Institut für Architektur von Anwendungssystemen
Universität Stuttgart
Universitätsstraße 38
D-70569 Stuttgart

Diplomarbeit Nr. 3054

Visualisierung von Person-Centric Flows

Hristina Hristova

Studiengang:	Informatik
Prüfer:	Prof. Dr. Frank Leymann
Betreuer:	Dipl.-Inf. Tobias Unger
begonnen am:	15.02.2010
beendet am:	17.08.2010
CR-Klassifikation:	H.1.2, H.4.1, H.5.2

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Unternehmen müssen heutzutage viele Geschäftsprozesse unterstützen, an denen verschiedene Partner, Abteilungen und Bearbeiter beteiligt sein können. In diesem Zusammenhang existiert ein steigendes Interesse an Prozessmanagement-Technologie ebenso wie aufkommenden Standards für die Orchestrierung und Choreographie von Prozessen bzw. der Services, die durch sie verknüpft sind.
Daher setzt die Forschung im Bereich des Workflow Managements ihren Fokus zunehmend auf die Service-Orchestrierung. Oft wird die Tatsache vernachlässigt, dass ein großer Teil der Aktivitäten von Geschäftsprozessen von Menschen durchgeführt wird. Insbesondere im Bereich von „Pervasive Computing Processes“ werden Abläufe von realen Aktivitäten beschrieben, die immer von Menschen erledigt werden. Daher ist es wichtig, die Rolle der Menschen, die an den Workflows beteiligt sind, von einer neuen Perspektive zu betrachten. In diesem Kontext wird das Konzept eines Person-Centric Flows eingeführt, welches einen impliziten Flow bezeichnet, der von einer einzelnen Person geplant und ausgeführt wird.
Durch die Fragmentierung von Prozessimplementierungen auf verschiedene operative Systeme sowie die unternehmensübergreifende Vernetzung der Prozesse steigt gleicherweise die Komplexität an. Vor allem der Gesamtzusammenhang und der Überblick über den Gesamtprozess gehen dabei oft verloren.
Eine integrierte, graphische Visualisierung des Gesamtprozesses wäre hier sehr nützlich. Mit einer derartigen Visualisierungskomponente könnten sich die Prozessbeteiligten Informationen über den aktuellen Ausführungszustand beschaffen. Der einzelne Prozessbearbeiter kann durch das Wissen der Prozessstruktur die eigenen Aufgaben besser in den Gesamtprozess einordnen. Zugleich wird die Wahrnehmung der eigenen Verantwortung für die Prozessergebnisse erhöht. Aus der Visualisierung des Gesamtprozesses könnten außerdem die anderen Prozessbeteiligten ersichtlich werden, so dass die Kommunikation mit Bearbeitern nachfolgender Aktivitäten vereinfacht bzw. erst ermöglicht wird. Der aus der Prozessdarstellung zu entnehmende Bearbeitungszustand des Prozesses macht dem Beteiligten eine bessere Planung der eigenen zukünftigen Aufgaben möglich. Er kann rechtzeitig auf Verzögerungen reagieren und besser einschätzen, wie sich Verzögerungen der eigenen Aufgaben auf den Geschäftsprozess auswirken (z.B. ob sie auf dem kritischen Pfad liegen) [3], [22].
Ziel dieser Arbeit ist es, die Visualisierung von Person-Centric Flows zu untersuchen und zu realisieren. Es soll ein konkretes Szenario der Visualisierung prototypisch implementiert werden.

1.1 Problemstellung und Fragestellungen

Für prozessorientierte Applikationen besteht die konkrete Notwendigkeit, Prozesse zu visualisieren. Dabei soll die Visualisierung an die Bedürfnisse der Betrachter angepasst werden können, beispielsweise in Hinsicht auf Detaillierungsgrad, verwendete Symbole sowie Menge der angezeigten Daten. Im Allgemeinen ergeben sich folgende Forschungsfragestellungen:

Integration der heterogenen Prozesssysteme:
In verteilten Prozesslandschaften müssen vor der Visualisierung die Prozessmodelldaten aus den unterschiedlichen Quellsystemen integriert und homogenisiert werden. Des Weiteren müssen für die Visualisierung von Prozessinstanzen sowohl aktuelle Statusdaten als auch Applikationsdaten betrachtet werden. Mögliche Quellsysteme beziehen neben Workflow Management Systemen (WfMS) auch konventionell implementierte, prozessorientierte Anwendungssysteme, Altapplikationen und Datenbanken ein.

Anpassung des Detaillierungsgrades der Prozessmodelle:
Prozessmodelle sind häufig sehr detailliert und umfassen viele Schritte, d.h. sie enthalten Prozesselemente, die je nach Szenario für den Betrachter unwichtig sein können bzw. unnötig in der Komplexität der Prozessmodelle und somit in der Verunsicherung des Betrachters mitwirken (z.B. zugeordnete Systeme oder Dokumente). Jedoch benötigen die einzelnen Benutzergruppen unterschiedlich detaillierte Prozessdarstellungen. Man braucht daher ein Konzept, um Details der Prozessmodelle gegebenenfalls zu verbergen. Dadurch kann einerseits die Komplexität reduziert und andererseits die Visualisierung besser an die Bedürfnisse der Betrachter angepasst werden. Natürlich erfordert dies auch Änderungen an der Prozessstruktur.

Konfiguration der Darstellung:
Gegenwärtige Prozesswerkzeuge sind bezüglich der graphischen Gestaltungsfreiräume einer Prozessdarstellung sehr stark beschränkt. Oft besteht allerdings der Wunsch, die verwendete Notation individuell an die Erfordernisse der Anwendung bzw. Benutzer anzupassen. Die Prozessnotation muss dazu frei konfigurierbar sein, zum einen was die Gestaltung der Symbole (Form und dargestellte Daten) und zum anderen die flexible Zuordnung der Symbole zu Prozesselementen betrifft (in Abhängigkeit von beliebigen Prozessdaten, Ausführungszuständen und Applikationsdaten).

Bereitstellung alternativer Darstellungsformen:
Prozesse werden herkömmlich in Form eines Graphen dargestellt. Jedoch eignen sich für bestimmte Anwendungen andere Formen der Darstellung besser, abhängig davon welcher Aspekt für den Betrachter äußerst wichtig ist. Infolgedessen muss ermittelt werden, welche alternativen Darstellungsformen realisiert werden können und sich für eine zielgruppengerechte Visualisierung anbieten.

Berechnung eines Prozessgraph-Layouts (d.h. Positionierungsinformation für die Prozesselemente auf der Zeichenebene in einem Prozessgraphen):
Prozesse werden in heutigen Systemen im Prinzip so dargestellt, wie sie zuvor vom Prozessentwickler gezeichnet wurden. Die Darstellungen ändern sich mit der Umsetzung der oben beschriebenen Anpassungsmöglichkeiten dynamisch. Damit sind die vorher gültigen Positionsdaten nicht mehr ideal. Wenn die Prozessvisualisierung darüber hinaus dynamisch ist, d.h. der Detaillierungsgrad und das Aussehen hängen

von Laufzeitdaten (z.B. Ausführungszustand des Prozesses) ab, wird eine Positionierung von Hand zu anstrengend. Man braucht deswegen passende Algorithmen, um ein geeignetes Prozess-Layout berechnen zu können [22].

1.2 Gliederung und Aufbau der Arbeit

Die Diplomarbeit ist in einen theoretischen und einen praktischen Teil gegliedert und folgendermaßen strukturiert:
Kapitel 2 gibt einen Überblick und eine Einleitung über das Konzept von Person-Centric Flow. In diesem Zusammenhang werden die Begriffe Tasks und Aktivitäten eingeführt und die Rolle des Task Managers erläutert. Des Weiteren werden Person-Centric Flows definiert, sowie auf deren Charakteristiken und Eigenschaften näher eingegangen.
In Kapitel 3 wird der Einsatz von Person-Centric Flows in der Praxis diskutiert und ihre Nützlichkeit anhand von Beispielen im Bereich des Gesundheitswesens und innerhalb eines Unternehmens gezeigt. Darüber hinaus werden hier zwei Möglichkeiten für die Verteilung von Anleitungsinformationen – Aktivitäts-Erkennung (Activity Recognition) und direkte Navigation – erörtert.
In Kapitel 4 werden Definitionen des Begriffs Visualisierung eingeführt. Betrachtet werden hier die Anpassung der Visualisierung bezüglich der Graphik und der Struktur, sowie die Daten als Ausgangspunkt einer jeden Visualisierung. Anschließend wird der grobe Ablauf der Visualisierung beschrieben und auf deren Darstellungsformen eingegangen.
Kapitel 5 stellt die Anforderungsanalyse dar, die sowohl Anforderungen an eine graphische Konfiguration als auch an die Inhalte von Prozessvisualisierungen mit einbezieht, wobei hier zwischen funktionalen und nicht-funktionalen Anforderungen unterschieden wird.
Kapitel 6 beschreibt das umzusetzende Konzept und die Annahmen, die im Rahmen dieser Arbeit getroffen werden. Des Weiteren wird in diesem Kapitel auf die Idee, das zu implementierende Szenario und die verwendete Prozessnotation eingegangen.
Kapitel 7 wird dem praktischen Teil dieser Diplomarbeit gewidmet, der Implementierungsphase. Besprochen werden hier die getroffenen Entscheidungen und die Details in Bezug auf die prototypische Realisierung der Visualisierung. In diesem Zusammenhang wird auf den eigentlichen Ablauf und das User Interface, und auch auf die Architektur der prototypischen Visualisierung eines Person-Centric Flows als eine Anwendung, deren Schichten und die eingesetzten Technologien eingegangen.
Eine Zusammenfassung, die Ergebnisse dieser Arbeit, sowie ein Ausblick werden in Kapitel 8 dargestellt.

2 Einführung von Person-Centric Flow

Menschen sind zunehmend in eine Umgebung eingebettet, bestehend aus unzähligen Computern und Produkten, die verschiedene Rechenleistungsgrade und Erkenntnisse bieten. Pervasive Anwendungen sind Softwaresysteme, die in solchen Umgebungen in einer weit verteilten Art und Weise laufen und mobile menschliche Benutzer in ihren täglichen Aktivitäten unterstützen [17].

In dieser Arbeit wird die Aufmerksamkeit auf eine neue Forschungslinie konzentriert, gerichtet auf die Entwicklung einer neuen Programmierung und eines menschlichen Schnittstellenansatzes für pervasive Systeme, basierend auf Modellen von menschlichen Aktivitäten auf hoher Ebene, sogenannte Person-Centric Flows oder kurz nur Flows genannt [1].

Bevor im Folgenden die Definition eines Person-Centric Flows eingeführt wird, werden zunächst die Bestandteile eines solchen Flows erörtert, und zwar die Tasks und die Aktivitäten.

2.1 Tasks und Aktivitäten

Da Menschen zur gleichen Zeit an vielen Business-Prozessen beteiligt sind, wird eine angemessene Unterstützung der zugrundeliegenden IT-Infrastruktur zur Organisation der Arbeit, die sie durchführen müssen, erforderlich [3].
Person-Centric Flows stellen Technologien bereit, um die Beziehung zwischen Menschen und ihren Flows auszuwerten.
Menschen planen ihre Tasks in einer Art idealer Reihenfolge auf der Basis von externen Einflüssen (z.B. Kontext, Situation) oder internen Einflüssen (z.B. mentaler Kontext).

„Eine Human Task-Komponente implementiert einen Task, der von einer Person ausgeführt wird. Sie stellt die Beteiligung einer Person an einem Geschäftsprozess dar.“ Ein Human Task ist eine Aufgabe, die Menschen zugeteilt wird [19], [33].
In dieser Arbeit wird der Begriff Task auch als Synonym für menschliche Aktivität verwendet.
Ein Geschäftsprozess besteht aus vielen manuellen Aktivitäten (Tasks), z.B. im Bereich des Gesundheitswesens oder der Logistik. Eine Person muss aus einer Menge von Tasks auswählen, welchen Task sie als nächsten ausführen soll.

Ein Workflow enthält verschiedene Aktivitäten, die ausgeführt werden sollen, um ein bestimmtes Geschäftsziel zu erreichen. Eine Aktivität ist eine grundlegende Arbeitseinheit, die von einem Menschen oder automatisch von einem Computerprogramm

ausgeführt werden soll. Human Workflows enthalten Aktivitäten, die Human Tasks genannt werden (kurz Tasks) [18].

Aktivitäten sind das zentrale Element des Modells. Jede Aktivität kann eine beliebige Anzahl an Ressourcen in ihrem Kontext haben (z.B. für die Bereitstellung der Verbindungen für die Dokumente, die einen Task betreffen). Eine Aktivität kann auch Unteraktivitäten (für funktionale Zerlegung) und Unterabhängigkeiten haben. Abhängigkeiten stellen Constraints zwischen Aktivitäten dar. Die Abhängigkeit muss von einer Aktivität koordiniert werden, um das Constraint zuzusichern, das durch die Abhängigkeit repräsentiert wird. Letztendlich können Aktiväten, Ressourcen und Abhängigkeiten jeweils eine beliebige Anzahl Constraints haben, die auf ihren Attributen und Bestandteilen definiert sind. Abbildung 2.1 veranschaulicht diese Relationen [14].

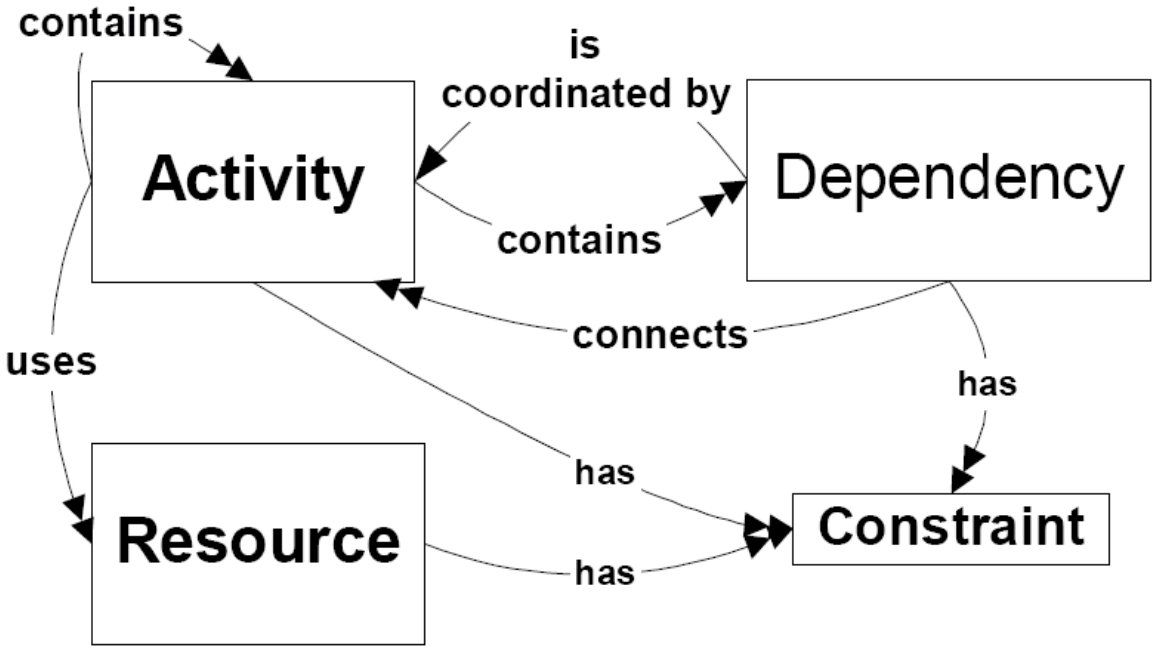

Abbildung 2.1: Prozessmodellbestandteile

Menschliche Interaktionsaktivitäten sind Aktivitäten, die eine Interaktion mit einem Menschen erfordern, z.B. Anzeigen oder Abrufen von Information durch ein Gerät [11].

2.2 Definition von Person-Centric Flow

Ein Person-Centric Flow definiert eine partielle Ordnung auf einer Menge von Tasks, welche von einer einzelnen Person mit einer bestimmten Wahrscheinlichkeit ausgeführt werden sollen. Tasks eines Person-Centric Flows können in drei Zeitformen klassifiziert werden: vergangene, gegenwärtige und zukünftige Tasks.
Vergangene Tasks sind entweder korrekt oder unkorrekt abgeschlossen und ihre Reihenfolge ist bekannt. Gegenwärtige Tasks sind Tasks, die aktuell auf der Worklist einer Person vorhanden sind. Ihre Reihenfolge ist von der Person geplant, aber sie kann sich dynamisch ändern. Zukünftige Tasks sind Tasks, von denen angenommen

wird, dass sie in der Zukunft ausgeführt werden. Sowohl die Menge der künftigen Tasks als auch ihre Reihenfolge können sich dynamisch ändern.
Die Verfügbarkeit von Informationen über die Zukunft ist offensichtlich ein großer Vorteil für die Anpassungsfähigkeit, da sie Proaktivität gewährleistet.
Abbildung 2.2 zeigt eine einfache graphische Darstellung eines Person-Centric Flows.

