachrichten, enthält die Aufzählungen aller möglichen Nachrichtentypen und Dienste der Agenten und ermöglicht die Kontrolle der Einhaltung der syntaktischen Kommunikationsprotokolle).

Da im Mittelpunkt dieser Ausführungen der inhaltliche Entwurf des Systems steht, wird hier nicht näher auf die konkrete Implementation der einzelnen Komponenten eingegangen.

### 3 Interessenvertretung durch Agenten

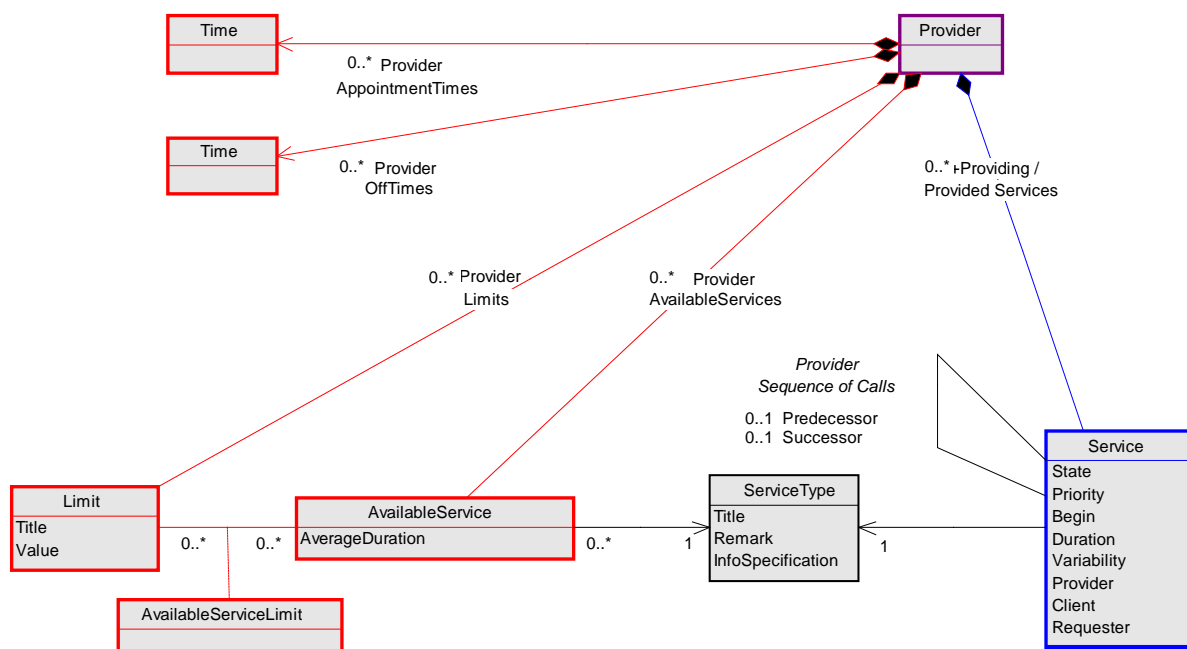
#### 3.1 Fachwissen der Agenten

Beide beschriebenen Agententypen arbeiten mit dem Fachwissen der Organisationseinheiten. Das Fachwissen ist in Form eines Objektmodells implementiert worden. Der zu persistierende Anteil der konkreten Informationen wird in einer relationalen Datenbank abgelegt.

Zwischen den einzelnen Organisationseinheiten sind die verschiedensten Beziehungen denkbar. Alle Typen von Beziehungen zwischen Organisationseinheiten werden mit einer Beschreibung der möglichen Kardinalitäten zwischen den an der Beziehung beteiligten Organisationseinheiten festgehalten. So ist z. B. die Abbildung einer Organisationsstruktur (eine bzw. mehrere aufgabenbezogene organisatorisch verantwortliche Einheiten) möglich.