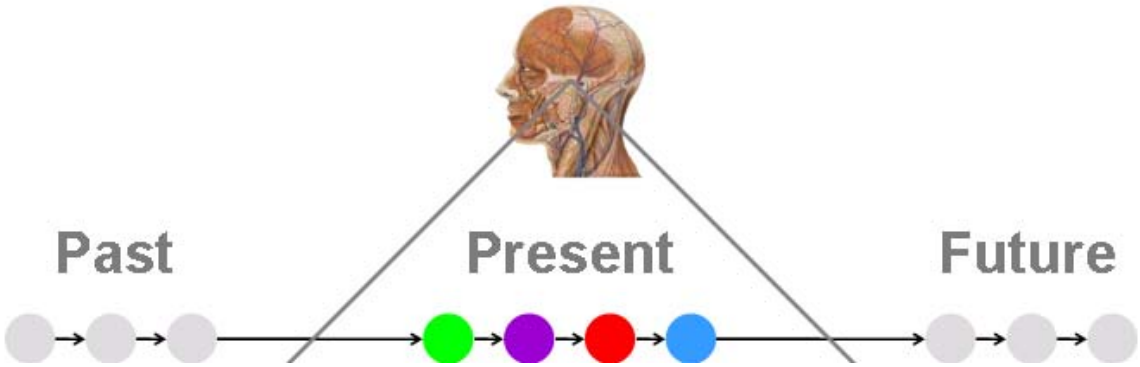

Abbildung 2.2: Person-Centric Flow

Gegenwärtige und künftige Tasks können innerhalb eines Person-Centric Flows vermischt werden. Nur die derzeitig ausgeführten Tasks müssen in die Menge der gegenwärtigen Tasks eingeschlossen werden. Da ein Person-Centric Flow implizit im Kopf einer Person gebildet wird, muss er durch das WfMS vorausgesagt werden, um die Reihenfolgeinformation anzuwenden. Eine Person kann nicht aufgefordert werden, dem WfMS ihren aktuellen Zustand mitzuteilen. Da Tasks in einer hohen Frequenz erscheinen und verschwinden, wäre dies ein zusätzlicher Stressfaktor. Als Folge können die Voraussagen falsch sein. Das Paradigma von Person-Centric Flow steht darüber hinaus zum Teil im Gegensatz zum existierenden Workflow-Paradigma. Z.B. hat ein Person-Centric Flow kein vorgeschriebenes Flow-Modell, da sich die Menge von Tasks dynamisch ändert und folglich auch die Reihenfolge der Tasks. Da jeder Task einmal ausgeführt wird, sind Kontrollmuster wie Schleifen in Person-Centric Flows nicht notwendig. Es existiert also eine einzelne Flow-Instanz, die genau einer Person zugeordnet ist.
Als Flow-Instanz wird hier eine Ausführung eines Flow-Modells bezeichnet, das alle möglichen Ausführungen eines realen Flows beschreibt. Das Metamodell des Flow-Modells ist die Flow-Modellierungssprache [3], [13].

Ein Flow ist eine höhere Programmiersprache für die Modellierung von Echtzeit-Prozessen und menschlichen Aktivitäten. Er besteht aus einer Menge von Tätigkeiten, zusammengefügt durch ein Modell (eine Menge von kontextabhängigen Transitionen), welches definiert, wie Tätigkeiten ausgeführt werden sollen, um spezifische Ziele unter einer Menge von Einschränkungen (Constraints) zu erreichen. Tätigkeiten modellieren Aktivitäten von Menschen oder digitale Prozesse.
Abbildung 2.3 veranschaulicht ein Beispiel für Flows für ein Pflegeheim-Szenario. Hier kann man Flows identifizieren, die die täglichen Runden einer Krankenschwester in einem Pflegeheim, den täglichen Pflege-Zeitplan eines Patienten und die Arbeitsabläufe oder Bedienungsanweisungen von verschiedenen Teilen medizinischer Geräte modellieren [1].

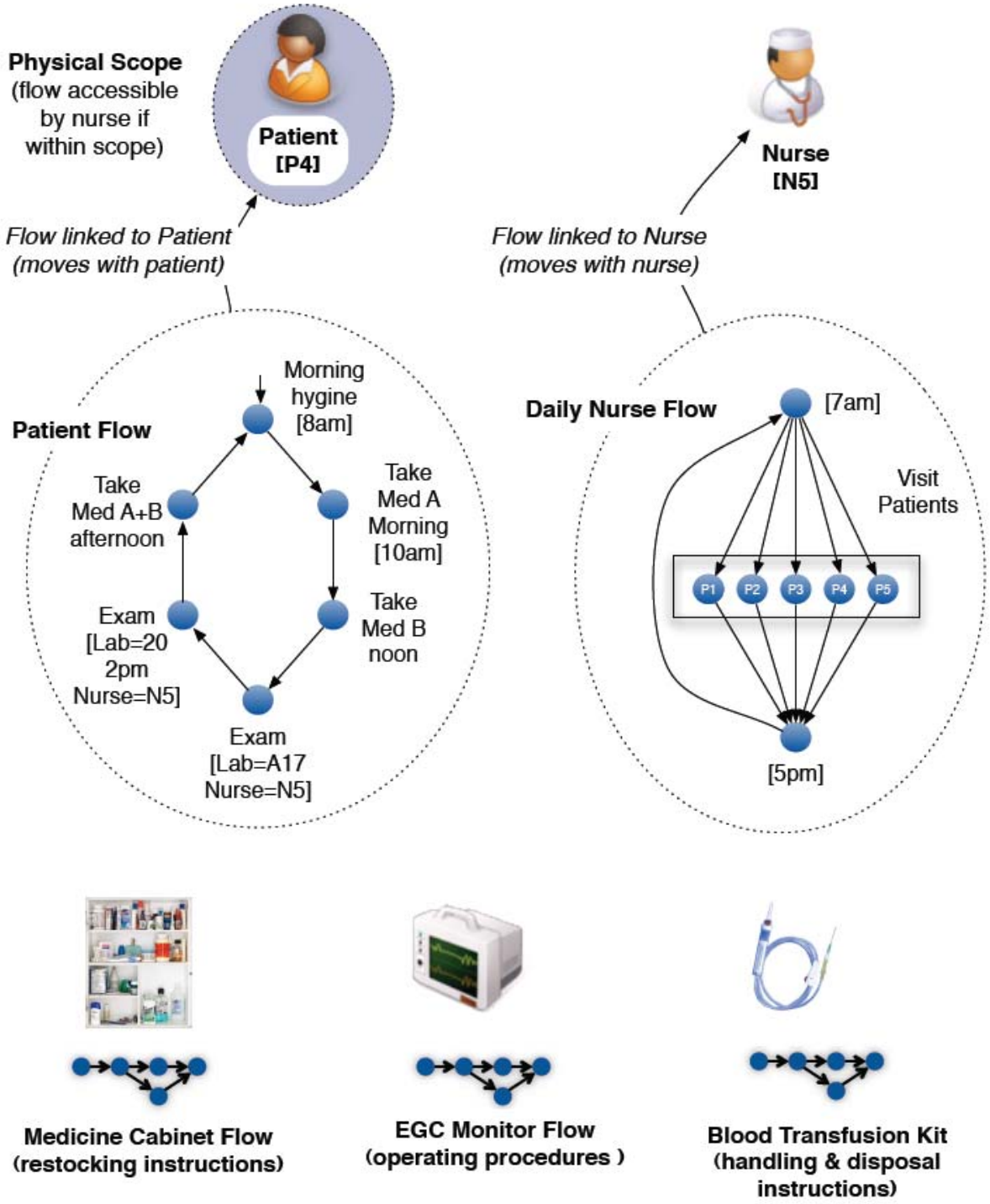

Abbildung 2.3: Angesiedelte Flows in einem Pflegeheim-Szenario, Patienten-Flow, Krankenschwester-Flow und drei Geräte-Flows

2.3 Task Manager

Es besteht ein steigender Bedarf an Computertools zur Unterstützung des verteilten Task Managements, um einen besseren Überblick bereitzustellen und teuere Ineffizienz, Fehler, und Verzögerung zu vermeiden. Die Koordinationsmodelle und Systeme, die in den letzten Jahren entwickelt wurden, leiden an der Steifigkeit von vordefinierten Prozeduren und eingeführten Strukturen, was zu einem begrenzten Anwendungsbereich und nicht adäquater Ausnahmebehandlung in vielen Situationen führt. Sie leiden auch an fehlender informeller Kommunikation, Informationsaustausch und anderen Formen von Computerunterstützung.
Mit dem in [16] beschriebenen Task Manager stellt man ein einfaches, mächtiges und generisches Tool für die Verwaltung verteilter Arbeit zur Verfügung, das diese Probleme durch seine Flexibilität zur Anpassung an ein breites Spektrum von gemeinschaftlichen Arbeitssituationen, und die Möglichkeiten, die es zur Interoperabilität mit anderer Computerunterstützung bietet, behandelt.
Der Task Manager ist für die Zuordnung der Tasks zu den Benutzern verantwortlich und auch für die Bereitstellung der Menge von Tasks und Constraints an den Client. Gemeinsame Tasks können gemeinsam benutzt und unabhängig von jeder beteiligten Person manipuliert werden. Sie sind als gemeinsam benutzte To-Do-Listen auf der Benutzerschnittstelle dargestellt. Mit Hilfe dieses Tools können die Benutzer kooperative Tasks organisieren (erstellen, verfeinern, und modifizieren), ihre Abläufe überwachen, Dokumente und Dienste gemeinsam benutzen, und Nachrichten während der Task-Erledigung austauschen.
Der Task Manager ist ein Softwaresystem für die Spezifikation und Verwaltung kooperaviter Aktivität. Er verteilt Task-Spezifikationen, beigefügte Dokumente und Nachrichten an die beteiligten Benutzer in einer konsistenten Art und Weise. Der Task Manager ist bestimmt für die Unterstützung des Managements verteilter Arbeit in Zeit und/oder Raum durch

- die Unterstützung von Organisation und Planung gemeinschaftlicher Arbeit (wer tut was, mit wem, bis wann, womit?),
- einen aktuellen Überblick von gemeinschaftlicher Aktivität und Arbeitsfortschritt,
- eine dynamische Modifikation von Arbeitsorganisation während der Ausführung,
- die Verfügbarkeit und den Austausch von Dokumenten und Nachrichten innerhalb Gruppen von Menschen, die sich an der Task-Ausführung beteiligen.

Der Task Manager organisiert verteilte Arbeit in Tasks. Die Tasks beinhalten eine verantwortliche Person, eine Frist, andere Teilnehmer, das notwendige Material für die Task-Erledigung, und eventuell Unter-Tasks. Die primäre Benutzerschnittstelle des Task Managers ist eine hierarchisch strukturierte To-Do-Liste, welche alle Tasks anzeigt, an denen ein Benutzer beteiligt ist und welche direkten Zugang zu der wichtigen Information erlaubt, die an einen Task angefügt ist.

Im Folgenden wird auf die Komponenten des Task Managers eingegangen. Der zentrale Begriff des Task Managers ist der Task. Menschen verwenden gemeinsame Dokumente und/oder Dienste und kommunizieren miteinander durch Senden von Nachrichten, um einen Task auszuführen.

- Tasks
  Die Bedeutung eines Tasks für den Task Manager hat verschiedene Aspekte: man könnte bei einem Task auch an ein Projekt denken, d.h. ein gemeinsames

Ziel einer Menge von Menschen (*ergebnisorientiert*). Ein Task kann in mehrere Unter-Tasks aufgeteilt werden und Abhängigkeiten zwischen ihnen und ihren Dokumenten können definiert werden. Je detailliertere Spezifikationen gegeben sind, desto mehr ähnelt ein Task einer Büroprozedur mit kausalen Abhängigkeiten zwischen Unter-Tasks und Dokumenten eines Tasks (*verfahrensorientiert*). Ein Task kann auch als ein einfacher Ordner mit wenig oder nicht definierter Struktur verwendet werden: in diesem Fall ist ein Task einfach ein gemeinsamer Container von Unter-Tasks, Dokumenten und/oder Diensten, und Nachrichten, die Menschen in einem gemeinsamen Task austauschen (*informations-mitbenutzungs-orientiert*).

- Dokumente/Dienste
  Jeder Task kann Ressourcen haben, die ihm zugeordnet sind. Im Büroalltag gibt es in der Regel Ressourcen aller Art, die notwendig zur Erreichung des Ziels eines Tasks sind. Ressourcen sind „Zeiger“ auf verschiedene Arten von computerbasierten Objekten: die erste Art Ressource, an die man natürlich denken könnte, sind Dokumente, die von den Teilnehmern eines Tasks gemeinsam benutzt werden. Aber es gibt auch andere Ressourcen wie Räume, Finanzen, Maschinen, etc., die entscheidend für die Ausführung eines gemeinsamen Tasks sein können.

- Menschen/Benutzer
  Die wichtigste „Ressource“ sind die Menschen, die an einem gemeinsamen Task beteiligt sind. Man unterscheidet zwischen verschiedenen Ebenen von Teilnahme und Kompetenz. Es gibt eine Menge von Menschen, die an einem Task beteiligt sind, die *Teilnehmer*, die alle dieselben Zugriffsrechte auf die Attribute eines Tasks, seine Dokumente und Dienste und seine Nachrichten haben. Teilnehmer können andere Menschen zur Teilnahme an dem Task einladen, d.h. neue Teilnehmer oder Beobachter zu werden. *Beobachter* sind Menschen, die an dem Abschluss eines Tasks interessiert sind, nur Lesezugriff auf Informationen und das Recht auf Teilnahme an dem mit einem Task verbundenen informellen Informationsaustausch haben.
  Eine der Personen, die an dem Task beteiligt ist, ist „mehr berechtigt“ als die anderen: die *verantwortliche Person* für die Ausführung und das Ergebnis des Tasks. Sie hat ausschließlich Schreibzugriff auf die Task-Attribute, z.B. Zustand, Startdatum und Frist, und nur sie kann die Verantwortlichkeit des Tasks einer anderen Person neu zuordnen.

- Nachrichtenvermittlung
  Teilnehmer und Beobachter können frei informelle elektronische Mailnachrichten im Rahmen des Kontextes eines Tasks austauschen.

- Attribute
  Tasks, Teilnehmer, Dokumente, und Dienste sind durch eine Menge von Attributen detaillierter spezifiziert. Der Titel eines Tasks identifiziert sowohl einen Task zu seinen Teilnehmern, als auch gibt er eine kurze und präzise Beschreibung seines Ziels. Wenn ein Benutzer anfangs einen Task erstellt, ist er automatisch die verantwortliche Person für ihn. Wie schon oben erwähnt, hat diese Person spezielle Rechte; insbesondere kann sie den Zustand eines Tasks setzen. Ein anderes wichtiges Attribut ist die Frist eines Tasks. Das System erinnert den Benutzer an bald ablaufende Fristen, aber erzwingt keine Aktionen. Neben diesen wichtigsten

Attributen gibt es eine Reihe anderer Attribute, die dem Benutzer eine detailliertere Spezifikation erlauben, wie ein Task ausgeführt werden sollte: zeitbezogene Daten, wie Startdatum, Daten, die kausale Abhängigkeiten zwischen Tasks und zwischen Tasks und Dokumenten beschreiben, und persönliche Daten, angehängt an einen Task, wie z.B. Notizen.

- Operationen
  Benutzer können Tasks und Unter-Tasks, Abhängigkeiten zwischen Tasks und zwischen Tasks und Dokumenten erstellen, sie können Attribute setzen und modifizieren, Dokumente und Dienstanfragen hinzufügen, ändern und entfernen. Für einen Task zuständige Personen können Verantwortlichkeit ablehnen und sie anderen Benutzern neu zuordnen. Tasks können kopiert und frei eingefügt oder verschoben werden. All dies kann jederzeit geschehen, wodurch eine dynamische Modifikation der Arbeitssituation zur Laufzeit ermöglicht wird.
  Jeder Benutzer hat sofortigen Zugang zu den gemeinsam benutzten Tasks, an denen er beteiligt ist. Das System garantiert eine konsistente Sicht auf Tasks für jeden Teilnehmer [16].

In dieser Arbeit wird der Human Task Manager (HTM) verwendet, der in [18] entwickelt und implementiert wurde. Im Allgemeinen ermöglicht er den Menschen die Beteiligung an der Interaktion mit einem Prozess durch Verwendung von Human Tasks.

2.4 Charakteristiken und Eigenschaften von Person-Centric Flows

Flows werden parallel zu den Echtzeit-Tätigkeiten, die sie beschreiben, ausgeführt: Wenn eine in einem Flow beschriebene Tätigkeit in der realen Welt (durch eine Person) ausgeführt wird, geht der Flow einen Schritt weiter.
Im Gegensatz zu traditionellen Workflows befinden sich Person-Centric Flows in der Realität und sind kontextorientiert: sie sind mit physikalischen Echtzeit-Entitäten wie Ausrüstung und Menschen verbunden. Person-Centric Flows werden von den Menschen ausgeführt und bewegen sich zusammen mit ihnen durch verschiedene reale Umgebungen. Dabei reagieren sie auf deren Kontext und werden durch ihn beeinflusst [1].
Sie modellieren das Verhalten, das für ihre Entität bestimmt ist und die Bedingungen über den Ausführungskontext, die ein korrektes Verhalten garantieren. Diese Flows sind daher geeignet, die Abweichungen des Verhaltens der Entität, an die sie angefügt sind, sowie Probleme im Ausführungskontext zu überprüfen, und Anpassung auszulösen [11].

Die Ära von "Pervasive Computing" bringt eine große Herausforderung mit sich: Pervasive Anwendungen müssen sich an die Dynamiken, die im menschlichen Verhalten aufkommen und die sich ständig ändernden Umgebungen anpassen.
Flows sind ein neuartiges workflow-basiertes Programmierparadigma, das ideal geeignet für den Entwurf und die Ausführung von menschenorientierten, anpassungsfähigen pervasiven Anwendungen ist. Dabei sind dynamische, in der realen Welt angesiedelte Workflows in der Lage, ihre Ausführung zur Anpassung an die Änderungen in ihrer Ausführungsumgebung zu modifizieren.

Flows sind dynamische Modelle, die eine Menge von auszuführenden Aktivitäten und ihre Beziehungen zueinander definieren. Sie sind in vielen Bereichen angesiedelt, einschließlich Geschäftsprozessen und werden als eine Erweiterung von traditionellen Workflow-Konzepten vorgeschlagen, um sie flexibler in Bezug auf ihre pervasive Ausführungsumgebung zu machen.
Obwohl Person-Centric Flows eng mit klassischen Workflows verbunden sind, ist ihr beträchtlicher Vorteil hierbei, dass Anwendungen sowie ihre Umgebung proaktiv, basierend auf dem Wissen über zukünftige Tasks, angepasst werden können. Die Flows sind in der Lage, sich und die Benutzerumgebung einer Person an ihre Ziele und Aktivitäten anzupassen.
Wenn ein mobiler Benutzer einen Flow ausführt, der seine zukünftige Tätigkeiten spezifiziert, wird die pervasive Workflow-Engine in seiner Umgebung für ihn durch den Flow festgelegt. Da sich die Menschen anders entschließen können, und da Artefakte und Menschen Änderungen in der Umgebung unterliegen, kann sich der Flow selbst anpassen, um solche Änderungen widerzuspiegeln. Dies erfordert, dass Flows kontextorientiert sind: während der Ausführung muss es möglich sein, Informationen über die zugrundeliegende Umgebung (z.B. Status von anderen Flows, menschliche Aktivitäten) zu erhalten. Sie können den Kontext in Bezug auf die aktuelle Umgebung ihrer Entitäten sowie die derzeitigen Aktivitäten der Entitäten in Betracht ziehen, um sich dynamisch an sich ändernde Situationen anzupassen.
Andererseits müssen Flow-Modelle flexibel genug sein, um eine einfache und kontinuierliche Anpassung zu ermöglichen. Ein besonderes Merkmal von Flows ist, dass sie sich in der realen Welt befinden. Dies führt zur Verbreitung der Flows und wird auf zwei Wegen erreicht. Der erste Weg ist, dass die Flows logisch an reale Entitäten (welche entweder Objekte oder Menschen sein können) angefügt werden und dass sie sich mit ihnen durch verschiedene Kontexte bewegen. Der zweite Weg ist, dass sie auf physischen Geräten (z.B. PDAs, Desktops) laufen. Ein anderer wichtiger Aspekt von Flows ist, wie bereits erwähnt, ihre Anpassungsfähigkeit.

Das System, das für die Ausführung von Flows verantwortlich ist, wird einfach Flow-System genannt. Dieses System ist in der Regel verteilt und befindet sich in der Umgebung der Entität. Es ist zuständig dafür, dass ein Flow gleichzeitig mit den realen Tätigkeiten seiner Entität ausgeführt wird. Die grundlegende Idee des Flow-Ansatzes ist, dass diese synchronisierte Ausführung dem Flow-System erlaubt, Abweichungen der realen von den geplanten Tätigkeiten zu erkennen und diese Abweichungen als die Grundlage für adäquate Anpassungen zu verwenden. Um dies zu erreichen, entwickelt man Aktivitäts-Sensortechnologien (s. Kapitel 3.3.1) zur Erkennung von Benutzeraktivitäten und anderen Quellen von Kontextinformation (z.B. Position). Basierend auf der Positionsinformation, bewegt sich der Flow durch seine Tasks, und jeder Task kann unterstützende Dienste in der lokalen Umgebung aufrufen, um den Benutzer durch den gesamten Prozess anzuleiten (s. Abbildung 2.4). Ein sehr wichtiges Element des Flows ist die Menge der Metadaten, die die Ziele und Einschränkungen des Flows ausdrücken.

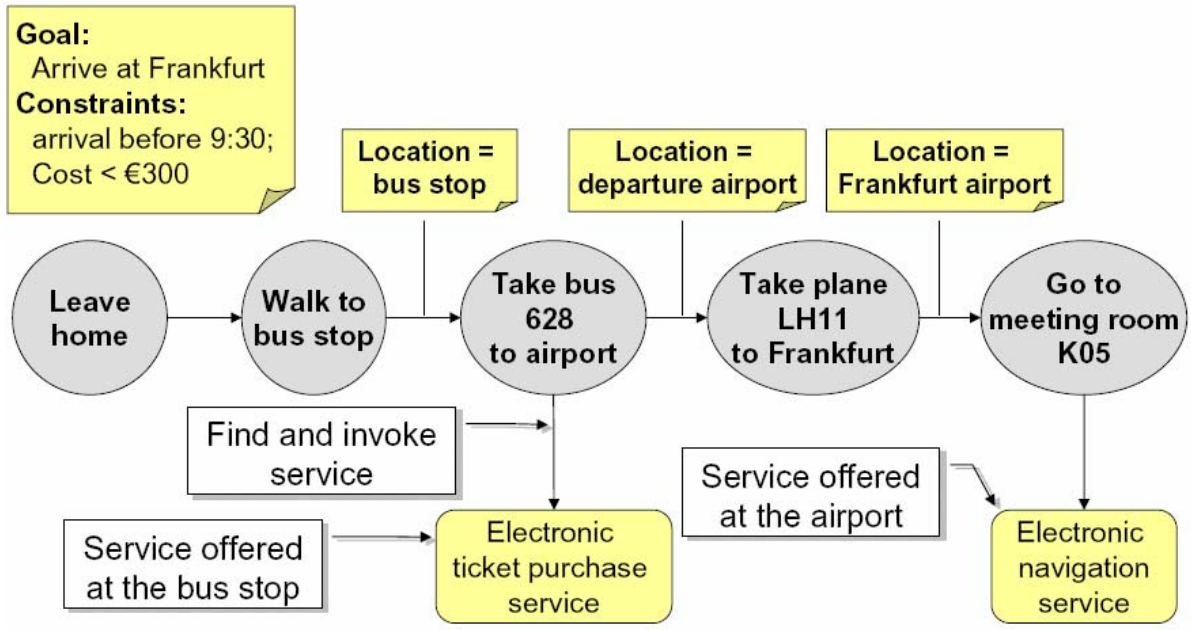

Abbildung 2.4: Beispiel-Flow für eine Reise eines Geschäftsreisenden

Eine entscheidende Innovation der Flow-Technologie ist, dass ein Flow Information über vergangene, gegenwärtige und (potentiell) zukünftige Aktivitäten seiner Entität darstellt [12], [13].

Der wesentliche Vorteil der flow-basierten Anwendungen ist, dass das Modell, welches den Kontrollfluss festlegt, sowie die Ziele und Einschränkungen explizit definiert sind. Da das Flow-Konzept auf formalen Modellen basiert, ist es möglich, das aktuelle und auch das zukünftige Verhalten von Anwendungen und/oder beteiligten Nutzern in Flows zu bestimmen. Diese Information ist sehr wertvoll, um verschiedene Formen der Anpassung und Weiterentwicklung zu ermöglichen [17].

Der Kontrollfluss von Business Workflows ist durch die strikte Ausführungsreihenfolge von Aktivitäten charakterisiert, die schon zur Entwurfszeit definiert wird, wie z.B. bei vollautomatisch auszuführenden Workflows. Dies ist jedoch für Person-Centric Flows nicht immer geeignet. Person-Centric Flows sollen viel flexibler sein. In Szenarien, wo die Echtzeit-Prozesse nur semi-strukturiert sind, sollen sie den Menschen mehr Entscheidungsfreiheit geben, in welcher Reihenfolge sie bestimmte Aktivitäten ausführen möchten. Demzufolge wird hier ein Ansatz zum Entwurf von Person-Centric Flows durch Einschränkungen (Constraints) nahegelegt, um sie flexibler zu gestalten.
Dieser höhere Grad an Flexibilität kann durch die Verwendung von deklarativen Workflows erreicht werden (auch constraint-basierte Workflows genannt). Diese Workflows verfolgen den Ansatz, dass jede Ausführungsreihenfolge der Aktivitäten möglich ist, solange sie nicht ausdrücklich durch eine Einschränkung verboten wird.
Den imperativen Sprachen mangelt es an Flexibilität, da sie keine Abweichungen von der Ausführungsreihenfolge von Aktivitäten, festgelegt im Prozessmodell, erlauben. Dies ist nicht geeignet für Human Workflows, da Menschen mehr Entscheidungsfreiheit haben sollten, welche Aktivitäten oder Tasks sie ausführen wollen.
Ein Ansatz, die Steifigkeit von heutigen imperativen Workflows zu überwinden, sind deklarative Workflows. In diesem Modellierungsparadigma ist das „Was“ angegeben,

und nicht das „Wie“. Genauer gesagt, sind deklarative Workflows durch Verwendung von Einschränkungen spezifiziert bzw. modelliert, die festlegen, „was“ getan werden muss oder welche Einschränkungen bei der Ausführung des Prozesses erfüllt sein müssen. Diese Einschränkungen nähern sich dem gewünschten Verhalten des Prozesses und stellen somit bestimmte Business-Anforderungen dar. Jede Ausführungsreihenfolge von Aktivitäten ist erlaubt, solange sie nicht eine der Einschränkungen verletzt. Mit anderen Worten, hängt die Art und Weise, wie die Aktivitäten ausgeführt werden, von den Präferenzen der Benutzer ab.
Andererseits definieren imperative Workflows explizit, „wie“ der Geschäftserfolg erreicht werden muss, indem sie einen Arbeitsablauf vorschreiben. Imperative Sprachen neigen dazu, sehr einschränkend bzgl. der ursprünglichen Anforderungen zu sein. Dies führt dazu, dass Menschen weniger Entscheidungsfreiheit haben, wenn sie an einem Prozess arbeiten. Die Tatsache, dass deklarative Workflows mehr Entscheidungsfreiheit geben, in welcher Reihenfolge die Aktivitäten ausgeführt werden, macht dieses Paradigma geeigneter für Human Workflows als imperative Workflows [2], [18].

Da die kognitiven Fähigkeiten der Menschen die gleichzeitige Ausführung von vielen Tasks nicht erlauben, werden die verschiedenen Tasks (Aktivitäten) durch die Menschen in einer für sie bequemen Art und Weise angeordnet. In einer sog. Worklist wird der aktuelle Zustand der Gestaltung im Workflow Management abgebildet. Die Benutzer können die Worklists in einer gewünschten Art unter Verwendung der Eigenschaften sortieren, die mit einem Task verbunden sind. Die Effizienz von Benutzern hängt stark von der Möglichkeit zur individuellen Anordnung der Worklist ab. Studien haben gezeigt, dass Menschen nicht nur die grundsätzliche Anordnung der Worklists zur Auswahl der nächsten Tasks verwenden, sondern auch Zusammenhänge oder ähnliche Kontextinformation der verschiedenen Tasks zu deren Auswahl verwenden. Dies ähnelt dem traditionellen Workflow-Paradigma, bei dem die Ausführung der verschiedenen Aktivitäten durch den Kontext angetrieben wird, der mit einer Prozessinstanz verbunden ist.
Der Person-Centric Flow wird von einer einzelnen Person geplant und ausgeführt. Grundsätzlich besteht ein Person-Centric Flow aus den Tasks einer Worklist einer Person, erweitert durch zusätzliche Anordnungsinformation.
Alle Tasks werden durch das Workflow Management System generiert, das auch die Worklist der Person bereitstellt.
Zur Nutzung von Person-Centric Flows muss eine explizite Darstellung von ihnen erstellt werden. Die explizit sichtbaren und beobachtbaren Person-Centric Flows eröffnen große Möglichkeiten zur Unterstützung der Menschen bei der Ausführung ihrer Arbeit.
Das Ziel von Person-Centric Flows ist es, die Menschen zu unterstützen, und nicht die Ausführungsreihenfolge ihrer Tasks vorzuschreiben. So wird z.B. ihre Aufmerksamkeit auf mögliche Fehler oder auf die Tatsache gelenkt, dass die gewählte Reihenfolge unzulässig ist. Ein anderer Vorteil von Person-Centric Flows ist die Möglichkeit, die Umgebung an die Bedürfnisse eines Zukunft-Tasks anzupassen, was besonders nützlich im Fall von Tätigkeiten ist, die eine zeitaufwendige Vorbereitung benötigen. Z.B. das Füllen der Badewanne kann rechtzeitig gestartet werden, sobald bekannt ist, wann eine Krankenschwester einen Patienten waschen wird [3].

Zusammenfassend kann man Person-Centric Flows als Workflows mit den folgenden Charakteristiken bezeichnen:

- angesiedelt in der realen Welt: sie können logisch oder physikalisch an Entitäten wie Artefakte oder Menschen angefügt werden, und bewegen sich mit ihnen durch verschiedene Kontexte.

- kontextorientiert: sie können den Kontext in Bezug auf die aktuelle Umgebung sowie die eigentlichen Aktivitäten ihrer Entitäten erkennen.

- Während sie ausgeführt werden, modellieren sie das vorgesehene Verhalten für ihre Entitäten und passen die Umgebung der Entitäten an dieses Verhalten an.

- Der Flow passt sich selbst an, um die Änderungen in der Umgebung widerzuspiegeln.

3 Einsatz von Person-Centric Flows in der Praxis

Das Konzept von Person-Centric Flows findet praktischen Einsatz in der Realität, wodurch eine erhebliche Entlastung des Personals erzielt wird. Es werden im Folgenden zwei Beispiele dafür beschrieben: Einsatz in Krankenhäusern und Einsatz in Unternehmen.