Das Wissen über eine Organisationseinheit (*OrgUnit*) ist rollenspezifisch aufgebaut. Für jede *OrgUnit* kann über die Eigenschaften *IsProvider*, *IsClient*, *IsRequester* und *IsSeller* festgelegt werden, welche Rollen sie bezüglich des Dienstleistungsmanagements potenziell einnehmen kann. Dementsprechend verfügt sie über den zur Wahrnehmung dieser Rolle notwendigen Anteil an Informationen. Konkret sind dies die bei der Terminierung einer Dienstleistung zu berücksichtigenden Fakten.

Beispielhaft wird im folgenden für die Rolle Leistungserbringer erklärt, welche Angaben die Agenten benötigen und wie sich diese im Objektmodell widerspiegeln.



**Bild 2** Klassendiagramm der Angaben eines Leistungserbringers

Der potenzielle Leistungserbringer verfügt über das Spektrum der durch ihn erbringbaren Leistungen (*AvailableServices*) einschließlich ihrer durchschnittlichen Ausführungsdauer.

Jede Leistungsart enthält einen Verweis auf die „typisierte Bezeichnung“ *ServiceType*, unter der sie den erreichbaren Leistungsnehmern angeboten wird. Es muss sichergestellt sein, dass eine gemeinsame Vorstellung der Kommunikationspartner von den sich hinter diesen Dienstleistungsarten verbergenden Leistungen herrscht, d.h. sie müssen gegebenenfalls noch öffentlich beschrieben werden.

Es werden die regelmäßigen Arbeitszeiten und Auszeiten (pro Wochentag) in den Sperrzeiten für die Leistungserbringung (*ProviderOffTimes*) erfasst, die die generelle (Nicht-) Verfügbarkeit des Leistungserbringers kennzeichnen. Zudem legt jeder Leistungserbringer fest, welches mögliche Zeiträume sind, in denen Dienstleistungen durch das System terminiert werden dürfen (*ProviderAppointmentTimes*). Diese Zeiten können sowohl in Form von Zeitregeln als auch in Form von konkreten Zeiträumen angegeben werden.

Zur zeitlichen Einordnung der Dienstleistungen sind neben der generellen Verfügbarkeit die speziellen Sperrzeiten in Form der für andere Ereignisse bereits reservierten Zeiträume wichtig (andere Leistungen inkl. ihrer Eigenschaften *Begin* und *Duration*). Zusätzlich zu diesen offensichtlichen zeitlichen Einschränkungen sollen sowohl personelle als auch gerätetechnische Beschränkungen berücksichtigt werden, die als Einschätzungen der aus ihnen resultierenden Leistungsfähigkeit des Leistungserbringers einfließen (*Limits*).

Alle terminierungsrelevanten Angaben werden durch einen Transformationsprozess in prädikatenlogische Ausdrücke überführt (siehe [HaMü00]) und vom KnowledgeAgent zur Realisierung seiner Aufgaben im Rahmen der Terminkoordination verwendet.

## 3.2 Verarbeitung des Wissens durch die Agenten

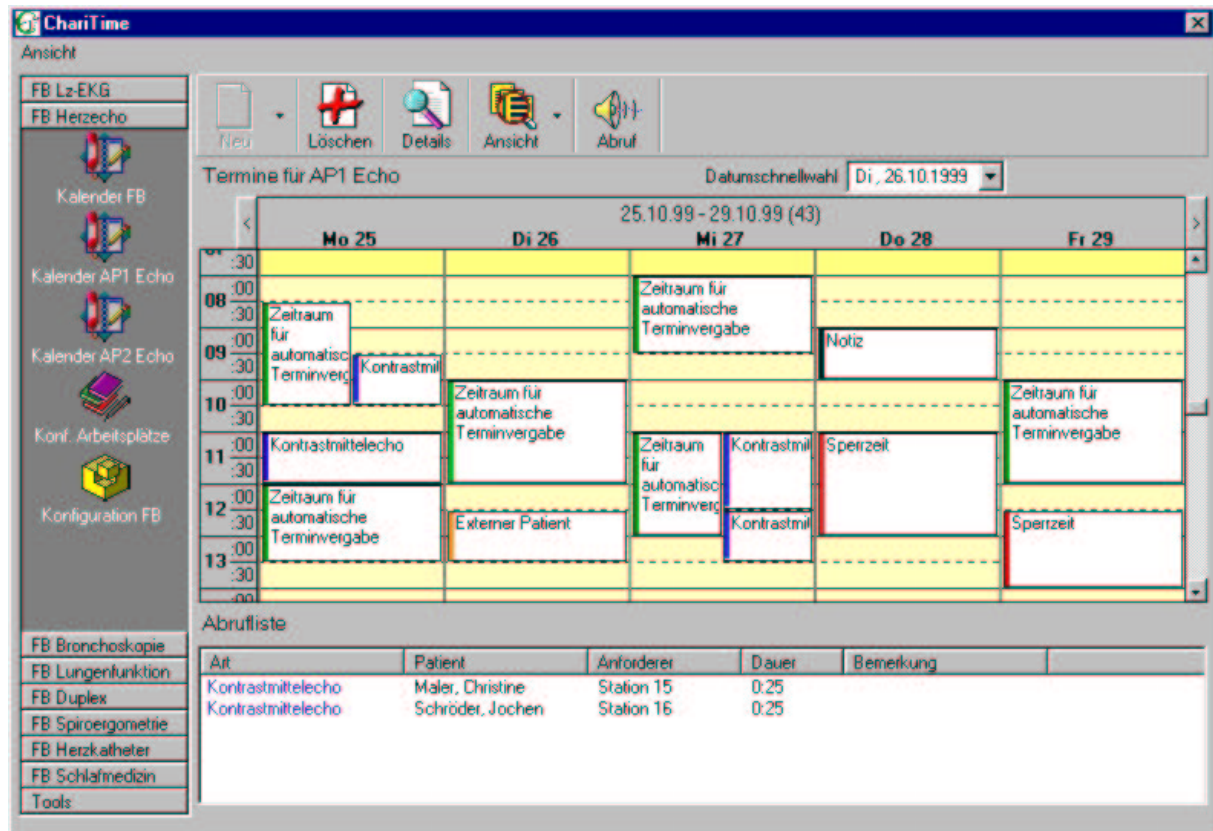
KnowledgeAgents verfügen nur über das Wissen der Organisationseinheiten, die ihnen zugeordnet sind. Sie haben keine Informationen über Organisationseinheiten anderer KnowledgeAgents. Diese müssen sie bei Bedarf von den anderen Agenten über das Nachrichtensystem erfragen.

Auf Grund der Verantwortung des KnowledgeAgent für reale Organisationseinheiten muss er aktiv Umweltbedingungen im System überwachen und unter Umständen selbstständig aktiv werden (z. B. nach Notfall wird der Funktionsbereich geschlossen, er muss sowohl UserAgents benachrichtigen als auch einen neuen Termin aushandeln bzw. alternative Dienstleister suchen). Dieses Verhalten wird durch eine BDI-Architektur des KnowledgeAgent realisiert (siehe [RaGe95]). Der KnowledgeAgent hat ein Umweltwissen (Belief), leitet daraus Wünsche (Desire) ab und versucht, diese Wünsche mit konkreten Handlungen (Intentions) umzusetzen.

In Abhängigkeit von den Rechten der UserAgents werden die Informationen für die entsprechenden Organisationseinheiten an der grafischen Oberfläche präsentiert. Dabei werden die zwischen den Organisationseinheiten bestehenden Beziehungen berücksichtigt. Auf der Seite der Patientenverwalter und Anforderer ist dies die organisatorische Verantwortlichkeit für Patienten, auf der Seite des Anbieters die Zusammensetzung der diagnostischen Bereiche aus einzelnen Untersuchungsplätzen. Das Recht zur Ansicht der jeweils übergeordneten Organisationseinheit schließt das Recht auf die Informationen derart „untergeordneter“ Organisationseinheiten ein.