3.1 Einsatz von Person-Centric Flows in Krankenhäusern

Die medizinische Arbeit unterscheidet sich grundsätzlich von typischen Bürotätigkeiten durch ihre herausfordernden bzw. anspruchsvollen Eigenschaften: extreme Mobilität, Ad-hoc Zusammenarbeit, Unterbrechungen und ein hohes Maß an Kommunikation. Dies macht den Gesundheitsbereich zu einem interessanten Einsatzbereich für das Design der „Pervasive Computing“-Technologie [38].
Krankenhäuser werden in ansteigendem Maße infolge des wachsenden Kosten- und Wettbewerbsdrucks gezwungen, Kosten zu senken, ohne dabei die Qualität der Patientenversorgung einzuschränken. Ein wichtiger Ansatz ist hier die Neuregelung der medizinischen, pflegerischen und administrativen Leistungsprozesse unter Wirtschaftlichkeits- und Qualitätskriterien. Die Überprüfung der Aufbau- und Ablauforganisation bildet eine Basis für das Aufdecken von Schwachstellen und damit für die Neugestaltung der Abläufe. Der Erfolg dieser Maßnahmen wird doch nur dann nachhaltig sein können, wenn die optimierten Prozesse und Strukturen durch geeignete Krankenhausinformationssysteme intelligent und flexibel unterstützt werden können. Es ist wichtig, dass sich diese Systeme schnell und kostengünstig an sich ändernde Strukturen und Prozesse anpassen lassen. Heutige Anwendungssysteme, die in konventioneller Implementierungstechnik realisiert sind und eine „hart-verdrahtete“ Ablauflogik im Programm haben, lassen dies normalerweise kaum zu. Einen vielversprechenden Ansatz bieten Workflow Management Systeme, indem sie den Kontroll- und Datenfluss wie auch organisatorische Aspekte eines Ablaufs vom eigentlichen Anwendungscode trennen. Die Ausführungslogik ist nicht mehr wie bisher im Programmcode versteckt, sondern für entsprechende Steuerungs- und Überwachungskomponenten erreichbar. Es lassen sich so Anwendungssysteme realisieren, die – wenigstens im Prinzip – sehr viel einfacher als bisher an geänderte Geschäftsprozesse angepasst werden können.
WfMSe können aus Anwendersicht bei der Verkürzung der Durchlaufzeiten und der Verbesserung der Prozessqualität helfen, indem sie Aufträge und Informationen an die zuständigen Stellen automatisch weiterleiten, die (Rück-)Verfolgung offener und abgeschlossener Vorgänge möglich machen, den Benutzer an Terminüberschreitun-

gen erinnern und am Arbeitsplatz die notwendigen Programme und Kontextinformationen für die Ausführung von Prozessschritten zur Verfügung stellen.
Das klinische Personal ist durch die Planung und Abstimmung von Untersuchungsterminen zeitlich stark belastet und deswegen soll es von organisatorischen, nichtwertschöpfenden Aufgaben entlastet werden.
Funktionen zur Steuerung und Überwachung von Prozessen und zur Verwaltung von Arbeitslisten werden für den Anwender angeboten. Es ist in gewissem Umfang möglich, zur Laufzeit Abweichungen vom Standardablauf vorzunehmen, indem z.B. Aktivitäten übersprungen, wiederholt oder hinzugefügt werden [10].

Die Arbeit von Ärzten, Krankenschwestern, und Technikern in Krankenhäusern ist erheblich durch organisatorische Tasks belastet. Medizinische Prozeduren (z.B. Labortests) müssen geplant und vorbereitet werden, Termine mit verschiedenen Dienstleistern müssen geplant, Proben oder die Patienten selbst transportiert, Besuche von Ärzten aus anderen Abteilungen festgelegt und Berichte geschrieben, übermittelt und ausgewertet werden. Daher ist die Kooperation zwischen Menschen aus verschiedenen Abteilungen ein wichtiger Task mit wiederholendem, aber nicht-trivialem Charakter. In der Regel müssen organisatorische Tasks manuell vom notwendigen klinischen Personal koordiniert werden, z.B. zeitaufwendige Telefonate und Dokumentationsschritte. In der klinischen Praxis führt dies oft zu Organisationsproblemen und zu hoch administrativer Belastung von Krankenhausärzten. Als Konsequenz ergeben sich viele Fehler und unerwünschte Wirkungen treten auf. Patienten müssen warten, da Ressourcen (z.B. Ärzte, Räume, technische Ausstattung) nicht verfügbar sind (z.B. aufgrund schlechter Planung). Medizinische Prozeduren können nicht ausgeführt werden, wenn die Information fehlt, Vorbereitungen werden ausgelassen, oder eine Vorbereitungsprozedur wurde verschoben, abgesagt oder erfordert Wartezeit. Nachfolgende Verabredungen müssten dann neu geplant werden, was wieder zu zahlreichen Telefonaten oder Zeitverlusten führt. Aus all diesen Gründen sind Krankenhausaufenthalte oft länger als erforderlich, und die Kosten oder sogar die Invasivität ärzlicher Behandlung sind unnötig hoch. Das klinische Personal ist sich diesen Problemen bewusst und prozessbezogene Informationssysteme, die organisatorische Tasks koordinieren und Informationen patientennah bereitstellen, würden hoch willkommen sein. Zunehmend wurde verstanden, dass die Beziehung zwischen Medizin, Organisation und Information stark ist, und dass heutige Organisationsstrukturen und IT-Systeme keine optimale Unterstützung bieten [6].

Krankenhäuser sind komplexe, aber dennoch höchst strukturierte Umgebungen, in denen praktisch alle Aktivitäten in genau definierten Workflows durchgeführt werden. Ein zentrales Problem in vielen Krankenhaus-Workflows liegt in der Tatsache, dass eine komplexe Interaktion mit einem Computersystem bei Aktivitäten wie der Untersuchung eines Patienten oder der Ausführung eines chirurgischen Eingriffs nicht möglich ist. Als Folge ist Information oft vor einer Prozedur ausgedruckt, zuerst in Diktiergeräten aufgenommen, oder von den begleitenden Krankenschwestern geschrieben, bevor sie in das Krankenhausinformationssystem eingegeben wird. Zusätzliches Personal wird für die Bedienung von komplexen Geräten benötigt, da an der Prozedur beteiligte Ärzte und Krankenschwestern es nicht selbst tun können.
In vielen Situationen können die oben beschriebenen Probleme durch Verwendung von Person-Centric Flows gelöst werden. Die Idee ist es, die elektronischen Workflows mit den Prozeduren, die durch das Personal ausgeführt werden, durch Verwendung von Kontext- und Aktivitäts-Erkennung („Activity Recognition“) zu verbinden. Das System wird dann in der Lage sein, automatisch die entsprechenden Eingabe-

masken zu initialisieren. Wenn die Untersuchung weitergeht, kann erkannt werden, welcher Schritt der Untersuchung wann ausgeführt worden ist. Das System befüllt automatisch die entsprechenden Felder mit den Daten aus den Geräten (z.B. EKG, Puls oder Oximeter). Alternativ können die Anweisungen des Arztes in die entsprechenden Felder übertragen werden. Die Kenntnis des genauen Schrittes, der zusammen mit den Daten von Geräten ausgeführt wurde, kann zur Einschränkung des Suchraums für die Spracherkennung weit genug verwendet werden, um ein ausreichendes Maß an Zuverlässigkeit zu gewährleisten. Multimedia-Daten wie Fotos und Audio-Dateien von den Bemerkungen des Arztes können auch an die entsprechenden Felder des Formulars angehängt werden. Schließlich können entsprechende Geräte ohne den Bedarf zusätzlichen Bedienpersonals aktiviert und initialisiert werden, da die nächsten Schritte im Flow vom System vorhersagbar sind.

Behandlungsmethoden im Krankenhaus sind strukturierte Worklows, die komplexe Interaktionen und Informationsaustausch zwischen Patient, medizinischem Fachpersonal und heterogenen medizinischen Geräten erfordern. Die Verwaltung von diesen administrativen und organisatorischen Tasks steht im Kontrast mit der Bewältigung der medizinischen Behandlungen.

Die Lösung dafür sind die sogenannten Person-Centric Flows. Im Grunde genommen stellen sie eine Methodik für die Entwicklung von pervasiven Anwendungen dar, die in der Lage sind, sich und die menschliche Benutzerumgebung an die Ziele und Aktivitäten der Benutzer anzupassen.

Flows, die spezifische medizinische Behandlungen modellieren, sind mit medizinischen Instrumenten und Geräten, Krankenhauspersonal und dem Patienten mittels Kontext- und Aktivitäts-Erkennung verbunden. Behandlungsunterlagen von Patienten können leicht zwischen verschiedenen medizinischen Fachpersonal ausgetauscht werden, Untersuchungsergebnisse von Laboratorien sind sofort zugänglich, Verschreibungen werden direkt an Apotheker ausgeliefert [17].

Die Unterstützung der Mobilität ist zental für die Tätigkeiten in einem Krankenhaus. „Mobile Computing“ gilt im Allgemeinen als eine Unterstützung für tragbare Computer wie Mobiltelefone, PDAs und Tablet PCs.

Das Konzept von Activity-Based Computing (ABC) zeigt ein konkretes Beispiel für die Unterstützung der klinischen Arbeit und den Einsatz in Krankenhäusern in Dänemark. Das Ziel dabei ist, die Benutzer bei der Verwaltung der komplexen Menge von Tätigkeiten, Tools, Sachmitteln, Ressourcen und an einer „Activity“ beteiligten Menschen durch die Einführung einer expliziten Darstellung der Aktivität in das Computersystem zu unterstützen. Die „Activity“ ist eine computerbasierte Repräsentation einer realen menschlichen Aktivität. Sie spiegelt die menschliche Aktivität wider und bietet Zugriff auf die Ressourcen, die wichtig für ihre Ausführung sind. Die „Activity“ ist also eine Aggregation von Diensten, Ressourcen, Artefakten und Benutzern, die für eine reale menschliche Aktivität relevant sind.
Abbildung 3.1 veranschaulicht eine „Activity“, die Dienste, Ressourcen, Dokumente und Benutzer aggregiert und verbindet, die für die reale Aktivität der Behandlung von Leukämie von einem Patienten wichtig sind. Unter anderem ermöglicht sie Zugriff auf die Behandlungsunterlagen, die Information über die verabreichten Medikamente und die medizinischen Bilder. Damit sind alle Ressourcen und Materialien in Bezug auf die Behandlung eines bestimmten Patienten einschließlich digitaler Ressourcen

wie Krankenakten, Medizindiagrammen, Laborergebnissen und Röntgenbildern logisch miteinander verbunden. Der Zugang auf diese Materialien ist durch die entsprechenden beteiligten Computersysteme vermittelt: das System für elektronische Patientenakten (electronic patient record system, EPR), das elektronische Medikamenten-System (electronic medication system, EMS), und das Bilddatenarchivierungs- und -kommunikationssystem (picture, archiving, and communication system, PACS) [39].

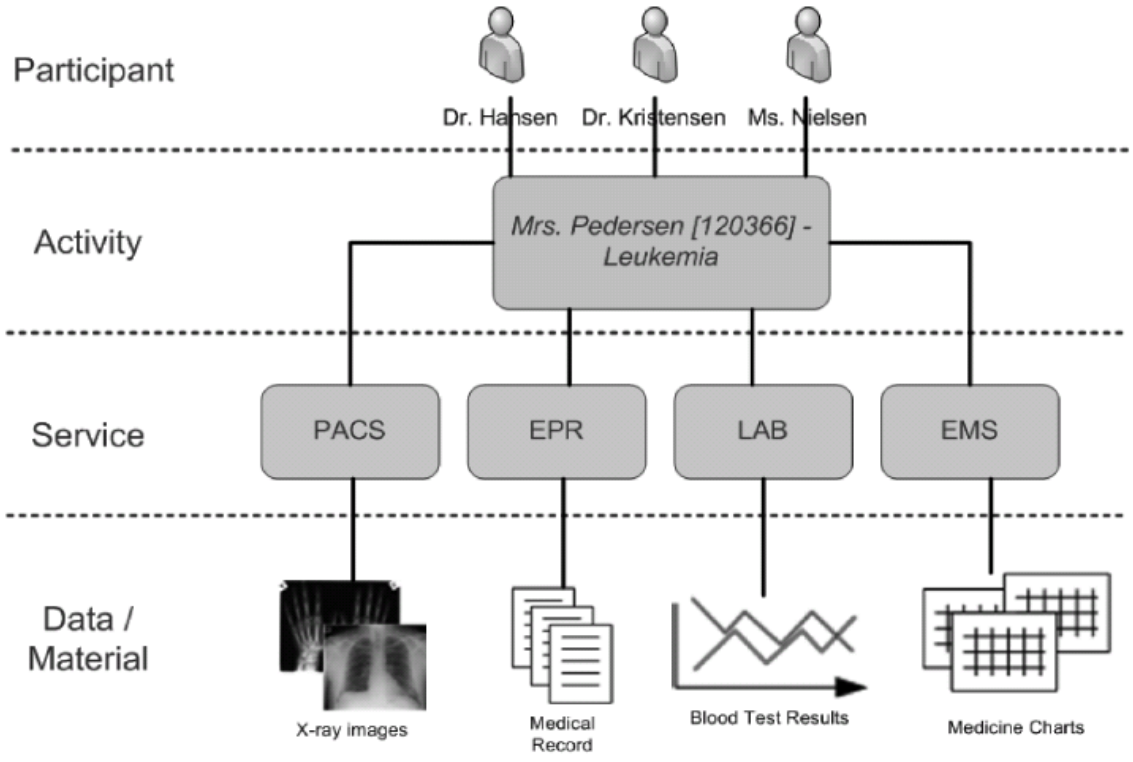

Abbildung 3.1: „Activity“, die eine Menge von Diensten, Datenressourcen und Benutzern aggregiert

3.2 Einsatz von Person-Centric Flows in Unternehmen

Ein konkretes Beispiel in einem deutschen Unternehmen, das das Personal durch den Einsatz eines dem Person-Centric Flow ähnlichen Systems von Organisationsaufgaben entlastet, beweist die Nützlichkeit eines solchen Konzepts. Eine Unternehmerin entwickelte ein System, das jeden Arbeitsschritt erfasst, überwacht und berechnet. Das Ziel war dabei, nicht ihre Mitarbeiter zu kontrollieren, sondern pünktlich mit der Arbeit fertig zu werden.
Das System ist ein Programm, das jeden der Arbeitsschritte kennt. Denn jeder Schritt ist im System geplant, jeder Fortschritt wird dokumentiert und schließlich freigegeben. Das System weiß seit dem ersten Projekt alles über jeden, der mit ihm arbeitet. Die Idee hierbei ist: Die Arbeit wird in systematische und in zahllose kleine Schritte gegliedert. Hintergrund dafür ist die Überzeugung, dass wenn jeder Schritt erfasst und geplant ist, auch wirtschaftlich gearbeitet wird und dass wer sich nicht mehr um Organisation und Planung kümmern muss, auch besser arbeitet. Die Folge davon: Leerlauf, Zeitüberschreitung und explodierende Projektkosten werden früh erkannt.

Davon profitieren besonders die Mitarbeiter: es gibt kaum Überstunden, Überforderung kommt praktisch nicht vor, da die Arbeit trotz Planung flexibel bleibt.
Wenn ein Projektleiter von einem Kundenbesuch kommt, wird die neue Aufgabe von ihm in Arbeitsschritte aufgeteilt. Er legt jeden einzelnen Schritt als Auftrag im System an, wofür er einige Felder in der Eingabemaske ausfüllt. Er plant, wann ein Schritt notwendig ist, wie wichtig er ist, wie lange ein Mitarbeiter dafür braucht und ob der Schritt abgerechnet wird. Eine Datenbank arbeitet im Herzen des Systems. Die Datenbankstruktur ist der Bauplan des Systems, ein Plan mit Kästen und Pfeilen, wie der Stadtplan einer Stadt aus Daten, eine Landschaft aus verschachtelten und miteinander verknüpften Tabellen [21].

3.3 Verteilung von Anleitungsinformation

In dieser Arbeit wird die Aufmerksamkeit auf die Nützlichkeit eines Person-Centric Flows im medizinischen Bereich fokussiert. Um die Frage zu beantworten, worauf eine Krankenschwester in so einem Szenario reagieren soll, werden zwei Möglichkeiten betrachtet: das Konzept der „Activity Recognition“ und die direkte Navigation des Benutzers. Im Folgenden wird auf diese beiden Möglichkeiten näher eingegangen.

3.3.1 Activity Recognition

Die erste Möglichkeit für die Darstellung und Verteilung von Anleitungsinformation in einer realen Umgebung über Zeit und Raum unter Betrachtung von Benutzerkontext und Arbeitsprozessen, um die menschliche Performanz zu maximieren, wird „Activity Recognition“ genannt.
Eine effektive Steuerung der Menschen in komplexen und hochdynamischen Arbeitsumgebungen erfordert Fortschritte in deklarativen Aktivitätsmodellen auf hoher Ebene, die den Flow der menschlichen Arbeitsaktivitäten und ihre beabsichtigten Ergebnisse beschreiben können, sowie neuartige Benutzerschnittstellen-Modelle für Verteilung von Anleitungsinformationen über Zeit und Raum. Pervasive Arbeitsunterstützungssysteme nutzen Sensoren, Organizer, sowie tragbare Geräte, um die Arbeitsaktivitäten in Echtzeit zu analysieren und die Benutzer mit relevanten und rechtzeitigen Informationen in Bezug auf ihre Arbeit zu versorgen. Projizierend in die Zukunft kann man sich vorstellen, dass künftige Arbeitsumgebungen dicht instrumentiert und in der Lage sein werden, genaue Details von Arbeitsaktivitäten und Prozessen zu verstehen.
Man verwendet Flows, sowie mobile Projektor-Schnittstellen, um die Darstellung und Verteilung der Anleitungsinformation in pervasiven Arbeitsumgebungen zu steuern: mitgeführte kontextabhängige Projektoren ermöglichen den Menschen, Flows und Task-Information, eingebettet in die umliegende reale Umgebung, zu erkennen. Dies sorgt für eine nahtlose Benutzererfahrung, wo Task-Information virtuell physischen Entitäten, auf die sie sich bezieht, überlagert ist.
Aus menschlicher Sicht definieren Flows Handlungsmöglichkeiten. Ein Flow, eingebettet in ein Objekt oder einen Raum, definiert Tasks und Tätigkeiten, die mit dem Objekt oder in dem Raum ausgeführt werden können oder sollten. Eine Tätigkeit kann eine körperliche Tätigkeit wie Verabreichen von Medikamenten oder eine digitale Tätigkeit wie Einschalten einer Maschine sein. Angenommen, eine Organisation hat Flows definiert und sie an Entitäten, Menschen und Positionen gebunden (dabei ein physikalisches feinverteiltes Business-Prozessmodell erzeugend), dann wird die

zu erforschende Schlüsselfrage lauten: Wie kann eine Person Flows in der umliegenden realen Umgebung erkennen und sich nach ihnen richten? Alternativ formuliert klingt die Frage folgendermaßen: Wie können Flows verwendet werden, um die Tätigkeiten der Menschen effektiv zu steuern und ihnen beim Erreichen von berufsbedingten Zielen zu helfen?
Die Lösung dieser Frage besteht aus zwei Komponenten: mobile Anzeigeschnittstellen zur Anzeige der Task-Information, eingebettet in die realen Umgebungen (s. Abbildung 3.2), und Ansteuerungsstrategien zur Entscheidung, welche Information zugänglich sein soll und wann, wo und wie sie in der Umgebung dargestellt werden soll (s. Abbildung 3.3) [1].

Abbildung 3.2: Links: Anleitungsschnittstelle, projektiert auf das Medikamentenfach. Rechts: Gürtelmontierter mobiler Projektor mit Wand-projektierter Schnittstelle

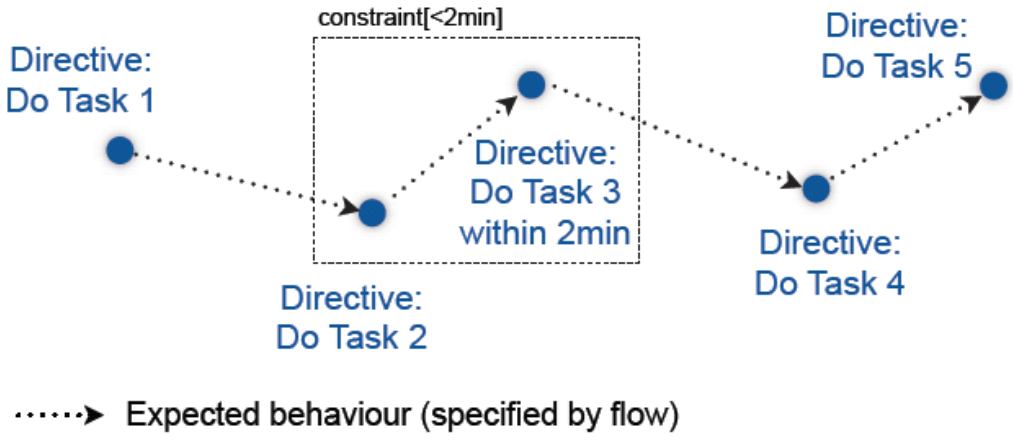

Abbildung 3.3: Ansteuerungsstrategien in der Umgebung

3.3.2 Direkte Navigation

Für die Interaktion des Benutzers und seines Flows sind adäquate Benutzerschnittstellen notwendig. Eine Möglichkeit ist die Interaktion mit einem Benutzer durch sein mobiles Gerät, wenn der Benutzer tatsächlich so ein Gerät trägt.
Die zweite Möglichkeit besteht also darin, dass der Benutzer über Eingabefelder (Schalter, Tastatur, berührungssensitiver Bildschirm) in den Prozess eingreifen kann, um Zustände bzw. Sollwerte einzeln zu verändern oder zu verwalten [13].
In dieser Arbeit wird diese Möglichkeit eingesetzt. Man nimmt hier also an, dass der Benutzer direkt auf ein Ereignis reagiert und direkt auf dem Objekt navigiert, indem er direkt den Touchscreen berührt.
Eine andere Möglichkeit wäre die Darstellung einer Liste, wobei sie über eine Multitouch-Scrollbar direkt angeklickt werden kann.

4 Visualisierung

Bildhafte Darstellungen sind ein weit verbreitetes Mittel zur Präsentation von Fakten und Informationen. Daten, Strukturen und Zusammenhänge werden graphisch dargestellt, damit eine effizientere Analyse und Kommunikation erreicht wird. Als Visualisierung wird der Prozess der Erzeugung solcher Darstellungen bezeichnet. Grundlage einer jeden Visualisierung sind die Informationen, die dargestellt werden sollen.
Das Ziel der Visualisierung ist die Erzeugung geeigneter visueller Repräsentationen einer gegebenen Datenmenge, um damit eine effektive Auswertung zu ermöglichen. Sie hat zwei Aufgaben: Sie soll einerseits Ergebnisse präsentieren und damit das Verständnis und die Kommunikation über die Daten und die zugrunde liegenden Modelle und Konzepte erleichtern. Andererseits soll sie die Analyse der Daten unterstützen, indem die Bilder so aufgebaut werden, dass der Betrachter nicht nur sehen, sondern auch erkennen, verstehen und bewerten kann. Innere Zusammenhänge, die sonst verborgen sind, sollen aufgezeigt werden.
Es muss eine Abbildung der Daten auf geometrische Beschreibungsformen gefunden werden, damit abstrakte Daten graphisch dargestellt werden können. Die Definition passender Abbildungen ist keine einfache Aufgabe. Es können bei einer falschen Wahl der Abbildung Bilder entstehen, die zu fehlerhaften Interpretationen der dargestellten Daten führen und damit fehlerhafte Entscheidungen zulassen.
Daher ist nach Robertson bei der Visualisierung anzustreben, dass der Betrachter ein mentales Modell entwickeln kann, bei dem die visuellen Attribute[1] einer Darstellung in definierter Form den Eigenschaften der Datenmenge entsprechen (vgl. [8]). Das mentale Modell des Betrachters umfasst folglich einen Kontext, in den der Betrachter die Darstellung einordnet. Der Erfolg der Visualisierung hängt deshalb in erster Linie davon ab, in welchem Maße der Betrachter fähig ist, den Kontext der realen Welt aus der Abbildung zu rekonstruieren und die wahrgenommenen Strukturen in der Abbildung mit den tatsächlich existierenden Korrelationen zwischen Parametern im Datenfeld zu assoziieren. Nur wenn dies gelingt, kann die Visualisierung als ein entscheidendes Hilfsmittel in der Datenanalyse eingesetzt werden [9].

Zusätzlich zu quantitativen Fragen ist eine große Zahl von Fragen, gezielt durch Visualisierung von qualitativer Natur, z.B., „bei einem gegebenen medizinischen Ultraschallbild eines Patienten, gibt es welche Anomalien, die klinische Probleme zeigen können?“ Eine typische Antwort auf diese Frage würde die Entdeckung von Mustern einbeziehen, die bestimmte Charakteristiken bezüglich Form, Position, oder Datenwerten haben, welche ein menschlicher Experte wie ein Arzt, basierend auf seiner früheren klinischen Erfahrung, als Anomalien klassifizieren würde. In solchen Fällen ist es ziemlich schwer, wenn nicht unmöglich, die Fragen unter Verwendung von voll-

[1] Sie bezeichnen die einzelnen Elemente, aus denen sich eine graphische Darstellung zusammensetzt.

automatischen Mitteln angesichts der vagen Definition der Frage und der hohen Variabilität der Eingabedaten zu beantworten. Der entscheidende Input des menschlichen Experten, unterstützt durch interaktive Visualisierungen, ist unabdingbar.
Ein weiteres fundamentales Feature von Visualisierungen ist ihr interaktiver Aspekt. Der Visualisierungsprozess ist nur selten statisch. In den meisten Applikationen gibt es ein Bedürfnis, große Menge von Daten, die nicht direkt auf einem einzelnen Bildschirm passen würden, eine hochdimensionale Datenmenge, die große Anzahl an unabhängigen Datenwerten pro Datenpunkt enthält, oder beides zu visualisieren. In solchen Fällen ist das Anzeigen eines statischen Bildes, das alle Daten enthält, nicht möglich. Außerdem, selbst wenn dies möglich ist, gibt es normalerweise viele Möglichkeiten für den Aufbau der Daten-zu-Bild-Abbildung, welche die Benutzer gerne ausprobieren würden, um die vorhandenen Daten besser zu verstehen. All diese Aspekte gewinnen von der Verwendung der interaktiven Visualisierungen. Solche Applikationen bieten die Möglichkeit zur Modifikation mehrerer Parameter, von dem Blickwinkel, Zoom-Faktor und Farbennutzung bis zu der Art der verwendeten Visualisierungsmethode und Beobachtung der Änderungen in dem erzeugten Bild [28].

Die Visualisierung findet in verschiedenen Anwendungsbereichen Einsatz. Die Visualisierung in mobilen Umgebungen stellt ein Gebiet dar, das von einem sehr stark ansteigenden Interesse gekennzeichnet ist. Generell verfügen mobile Endgeräte über eine deutlich geringere Graphikleistung, so dass einfachere Visualisierungstechniken zu verwenden sind oder eine stärkere Verteilung der Visualisierung durch die Einbeziehung von Rechnerkapazitäten auf der Server-Seite erfolgen muss. Zusätzlich bildet auch die begrenzte Netzwerkbandbreite einen limitierenden Faktor, der die Auswahl der einsetzbaren Visualisierungstechniken einschränkt. Unter diesen stark veränderten und deutlich beschränkenden Vorgaben wurden spezifische Lösungen im Bereich der Visualisierung entwickelt. Der Bereich der Visualisierung in mobilen Umgebungen wird trotz wachsender Rechenleistung mobiler Endgeräte auch in Zukunft ein Gebiet darstellen, in dem spezielle Lösungen benötigt werden [9].

4.1 Definitionen

Um den Begriff Visualisierung besser verstehen zu können, werden an dieser Stelle die Definitionen einer Visualisierung erklärt.

„Unter Visualisierung ist die rechnergestützte, visuelle Präsentation von Daten, Informationen und Wissen in einer für den Menschen adäquaten und für die jeweilige Anwendung in diesem Kontext sinnvollen Form zu verstehen.“ Einerseits ist die graphische Veranschaulichung großer Datenmengen von Bedeutung, andererseits werden auch an die Qualität der Visualisierung spezielle Anforderungen wie Expressivität, Effektivität und Angemessenheit gestellt.
Die Visualisierung von Daten, d.h. die bildliche Veranschaulichung ihrer relevanten Aspekte, spielt eine wichtige Rolle, damit sowohl das Verständnis als auch die Kommunikation entscheidend erleichtert werden.
Eine Vielzahl von Methoden und Techniken zur Visualisierung ganz unterschiedlicher Daten, z.B. medizinischer Daten wurde entwickelt [9].

Das Ziel von Visualisierung ist es, mittels interaktiver Graphiken, Einsichten in verschiedene Aspekte, bezogen auf einen Prozess, an dem man interessiert ist, wie eine wissenschaftliche Simulation oder einen realen Prozess, zu bekommen [28].
Es gibt verschiedene Definitionen von Visualisierung. Nach Williams et al., ist Visualisierung „ein kognitiver Prozess, ausgeführt durch Menschen bei der Bildung eines mentalen Bilds eines Domänenraums. In der Informatik und Informationswissenschaft ist das insbesondere die visuelle Repräsentation eines Domänenraums mit Graphiken, Bildern, animierten Sequenzen, und Geräuschverstärkung zur Darstellung von Daten, Struktur, und dynamischem Verhalten von großen, komplexen Datenmengen, die Systeme, Ereignisse, Prozesse, Objekte, und Konzepte repräsentieren“ [7].

Im Allgemeinen ist ein Visualisierungsprozess ein Prozess, der verschiedene Techniken wählt, damit verschiedene Arten von Daten (z.B. Skalar-Felddaten, Vektor-Felddaten, ...) auf Computern dargestellt werden können, so dass sie von anderen verstanden werden können. Ein solcher Visualisierungsprozess besteht aus einer Menge von Schritten. Jeder Schritt übernimmt einen Teil der Arbeit in dem gesamten Visualisierungsprozess [4].

Nach der Definition von McCormick, DeFanti und Brown: „Visualisierung ist eine Berechnungsmethode. Sie transformiert symbolische Informationen in geometrische und erlaubt dadurch Wissenschaftlern die Betrachtung ihrer Simulationen und Berechnungen. Visualisierung bietet eine Methode, um das Unsichtbare zu sehen. Sie bereichert den Prozess der wissenschaftlichen Entdeckung und fördert tiefgreifende und unerwartete Erkenntnisse. In vielen Bereichen gestaltete sie bereits die Art der Wissenschaftler, Wissenschaft zu machen, um [5].

4.2 Anpassung der Visualisierung

Durch die Visualisierung soll die Darstellung von komplexen Prozessmodellen in einer an die Bedürfnisse des Benutzers angepassten Art und Weise ermöglicht werden. Es existieren im Prinzip mehrere Möglichkeiten, wie eine Prozessdarstellung konfiguriert bzw. angepasst werden kann. Diese sind in Abbildung 4.1 zusammengefasst und werden hier kurz erklärt.

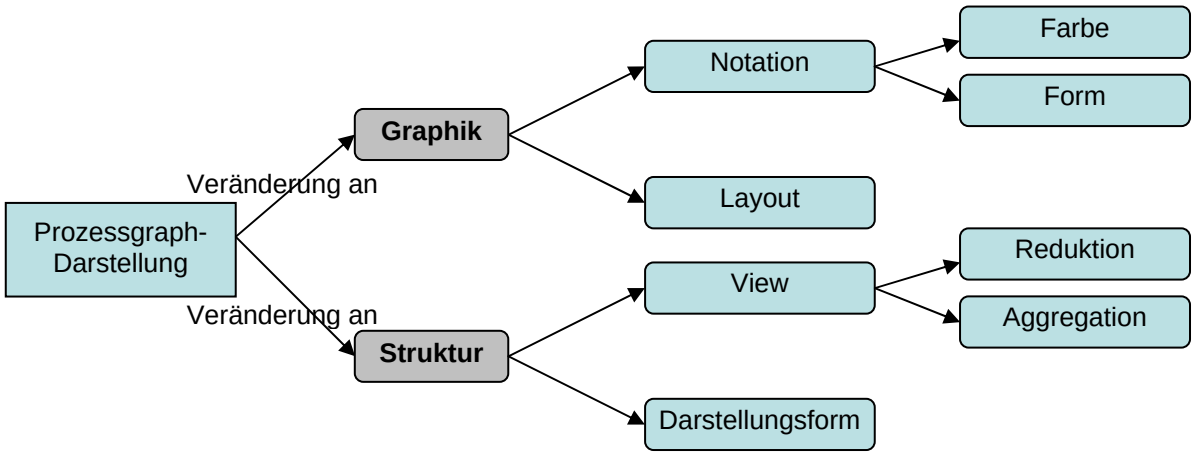

Abbildung 4.1: Freiheitsgrade der Gestaltung einer Prozessvisualisierung

Graphische Anpassung

Im Allgemeinen kann eine Prozessvisualisierung durch rein graphische Methoden verändert werden, ohne dass inhaltliche Änderungen vorgenommen werden.
Es kann einerseits die Notation, d.h. die zur Darstellung verwendeten Symbole, variiert werden. Durch Form- und Farbveränderungen der einzelnen Symbole wird ebenfalls eine unterschiedliche Wahrnehmung bei den Betrachtern erzielt. Aus diesem Grund hat die Gestaltung der Notation erheblichen Einfluss auf die Akzeptanz einer Visualisierung. Wenn ein Prozess in einer dem Betrachter vertrauten Art und Weise dargestellt wird, wird dadurch die Akzeptanz erhöht und der Einarbeitungsaufwand verringert [22].
Andererseits kann man das graphische Erscheinungsbild durch Anpassung des Layouts beeinflussen. Ein sehr wichtiger Aspekt einer Prozessvisualisierung ist beispielsweise die Position der verschiedenen Prozessobjekte. Das Bild, das sich beim Betrachten eines Prozesses beim Betrachter einprägt, wird auch als Mental Map bezeichnet [27]. Die Bewahrung der Mental Map spielt eine große Rolle für die Wiedererkennung eines Prozesses bzw. eines Graphen und ist daher ein Hauptziel aller Layout-Algorithmen.