Für die Visualisierung der Daten einer Organisationseinheit können mehrere UserAgents registriert werden. So ist es zum Beispiel denkbar, dass leitende Personen sich alle Daten aller Organisationseinheiten anzeigen lassen, während parallel dazu an den anderen Computer-Arbeitsplätzen nur die dort benötigte (Teil-) Ansicht an Organisationseinheiten sichtbar ist.

Einen Ausschnitt der Sicht an einem zentralen Arbeitsplatz (hier z. B. für alle Funktionsbereiche der Klinik I) zeigt Bild 4. Für die Anwender der anbietenden Abteilungen (Funktionsbereiche und Sprechstunden) werden in einer kalendarischen Übersicht alle vereinbarten Termine bzw. die abzurufenden Patienten angezeigt. Hier sollen durch die Anwender unter anderem die Zeiten, in denen KnowledgeAgents selbstständig Termine eintragen dürfen, freigegeben werden (bezeichnet als „Zeitraum für automatische Terminvergabe“).



**Bild 3** Bildschirmmaske aus Sicht der anbietenden Bereiche

Für die Anwender der anfordernden Abteilungen (Stationen und Sprechstunden) beinhalten diese die Patientendaten und alle Angaben über ihre Untersuchungen. Die Stationsschwester legt für die KnowledgeAgents der Patienten die Rahmenbedingungen für die Terminkoordination fest. Dazu gehört beispielsweise die maximal zumutbare Wartezeit zwischen zwei Untersuchungen.

## 4 Zusammenfassung

Mit ChariTime ist ein Agenten-orientiertes Softwaresystem entstanden, in dem Agenten als Interessenvertreter von Personen bzw. Personengruppen miteinander interagieren können, um bestimmte Aufgaben zu erfüllen. Die beschriebene Kombination von Konzepten zur Terminkoordination von Organisationseinheiten und der Definition von Verantwortlichkeiten und Rollen der Agenten kann als grundlegendes Modell für die Übertragung von Handlungsträgerschaft des Menschen auf Agenten verwendet werden. Die Agenten sind nicht nur in der Lage, Dienstleistungen in der Klinik I der Charité zu managen, sondern dieses System kann auch perspektivisch in mehreren Kliniken oder Verwaltungsbehörden mit komplexen Strukturen eingesetzt werden. Die dem System zugrunde liegenden Konzepte und Implementationsstrategien ermöglichen eine vielseitige Vermarktung des entstandenen Produktes.

## Literatur

- [Booc99] G. Booch, J. Rumbaugh, I. Jacobson *The Unified Modeling Language User Guide*. Addison-Wesley 1999
- [FIPA98] FIPA, *FIPA 98 Specification Version 1.0*. siehe <http://www.fipa.org>
- [Fowl99] M. Fowler *Analysemuster Wiederverwendbare Objektmodelle*. Addison-Wesley 1999
- [GnMü99] M. Gnoth, I. Münch *ChariTime Systementwurf*. Studienarbeit am Institut für Informatik der HU Berlin, Lehrstuhl für KI. 1999
- [HaMü00] M. Hannebauer, I. Münch *Transforming Object-Oriented Domain Models into Declarative CLP Expressions*. Accepted for WLP 2000, Würzburg
- [Münc00] I. Münch, G. Lindemann v. Trzebiatowski *ChariTime – Concepts of Analysis and Design of an Agent-Oriented System for Appointment Management*. Fundamenta Informaticae 2000
- [JeWo98] N. R. Jennings, M. J. Wooldridge; 1998. *Agent Technology – Foundations, Applications, and Markets*. Springer.
- [RaGe95] A. S. Rao, M. P. Georgeff 1995. *BDI Agents: From Theory to Practice*. In Lesser, V.~ed.~Proceedings of the 1st International Conference on Multi-Agent Systems (ICMAS):312--319. MIT Press.
- [Step98] Larry M. Stephens, M. L. Dowell and R. D. Bonnell *Using A Domain-Knowledge Ontology as a Semantic Gateway among Information Resources*, in Readings in Agents, M. N. Huhns and M. P. Singh (eds.), Morgan Kaufmann Publishers Inc., San Francisco, CA, 1998, pp. 255-260.