Strukturelle Anpassung

Die Darstellung von Prozessen kann neben graphischen Anpassungen auch strukturell verändert werden.
Beispielsweise kann die Menge der dargestellten Information reduziert werden, um die Komplexität der Prozessmodelle zu reduzieren, was die Unterstützung von Prozess-Views nahelegt. Mit Hilfe der Prozess-Views entfernt man Teile des Prozessgraphen (Reduktion) oder fasst sie zusammen (Aggregation), ohne dass die Integrität des Prozessmodells verletzt wird.
Die Informationen, die im Prozessmodell enthalten sind, werden in der Regel als Prozessgraph dargestellt. Alternative Darstellungsformen (s. Kapitel 4.5) verwenden dieselben Informationen, aber sie bereiten sie auf eine andere Art und Weise auf. Prozessinformationen können so zum Beispiel auch in Form von Tabellen oder Projektplänen dargestellt werden.
Durch Ausnutzung all dieser Freiheitsgrade können Prozessvisualisierungen auf die Bedürfnisse des Betrachters zugeschnitten werden.

Im Folgenden wird die graphische Anpassung der Visualisierung ausführlicher betrachtet.

- Notation

Prozessmodelle sind abstrakte Datenstrukturen. Sie werden erst durch die graphische Darstellung für den Benutzer greifbar. Eine graphische Prozessdarstellung wird durch die verwendete Prozessnotation definiert. Sie besteht aus einer Menge von Symbolen, von denen normalerweise ein bestimmtes Symbol einem Metamodellelement zugeordnet ist. D.h. eine Prozessnotation spezifiziert je ein Symbol für eine Aktivität, ein Datenelement, eine Kontroll- bzw. Datenflusskante etc. In der Wahrnehmung der Endbenutzer ist die Prozessnotation ein wesentlicher Aspekt. Eine Notation, die als unpassend empfunden wird, wirkt sich entsprechend negativ auf die Akzeptanz von Prozessdarstellungen aus. Es ist deshalb wichtig, für die jeweilige Zielgruppe oder sogar für jeden Benutzer eine passende Prozessnotation wählen zu

können. Dafür muss eine Prozessvisualisierungskomponente in der Lage sein, unterschiedliche Notationen und deren Elemente flexibel einzusetzen [22].
In den Ingenieurwissenschaften und in der Wirtschaft wird rot meistens als Gefahrensignal, blau als neutral und grün als positives Signal gewertet. Jedoch steht in der Medizin rot für Leben und ist daher positiv belegt. Grün und blau dagegen werden mit Infizierung bzw. Tod in Zusammenhang gebracht und werden deswegen in manchen Situationen durchaus negativ interpretiert.
Berufsspezifische Konventionen und Erfahrungen können sich auch auf einer höheren Ebene der Visualisierung auswirken. Ingenieure und Techniker besitzen so grundsätzlich bereits durch Ausbildung eine große Erfahrung im Umgang mit abstrakten Diagrammen und sind geschult in ihrer Interpretation. Der Umgang mit Farbe und Form steht dagegen im biologisch/medizinischen Bereich stärker im Vordergrund, gerade auch im Kontext realitätsnaher Darstellungen. So sind Mediziner beispielsweise geschult auch in einer Projektionsdarstellung eines Röntgenbildes, die sehr verzerrt ist, korrekte Positionen und Größenverhältnisse von Organen abzulesen [9].

- Layout

Das Thema Graph-Layout behandelt die Berechnung einer optimalen Positionierung der Graphelemente (Knoten und Kanten) auf der Zeichnungsebene. Dies ist für Prozessmodelle eine sehr anspruchsvolle Aufgabe. Es ergibt sich eine hohe Dynamik durch strukturelle und graphische Anpassung von Prozessen. Eine manuelle Anpassung des Layouts ist daher normalerweise zu aufwendig. Es sind folglich Algorithmen zur automatischen Berechnung des Layouts erforderlich.
Die zentrale Anforderung an einen Layout-Algorithmus kann man kompakt formulieren: Es ist ein schönes Layout gesucht, wobei „schön“ z.B. minimale Anzahl Kantenkreuzungen, Kantenlänge, Fläche bzw. Größe der Zeichnung, Erhaltung von Symmetrien, gleiche Darstellung ähnlicher Strukturen, etc. bedeuten kann.
Vielfältige Anforderungen können formuliert werden, die sich in zwei Kategorien einteilen lassen:

- Zeichenkonventionen: Kriterien, die vom Algorithmus unbedingt einzuhalten sind (Muss-Kriterien).

- Ästhetikkriterien: Zusätzlich zu den Zeichenkonventionen sind ebenfalls weitere Anforderungen zu erfüllen, die zu schöneren Zeichnungen führen (Nebenbedingungen bzw. Kann-Kriterien).

In [29] werden zwei Algorithmen vorgestellt, die speziell für das Layout von Prozessgraphen entwickelt wurden.
Der erste Algorithmus basiert auf einem Standardverfahren, das das Problem der Zyklen-verursachenden Schleifenkanten umgeht: dem Sugiyama-Algorithmus [30]. Für den Algorithmus werden Zeichenkonventionen, die sich konfigurieren lassen, definiert. Man legt beispielsweise die Mindestabstände zwischen den verschiedenen Prozessobjekten fest.
Unter den Ästhetikkriterien werden durch Constraints spezielle Anforderungen an das Layout beschrieben. Für den Algorithmus können folgende Constraints definiert werden:

- *Abstände zwischen Objekten:* Beispielsweise kann bei bestimmten Prozessvisualisierungen erwünscht sein, zwei bestimmte Aktivitäten in einem anderen als dem vorgegebenen Mindestabstand zu zeichnen.

- *Ausrichtung der Objekte:* Eine relative Ausrichtung von Aktivitäten kann mit Hilfe dieses Constraints vorgenommen werden (horizontal wie vertikal).

- *Reihenfolge von Verzweigungsästen:* Es wird durch dieses Constraint ermöglicht, eine bestimmte Reihenfolge von Verzweigungsästen in der Prozessvisualisierung festzulegen.

Ausgehend vom Prozessmodell und von den Constraints berechnet der Algorithmus ein neues Layout. Das Layout kann sich aber bei Änderungen am Prozess und seiner anschließender Neuberechnung vollständig ändern. Den Benutzern wird dadurch die Wiedererkennung des Prozesses erschwert.
Der zweite entwickelte Algorithmus – der Schiebealgorithmus für das Layout von Prozessgraphen, behebt dieses Problem. Er verwendet Informationen aus vorhandenen Prozessgraph-Layouts, kombiniert mit Informationen zu Änderungen am Prozessgraphen. Das Prinzip des Algorithmus basiert auf dem Verschieben existierender Teil-Layouts. Die unveränderten Teile des Graphen werden in Blöcken verschoben, es wird nur für veränderte Teile ein neues Layout berechnet. Der Vorteil dabei ist, dass das Verschieben der bestehenden Positionen sehr effizient realisierbar ist [22].

4.3 Daten als Ausgangspunkt einer Visualisierung

Als Gegenstand einer Visualisierung spielen Daten eine besondere Rolle im gesamten Visualisierungsprozess. Es ist daher auch wichtig, dass die Charakteristika einer Datenmenge beschrieben werden, um sie für Visualisierungsentscheidungen einzusetzen. Datenspezifikationen dienen dazu. Sie erfassen die Eigenschaften der beobachteten Merkmale (beispielsweise Datentyp), des Beobachtungsraumes wie auch der gesamten Datenmenge (Zusammensetzung). Eine Datenspezifikation sollte grundsätzliche Eigenschaften berücksichtigen und erweitert werden können, um anwendungsabhängige Besonderheiten erfassen zu können. Sie sollte zudem die gezielte Auswahl von Daten ermöglichen bzw. unterstützen [9].

Die Workflow-Management-Coalition (WfMC) unterscheidet drei Arten von Daten, die bei einer Prozessausführung wichtig sind:

1. *Workflow-Kontrolldaten* sind interne Statusdaten des Workflow Management Systems, beispielsweise zum Start/Ende von Aktivitäten.

2. *Workflow-relevante Daten* sind Anwendungsdaten, die dem WfMS bekannt sind. Das WfMS verwendet sie, um an alternativen Verzweigungen einen Ausführungspfad auszuwählen.

3. *Applikationsdaten* sind diejenigen Daten, die anwendungsspezifisch sind. Sie werden vom WfMS nicht interpretiert, sondern höchstens referenziert und an Applikationsprogramme weitergegeben [31].

Für die Visualisierung sind alle drei Datenarten interessant. D.h. sie müssen gegebenenfalls darstellbar sein bzw. mindestens verlinkt werden können. Für eine Instanzvisualisierung sind konkret folgende Daten wichtig und müssen deshalb der Visualisierungskomponente zugänglich gemacht werden:

- *Start- und Endzeitpunkte von Aktivitäten und Prozessinstanzen:* durch die Start- und Endzeiten einer Aktivität bzw. einer Prozessinstanz wird angegeben, wann die Aktivität bzw. die Prozessinstanz gestartet und wann sie abgeschlossen worden sind.

- *Zustand von Aktivitäten:* die einzelnen Aktivitäten können sich in verschiedenen Ausführungszuständen befinden, die dem jeweiligen Betrachter angezeigt werden sollen.

- *Konkreter Bearbeiter der Aktivitäten:* Es wird eine Bearbeiterformel im Modell zur Modellierungszeit des Prozesses hinterlegt. Sie beschreibt die in Frage kommende Gruppe von ausführenden Personen. Daraus ergibt sich der konkrete Bearbeiter erst zur Laufzeit (i. d. R. bei Start oder Auswahl der Aktivität).

- *Applikationsdaten:* Applikationsdaten werden in die Prozessdarstellung für eine aussagekräftige Visualisierung integriert. Dadurch erlauben sie dem Betrachter zum Beispiel durch die Darstellung der Teilenummer in einem Änderungsmanagement-Prozess eine schnelle Zuordnung der Instanzdarstellung zu einem konkreten Geschäftsvorfall. Während sich einfach strukturierte Datenwerte direkt in die Visualisierung einbauen lassen, werden komplexere Daten (wie Dokumente oder CAD-Zeichnungen) üblicherweise verlinkt.

Zusätzlich zu den Prozesselementen (d.h. Aktivitäten, Datenelementen, Bearbeiterzuordnungen etc.) sollten weitere Applikationsdaten wie bspw. Statusinformationen über den bearbeitenden Antrag angezeigt werden können, damit die Betrachter und die Prozessbeteiligten einen möglichst hohen Nutzen aus den Prozessdarstellungen ziehen können.
Man kann als anderes Beispiel für Applikationsdaten auch eine Fahrzeug-Baureihe in der Überschrift einer Prozessdarstellung angeben. Dadurch wird einem Ingenieur in der Fahrzeugentwicklung die Übersicht erleichtert, an welcher Baureihe er mitarbeitet. Wenn man zum Beispiel einer Instanzdarstellung eines Änderungsmanagement-Prozesses eine Beschriftung mit der Änderungsnummer hinzufügt, hilft dies dem betroffenen Ingenieur bei der Einordnung der Darstellung. Eine wesentlich anschaulichere Visualisierung lässt sich durch die Verwendung einfacher graphischer Mittel erreichen.
Weiter sollen Applikationsdaten auch innerhalb der Symbole anzeigbar sein, um zum Beispiel Daten, die während der Prozessausführung entstehen, in die Prozessvisualisierung einfließen zu lassen.

Oft werden in Unternehmen große Prozessmodelle zu Dokumentationszwecken angefertigt. Manchmal ergeben sich in Abhängigkeit von der Modellierungsmethodik riesige Prozessmodelle, welche die Größe von DIN A0 weit überschreiten ("Wandtapeten"). Es ist für die unterschiedlichen Prozessbeteiligten sehr schwer, "ihren" Anteil in der Gesamtvisualisierung des Prozesses zu identifizieren. Durch farbliche Hinterlegungen oder Anordnung der Aktivitäten in Bahnen wird in der Regel versucht, die Orientierung der Betrachter zu unterstützen. Hier sind Mechanismen dringend erfor-

derlich, um aus dem komplexen Gesamtprozessmodell übersichtliche, rollenspezifische Teilmodelle abzuleiten [22].
Wie bereits erwähnt, setzt jede Visualisierung bestimmte Daten voraus. Z.B. für eine Visualisierung auf einer Karte ist die Position eines Tasks von großer Bedeutung.
Im Rahmen dieser Arbeit ist die Information über die Reihenfolge der Tasks als Ausgangspunkt für die Visualisierung interessant. Diese Reihenfolge wird durch die Constraints definiert.

4.4 Grober Ablauf der Visualisierung

Ziel des Visualisierungsprozesses ist es, abstrakte Daten, die im Regelfall nicht-geometrischer Natur sind, in Form von Bildern zu veranschaulichen.
Das Sichtbarmachen von Zusammenhängen, die versteckt innerhalb von Systemen liegen, ist Aufgabe der Informationsvisualisierung. Die Zusammenhänge, speziell die Reihenfolge der Aktivitäten sowie deren Relation zu Prozessdaten, Bearbeitern und Systemen sind bei Prozessen graphisch darzustellen. Es werden bei der Erzeugung der Bilder von den Rohdaten im System zum fertigen Bild mehrere Schritte durchlaufen, die sich in einer so bezeichneten Visualisierungspipeline anordnen lassen. Abbildung 4.2 zeigt die vier Schritte, die sich in Datenbereitstellung und Darstellungsgenerierung unterteilen lassen.

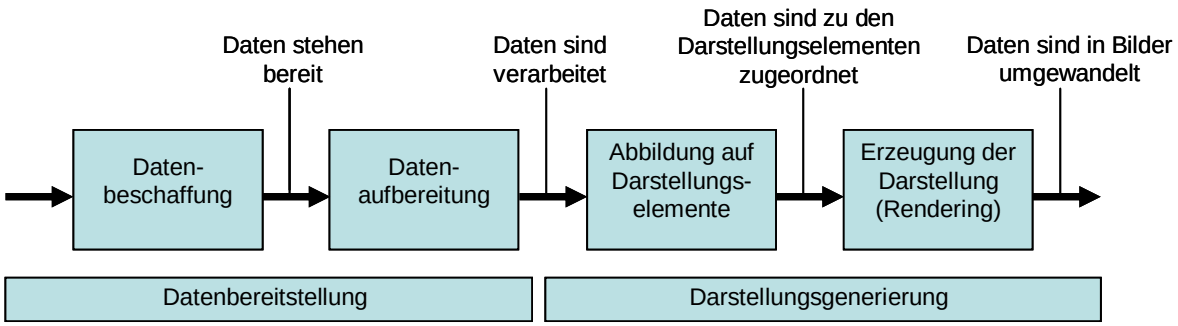

Abbildung 4.2: Visualisierungsprozess

Allgemeiner Ablauf für die Visualisierung von Daten

1. Datenbeschaffung: Die darzustellenden Rohdaten werden bereitgestellt, so dass sie für die Visualisierung verfügbar sind. Als Rohdaten bezeichnet man die in einer Anwendung erhobenen Daten.

2. Datenaufbereitung: Ausgangspunkt in diesem Schritt sind die Rohdaten. Durch die Datenaufbereitung wird eine Daten-zu-Daten-Abbildung realisiert. Die Rohdaten werden so aufbereitet, dass sie für die gewählte Art der Visualisierung geeignet sind. Die Daten können dazu zum Beispiel gruppiert, aggregiert oder gefiltert werden, oder neue Datenwerte werden auf Grundlage der existierenden berechnet. Die so bezeichneten aufbereiteten Daten liegen im Ergebnis der Datenaufbereitung vor und werden an die folgenden Schritte der Visualisierungspipeline übergeben.
3. Abbildung auf Darstellungselemente: Ausgangspunkt hier sind die aufbereiteten Daten, welche geometrischer und nicht-geometrischer Natur sein können. Die

nicht-geometrischen Daten werden in diesem Schritt in Geometriedaten überführt. Anders gesagt, es wird eine Daten-zu-Geometrie-Abbildung realisiert, bei der die Datenwerte auf geometrische Primitive einschließlich der zugehörigen Attribute (beispielsweise Farbe) abgebildet werden. Die Daten werden also in dieser Phase graphischen Objekten oder Eigenschaften zugeordnet. Anordnungsfragen und die Integration der bereits vorhandenen geometrischen Daten werden zudem geklärt. Es wird dadurch festgelegt, wie ein bestimmter Datenwert in der Darstellung repräsentiert werden soll und in entscheidendem Maße die spätere visuelle Repräsentation der Daten beeinflusst. Zum Beispiel kann dies bei Prozessen die Zuordnung eines Datenobjekttyps (z.B. Aktivität) zu einem Symbol (z.B. Aktivitätensymbol) sein, oder die Zuordnung eines Aktivitätenstatus zu einer Hintergrundfarbe.

4. Erzeugung der Darstellung (Rendering): In der Rendering-Phase erfolgt die Abbildung der Geometriedaten in Bilddaten, d.h. aus den Daten und deren zugeordneten Darstellungselementen wird ein Bild erzeugt. Es wird eine Graphikdatei aus den in einem internen Repräsentationsformat vorliegenden Informationen über die darzustellenden Objekte erzeugt, die auf dem Bildschirm des Betrachters angezeigt werden kann.

Die Bildausgabe (Display) auf einem Ausgabegerät erfolgt nach der Bilderzeugung.
In Abbildung 4.3 wird die Visualisierungspipeline noch einmal aus Sicht des Datenflusses zusammengefasst.

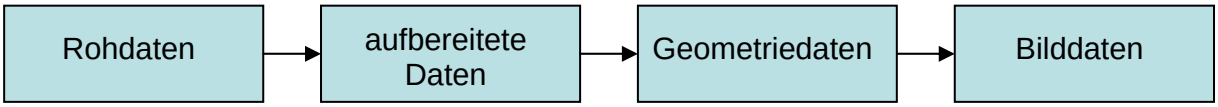

Abbildung 4.3: Datenfluss in der Visualisierungspipeline

Der allgemeine Ablauf für die Visualisierung von Daten wird nun auf die Domäne der Prozessvisualisierung übertragen. Im Folgenden werden die allgemeinen Phasen in Bezug auf ihre Aufgabe bei der Prozessvisualisierung erörtert.

Ablauf der Visualisierung von Daten bei der Prozessvisualisierung

1. Datenbeschaffung: Die Prozessdaten werden in dieser Phase zur Verfügung gestellt. Insbesondere zählen dazu die Prozessmodelldaten, die oft fragmentiert (d.h. über mehrere Informationssysteme verteilt), in diversen Formaten auf den Quellsystemen vorliegen. Nun müssen sie in ein einheitliches Format überführt werden, das dann als Grundlage für die Visualisierung dient. Die Prozessfragmente der verschiedenen Quellsysteme werden in diesem Zusammenhang noch zu einem Gesamtmodell integriert. Bei der Visualisierung von Prozessinstanzen soll eine Anbindung an die Laufzeitdaten der Quellsysteme realisiert werden, so dass aktuelle Daten zum Prozessstatus wie auch andere Applikationsdaten in die Visualisierung integriert werden können. Die Prozessdaten liegen als einheitliches Gesamtmodell im System als Ergebnis dieser Phase vor. Es kann auf die zugehörigen Laufzeitdaten der Prozesse über eine definierte Schnittstelle zugegriffen werden.

2. Aufbereitung der Daten: Für die Visualisierung im Allgemeinen ist das integrierte Gesamtmodell noch zu komplex und beinhaltet gegebenenfalls für den aktuellen Betrachter unwichtige Details. Es ist deshalb erforderlich, die Prozessmodelle auf-

zubereiten bzw. zu vereinfachen. Es können Strukturänderungen am Prozessmodell vorgenommen werden. Konkret kann man bestimmte Modellanteile aus dem Prozess entfernen (Reduktion) bzw. zusammenfassen (d.h. abstrahieren) (Aggregation). Anschließend wird die Visualisierung auf Grundlage solcher vereinfachender Prozess-Views generiert.

3. Abbildung auf Darstellungselemente: In dieser Phase wird für jedes Prozessobjekt (z.B. Aktivität) festgelegt, welches Symbol der Prozessnotation für dessen Darstellung zu verwenden ist. Das spätere Aussehen der Visualisierung wird dadurch definiert.

4. Erzeugung der Darstellung: Diese Phase umfasst bei der Visualisierung von Prozessen drei Abschnitte.

   a) Das Layout muss zunächst berechnet werden, bevor die fertige Prozessgraphik erzeugt werden kann. D.h. es muss für jedes darzustellende Element der Visualisierung festgelegt werden, an welcher Stelle der Zeichnungsebene es zu positionieren ist. Die Positionsdaten lassen sich entweder manuell oder durch einen Graph-Layout-Algorithmus berechnen.

   b) Nun wird die Prozessvisualisierung aus dem Prozessmodell, den zugeordneten Symbolen und den berechneten Positionsdaten als Graphik erzeugt.

   c) Die erzeugte Graphikdatei wird an den Visualisierungs-Client übertragen, der in Abhängigkeit vom jeweiligen Einsatzszenario unterschiedlich realisiert sein kann (Thin-Client im Inter-/Intranet oder Rich-Client) [9], [22].

4.5 Darstellungsformen

Ein wichtiger Aspekt der Prozessvisualisierung ist die gewünschte Darstellungsform. Oft werden Prozesse und zugehörige Informationen in Form eines Prozessgraphen dargestellt. Dieser umfasst je nach Detaillierungsgrad zusätzlich zum Kontrollfluss weitere Aspekte wie Datenfluss, Bearbeiterzuordnungen, IT-Systeme, zeitliche Beschränkungen (bspw. Aktivitätendauern, Fristen oder zeitliche Abstände von Aktivitäten) und Laufzeitdaten (z.B. Ausführungszustand, Bearbeitungszeiten, tatsächlicher Bearbeiter, Applikationsdaten). In verschiedenen Anwendungsfällen sind neben der Prozessgraphdarstellung auch weitere Darstellungsformen für Prozessinformationen wünschenswert. Hier wird auf einige Beispiele für Darstellungsformen näher eingegangen:

Prozessgraph

Der Prozessgraph konzentriert sich bei der Darstellung auf den Kontrollfluss. Die Darstellung kann je nach Bedarf um weitere Details wie Datenfluss und Bearbeiterzuordnungen angereichert werden. Dadurch kann aber die Komplexität der Visualisierung schnell steigen, so dass unerfahrene Betrachter häufig Probleme mit zu detaillierten Prozessdarstellungen haben. Abbildung 4.4 veranschaulicht ein Beispiel für einen Prozessgraphen eines Änderungsmanagement-Prozesses.

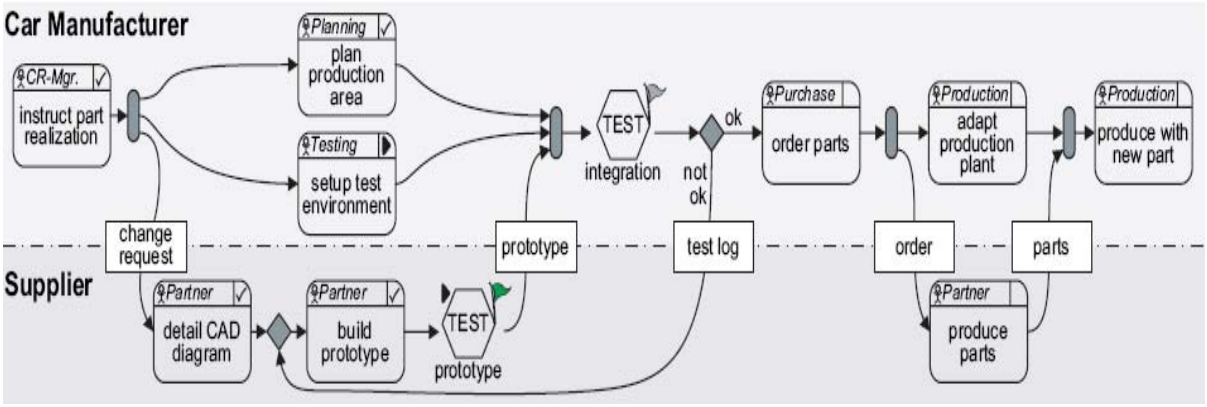

Abbildung 4.4: Prozessgraph eines Änderungsmanagement-Prozesses

Swimlane

Eine Besonderheit der Swimlane-Darstellung ist, dass der Prozessgraph in mehrere Bahnen partitioniert wird. Beispielsweise kann eine Organisationseinheit eines Unternehmens jeder Bahn zugeordnet sein, so dass aus der Darstellung erkannt werden kann, welche Einheiten welche Aktivitäten ausführen. Es können generell beliebige Prozessinformationen für die Partitionierung herangezogen werden. Hierarchisierungen der Bahnen sind ebenfalls möglich. Abbildung 4.5 zeigt ein Beispiel für eine Swimlane-Darstellung.

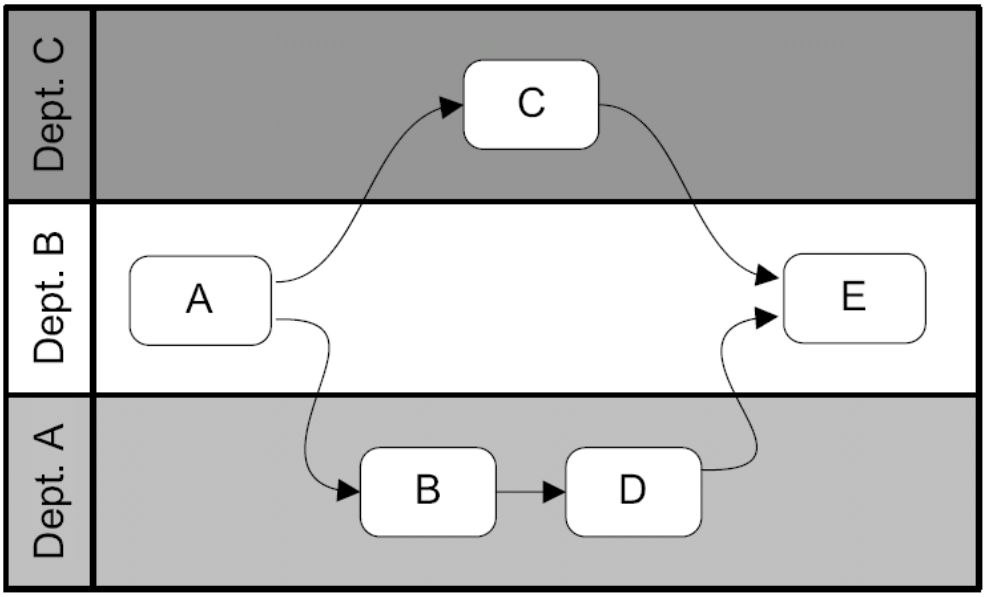

Abbildung 4.5: Swimlane

Gantt-Diagramm

Wenn zu den Aktivitäten des Prozesses entsprechende Start- und Endzeiten oder Ausführungsdauern hinterlegt sind, kann mit Hilfe dieser Daten ein Gantt-Diagramm generiert werden. Im Projektmanagement ist diese Darstellung sehr verbreitet und Managern deshalb vertraut. Man kann dieser Darstellungsform vor allem die zeitlichen Zusammenhänge innerhalb eines Prozesses besonders gut entnehmen (s. Abbildung 4.6).

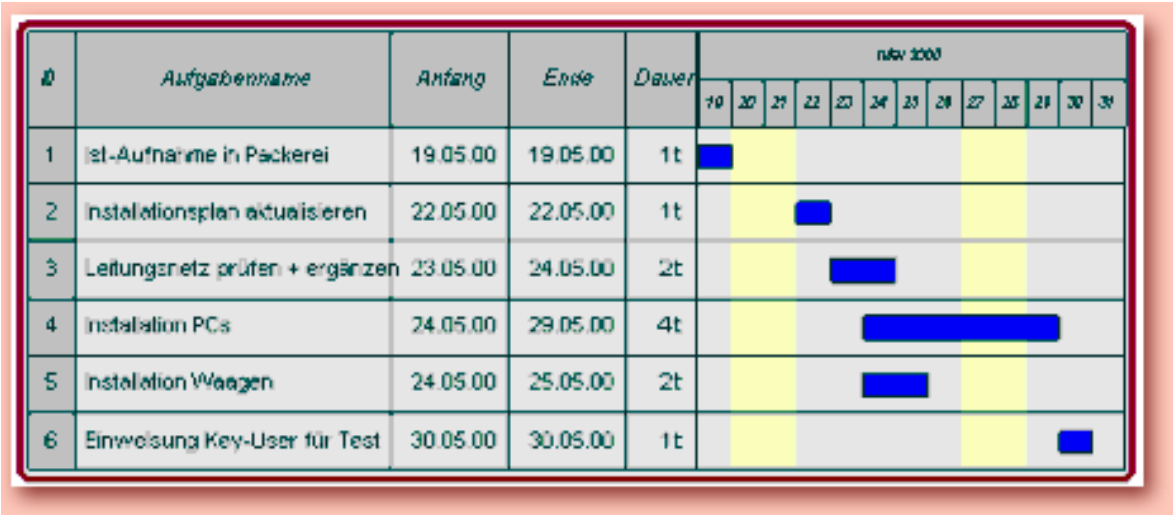

Abbildung 4.6: Gantt-Diagramm

Prozessterminplan
Der Prozessterminplan rückt ähnlich dem Gantt-Diagramm auch die temporalen Aspekte des Prozesses in den Vordergrund. Jedoch verwendet der Prozessterminplan im Gegensatz zum Gantt-Diagramm nicht für jede Aktivität eine eigene Zeile. Insbesondere bei großen Prozessen ergeben sich dadurch kompaktere Darstellungen.

Tabelle
Eine Tabelle ist ebenfalls für eine übersichtliche Darstellung eines Prozesses geeignet. Dabei werden in der einfachsten Version nur die Aktivitäten aufgelistet. Zusätzliche Tabellenspalten können bei Bedarf Informationen zu Daten oder Bearbeitern der Aktivitäten enthalten. Während sich sequentielle Prozesse mittels dieser Darstellungsform sehr gut und einfach abbilden lassen, gestaltet sich die Darstellung von nicht-sequentiellen Prozessen (z.B. parallele oder zyklische Prozesse) schwierig.

Matrix-Darstellung
Bei der Matrix-Darstellung wird von der Ablauflogik des Prozesses abstrahiert und versucht, Zusammenhänge zwischen Prozessobjekten zu vermitteln. In einer Matrix kann zum Beispiel übersichtlich dargestellt werden, welche Bearbeiter auf welche Dokumente innerhalb des Prozesses zugreifen. Wenn zu den Bearbeitern auch Informationen zu deren Position innerhalb der Organisationshierarchie (bspw. aus dem Organisationsmodell des Unternehmens) vorliegen, können die Daten in der Matrix hierarchisiert dargestellt und entsprechend „auf- bzw. zugeklappt“ werden. Diese Darstellungstechniken sind aus Data-Mining-Anwendungen bekannt. Sie dienen dort der Exploration von Datenwürfeln.

Interaktionsdiagramm
Eine andere Darstellungsform ist das Interaktionsdiagramm. Es kann aus dem Swimlane-Diagramm entnommen werden, welche Abteilungen an welchen Stellen der Prozesse interagieren. Das Interaktionsdiagramm abstrahiert nun von den einzelnen Aktivitäten und illustriert dafür nur die Schnittstellen (bspw. Austausch von Dokumenten) zwischen zwei Partnern. Diese Darstellung macht das Erkennen von Abhängigkeiten bei der Optimierung von Prozessen leichter (s. Abbildung 4.7).

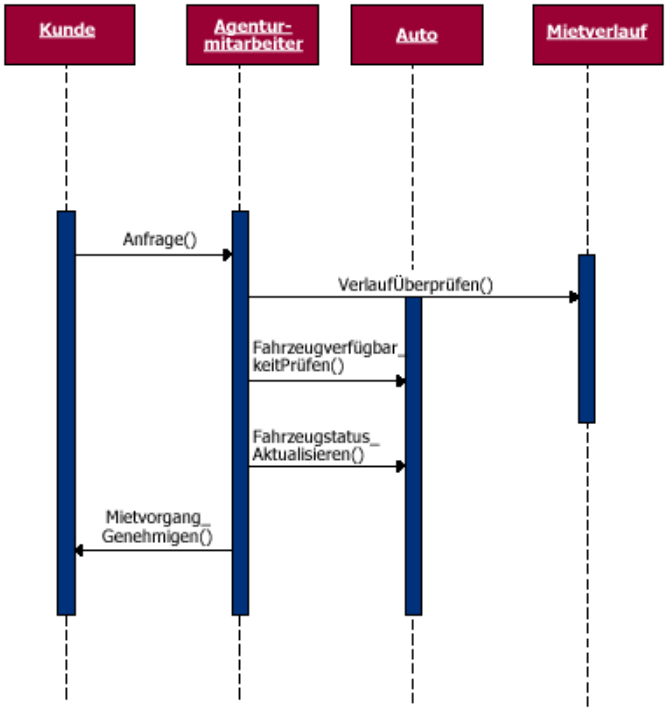

Abbildung 4.7: Interaktionsdiagramm

In der Literatur werden neben diesen Beispielen weitere Darstellungsformen vorgeschlagen bzw. verwendet. Z.B. in [23, 24, 25, 26] werden Prozessmodelle in Form von U-Bahnnetzplänen dargestellt.
Für die Realisierung der unterschiedlichen Darstellungsformen wird ein unterschiedlicher Aufwand erforderlich. Einige Formen (bspw. Tabellen oder Matrix-Darstellung) können graphisch sehr einfach realisiert werden. Es existiert eine Reihe mächtiger Werkzeuge für andere Formen, wie Gantt-Diagramme. Der Prozessgraph und die damit stark verwandte Swimlane-Darstellung sind in Bezug auf die Generierung die anspruchsvollsten Darstellungsformen. Sie zeichnen sich durch eine hohe Komplexität aus, die aus der komplizierten graphischen Struktur hervorgeht. Sowohl Gestalt und Inhalt der Prozesssymbole sowie deren Position lässt sich beliebig verändern. Bei Tabellen kann dagegen nur die Darstellung der Tabellenzeile variiert werden, wobei die Symbolik sehr beschränkt ist. Die Tabellenzeilen werden zudem immer sequentiell angeordnet, eine freie Positionierung oder eine Darstellung zeitlicher Abhängigkeiten durch Kanten ist nicht möglich [22].
Im Rahmen dieser Arbeit werden Prozessdarstellungen mittels Prozessgraphen realisiert, wobei sich die vorgestellten Konzepte teilweise auch auf andere Darstellungsformen anwenden bzw. übertragen lassen.

5 Anforderungsanalyse

5.1 Anforderungen an eine graphische Konfiguration von Prozessvisualisierungen

- *Verwendung beliebiger Symbole für Prozesselemente*
  Für die Darstellung von Prozesselementen müssen beliebige Symbole von einer Prozessvisualisierungskomponente zugelassen werden. Einfache Symbole, die nur den Namen der Aktivität enthalten, müssen auch dargestellt werden können, wie komplexe Symbole mit vielen Datenwerten. Es sollen sich zur ansprechenden Gestaltung der Symbole beliebige Graphiken einbinden lassen, zum Beispiel um eine automatisch ausgeführte Aktivität durch das Symbol eines Rechners zu repräsentieren.

- *Symbolauswahl abhängig von Attributwerten*
  Die Verwendung eines Symbols muss generell in Abhängigkeit von Attributwerten des Prozesselements gestaltet werden können. Beliebige Prozessdaten sollten dabei von den Verwendungsregeln referenziert werden können. Welches Symbol für die Repräsentation eines Prozesselements eingesetzt wird, ist somit nicht allein durch den Typ des Prozesselements (Aktivitätenknoten, Bearbeiterknoten, Kontrollflusskante, etc.) vorgegeben, sondern auch durch Attribute. Beispielsweise wird in Abbildung 4.4 für die Darstellung von Test-Aktivitäten ein spezielles Symbol verwendet.

- *Dynamische Verwendung von Darstellungssymbolen*
  Die Darstellung eines bestimmten Prozesselements muss sich dynamisch adaptieren lassen. Die Verwendung der Symbole sollte im Verlauf der Visualisierung einer Prozessinstanz mit fortschreitender Ausführung des Prozesses anpassbar sein. Für abgeschlossene Aktivitäten könnten zum Beispiel andere Symbole verwendet werden als für in Ausführung befindliche Prozessschritte. Die Dokumente in einem Prozess sollten sich gleicherweise in einem anderen Szenario in Abhängigkeit von ihrem Fertigungsgrad unterschiedlich visualisieren lassen. Auch unter Einbeziehung von Laufzeitdaten sollte eine solche dynamische Auswahl der Symbole erfolgen können.

- *Dynamische Symbole*
  Zusätzlich zu statischen Symbolen, die Datenwerte in Textfeldern darstellen können, müssen auch dynamische Symbole unterstützt werden. Die dynamischen Symbole verändern ihr Aussehen mit den sich ändernden Prozessdaten. Kostenattribute oder Zufriedenheitswerte sind typische Beispiele, die visuell aufbereitet als Balken- bzw. Tachodiagramme dargestellt werden sollen.

- *Personalisierung*
  Die Personalisierung der Darstellung für den jeweiligen Benutzer ist eine zentrale Anforderung für die Visualisierung von Prozessen. Mitarbeiter sind oft durch die Prozesswerkzeuge, die in ihrer Abteilung eingesetzt werden, bereits an eine bestimmte Notation gewohnt. Deshalb führt die Verwendung einer hiervon abweichenden Notation bei der Prozessvisualisierung zu einer verringerten Benutzerakzeptanz. Neben dem vollständigen Austausch einer Notation kann eine personalisierte Darstellung auch die Anpassung von Farben, Schriften und Linienformaten erfordern.

- *Präzise Spezifikation der Verwendung von Symbolen*
  Die Spezifikationen zur Verwendung von Symbolen in der Prozessdarstellung müssen korrekt und eindeutig sein. Wenn eine Auswertung entsprechender Regeln für ein Prozesselement mehrere mögliche Symbole ergeben sollte, muss klar definiert sein, wie mit derart widersprüchlichen Anweisungen umgegangen werden soll. Es muss alternativ durch die Art der Spezifikation sichergestellt werden, dass derartige Situationen nicht auftreten können.

- *Einfache Neuerstellung von Prozessnotationen und -symbolen*
  Der Aufwand für die Neuerstellung einer Prozessnotation sollte so gering wie möglich gehalten werden. Es ist anzustreben, dass Symbole, die in einem Graphikprogramm gezeichnet wurden, direkt in die Symboldefinition übernommen werden können. Es sollten darüber hinaus keine Änderungen an der Beschreibung des Graphikobjekts mehr notwendig sein, um zum Beispiel zu definieren, an welcher Stelle des Symbols bestimmte Daten angezeigt werden sollen. Ansonsten würde jede graphische Symbolanpassung dazu führen, dass die erforderlichen Änderungen erneut durchgeführt werden müssen.

- *Einblendung von Datailinformationen*
  Prozessdaten, die entweder nicht so wichtig oder für die Darstellung im Symbol zu lang sind, müssen sich bei Bedarf einblenden lassen.

- *Anzeige von Zusatzelementen*
  Neben dem Prozessmodell (bzw. der Prozessinstanz) erfordert eine benutzergerechte Prozessdarstellung die Anzeige weiterer Daten, die für den Benutzer relevant sind. Eine aussagekräftige Überschrift erlaubt zum Beispiel eine bessere Einordnung des Prozessmodells. Die Darstellung von Prozess- oder Applikationsdaten erleichtert es außerdem, betroffene Geschäftsvorfälle schnell zu identifizieren, ohne dass beispielsweise Dokumente, die mit dem Prozess verknüpft sind, geöffnet werden müssen. Diese globalen Zusatzelemente einer Prozessdarstellung müssen in ihrer graphischen Erscheinung wieder frei konfiguriert werden können. Ferner müssen beliebige Daten, die im Prozess oder in Applikationssystemen vorhanden sind, darstellbar bzw. verknüpfbar sein.

- *Parametrisierbare Beschreibung einer Prozessdarstellung*
  Die Gesamtheit aller möglichen bzw. erlaubten Darstellungen muss von einer Visualisierungskomponente als Kombinationen aus Prozessen, Instanzen, Prozess-Views, verschiedenen Darstellungsformen und Notationen verwaltet werden. D.h. ein entsprechend parametrisierbares Modell muss entworfen werden, das zum Zeitpunkt der Darstellung interpretiert wird. Die Darstellung des Prozesses wird hieraus generiert.

Die bisher beschriebenen Anforderungen beziehen sich auf die funktionalen Eigenschaften einer Prozessnotation, damit möglichst ansprechende und nützliche Prozessvisualisierungen erzeugt werden können. Die letzte Anforderung beschreibt des Weiteren eine Anforderung an eine Visualisierungskomponente als Ganzes, die für die Darstellung vieler Prozessmodelle mit jeweils unterschiedlichen Notationen zu verwenden ist.
Es existiert, neben diesen funktionalen Anforderungen, eine Reihe nicht-funktionaler Anforderungen, die für die Einsetzbarkeit einer Visualisierungskomponente in der Praxis relevant sind. Sie werden im Folgenden erläutert.

- *Einfacher Zugang zu Prozessvisualisierungen über Intranet*
  Prozessdarstellungen müssen im Intranet zur Verfügung stehen, um sie den Mitarbeitern im Unternehmen einfach zugänglich zu machen.

- *Verfügbarkeit von Viewern für verwendete Graphikformate*
  Es müssen passende Anzeigeprogramme (Viewer) für das bei der Prozessdarstellung verwendete Graphikformat verfügbar sein. Die Verteilung zusätzlicher Software ist gerade in großen Unternehmen mit organisatorischen Hürden versehen. Dem sind Graphikformate vorzuziehen, für die auf jedem Arbeitsplatz standardmäßig die passende Software installiert ist.

- *Skalierbarkeit der Graphiken*
  Die Skalierbarkeit der visualisierten Prozessdiagramme ist ein wichtiger Aspekt, der die Wiederverwendbarkeit und damit den Nutzen einer Prozessdarstellung erhöht. Wenn die Darstellungen für Dokumentationen oder als Diskussionsgrundlage eingesetzt werden sollen, ist es wichtig, dass die Grafik verlustfrei auf verschiedenen Medien und in unterschiedlichen Größen dargestellt werden kann.

- *Verwendung von standardisierten Formaten*
  Bei der Implementierung sollen möglichst standardisierte Formate eingesetzt werden, um die Unabhängigkeit von Produkten und deren Versionen zu sichern. Für solche Standardformate ist durch die breite Nutzerbasis am ehesten eine langfristige Verfügbarkeit gegeben. Außerdem entfällt der Wartungsaufwand, der für Eigenentwicklungen eingeplant werden muss [22].

Die vorgestellten Anforderungen werden nochmals in Tabelle 5.1 zusammengefasst.

Anforderung an die Konfigurierbarkeit der graphischen Darstellung
Verwendung beliebiger Symbole für Prozesselemente
Symbolauswahl abhängig von Attributwerten
Dynamische Verwendung von Darstellungssymbolen
Dynamische Symbole
Personalisierung
Präzise Spezifikation der Verwendung von Symbolen
Einfache Neuerstellung von Prozessnotationen und -symbolen
Einblendung von Detailinformationen
Anzeige von Zusatzelementen
Parametrisierbare Beschreibung einer Prozessdarstellung
Einfacher Zugang zu Prozessvisualisierungen über Intranet
Verfügbarkeit von Viewern für verwendete Graphikformate
Skalierbarkeit von Graphiken
Verwendung von standardisierten Formaten

Tabelle 5.1: Anforderungen an die Konfigurierbarkeit der graphischen Darstellung

5.2 Anforderungen an die Inhalte von Prozessvisualisierungen

Es werden im Folgenden sowohl funktionale als auch nicht-funktionale Anforderungen an die Prozessvisualisierung beschrieben (vgl. Tabelle 5.2).

Anforderung an die Inhalte von Prozessvisualisierungen
Visualisierung von Prozessschemata und Prozessinstanzen
Ansprechende Darstellung durch fortschrittliche Layout-Verfahren
Visualisierung von systemübergreifenden Prozessen
Einbindung von Applikationsdaten in die Darstellung
Anbindung von verschiedenen Systemtypen
Vereinfachung des Prozessmodells durch Prozess-Views
Flexible Definition von Prozess-Views
Flexible Definition der Prozessnotation
Einfache Konfigurierbarkeit einer Visualisierung
Wartbarkeit
Darstellung der Prozessinformation in verschiedenen Formen
Prozessdarstellung über Inter-/Intranet zugreifbar
Zugriffsbeschränkung
Benutzerfreundliche Gestaltung der Anwendung
Sicherheit/Datenschutz

Tabelle 5.2: Übersicht über die Anforderungen an eine Prozessvisualisierung

5.2.1 Funktionale Anforderungen

Visualisierung von Prozessschemata und Prozessinstanzen:
Eine Visualisierungskomponente muss sowohl Prozessschemata als auch Prozessinstanzen (inkl. Statusdaten) angemessen darstellen können.

Ansprechende Darstellung durch fortschrittliche Layout-Verfahren:
Durch die Prozessvisualisierung müssen ansprechende Darstellungen erzeugt werden. Bei der Zeichnung der Prozesse soll zum Beispiel die verfügbare Fläche sinnvoll ausgenutzt werden und Überlappungen von Symbolen bzw. Kantenkreuzungen – soweit möglich und sinnvoll – vermieden werden.

Visualisierung von systemübergreifenden Prozessen:
Meistens werden komplexe Prozesse verteilt durch mehrere Systeme ausgeführt. Eine durchgängige Visualisierung der systemübergreifenden Prozesse ist gerade bei solchen fragmentierten Prozessen wünschenswert, selbst wenn keine übergeordnete Workflow-Steuerung existiert.

Einbindung von Applikationsdaten in die Darstellung:
Es ist für die Betrachter einer Prozessvisualisierung wichtig, dass alle relevanten Informationen aus den verschiedenen Laufzeitsystemen in der Prozessvisualisierung zur Verfügung stehen. Einerseits sind dies Laufzeitdaten, wie zum Beispiel Nummer und Name des aktuellen Änderungsvorhabens. Andererseits müssen auch relevante Dokumente (z.B. CAD-Modell und Stellungnahmen) in die Visualisierung integriert werden, so dass diese bei Bedarf direkt zugänglich sind.

Anbindung von verschiedenen Systemtypen:
Die Prozesse werden nicht immer von Workflow Management Systemen mit wohldefinierten Schnittstellen für den Zugriff auf Laufzeitdaten ausgeführt. Es handelt sich oft bei den ausführenden Systemen um Legacy-Applikationen (z.B. konventionell implementierte Applikationen, ohne Prozesssteuerung). Die Visualisierung muss unabhängig vom Systemtyp Zugriff auf die Laufzeitdaten haben, damit der Prozessstatus und weitergehende relevante Informationen dargestellt werden.

Anpassbarkeit an den Benutzer
Es existiert neben diesen eher allgemeinen Anforderungen eine Reihe spezifischer Anforderungen, welche die Personalisierung, d.h. die Anpassbarkeit der Prozessdarstellungen an die Bedürfnisse des jeweiligen Betrachters, betreffen.

Vereinfachung des Prozessmodells durch Prozess-Views:
Oft sind zu viele Details für den jeweiligen Betrachter in den Prozessmodellen enthalten. Für manche Darstellungen sollen ferner nur Teile des Prozesses angezeigt werden. Der Rest des Prozesses ist entweder komplett aus dem Modell zu entfernen oder nur in abstrakter Form darzustellen. Es wird deshalb ein Mechanismus benötigt, der es erlaubt, eine Sicht auf das Prozessmodell zu definieren. D.h. mit diesem Mechanismus lässt sich dieses Modell durch Zusammenfassen oder Entfernen von nicht benötigten Prozessobjekten vereinfachen.

Flexible Definition von Prozess-Views:
Derartige Sichten auf Prozessmodelle müssen einfach und flexibel definierbar sein. Die Menge der zu entfernenden Prozesselemente sollte zum Beispiel anhand von Attributwerten festgelegt werden können. So lässt sich schnell eine Sicht definieren, die beispielsweise nur die Aktivitäten des aktuellen Betrachters enthält oder die alle bereits abgeschlossenen Aktivitäten zusammenfasst.

Flexible Definition der Prozessnotation:
Die Prozessnotation muss zur optimalen Anpassung der Prozessdarstellung an die jeweiligen Erfordernisse der Anwendung bzw. des Betrachters frei definierbar sein. Dabei sollten sich sowohl die Form und Farbe der verwendeten Symbole, als auch die darin dargestellten Daten konfigurieren lassen. Die Verwendung der Symbole kann statisch festgelegt sein oder dynamisch abhängig von Laufzeitdaten bestimmt werden.

Einfache Konfigurierbarkeit einer Visualisierung:
Die Prozessvisualisierungskonfiguration sollte möglichst einfach erfolgen können. Alle Konfigurationsparameter, die für eine Visualisierung benötigt werden, sollten deswegen am besten zentral festlegbar sein. Insbesondere wäre eine Werkzeugunterstützung für den Visualisierungsdesigner vorteilhaft.

Wartbarkeit:
Der erwartete Nutzen muss in Relation zum Aufwand für die Definition und Pflege einer Prozessvisualisierung stehen. Die Pflege kann besonders bei sich ändernden Prozessmodellen, Visualisierungsvorstellungen oder Quellsystemen schnell sehr viel Aufwand generieren. Die Wartbarkeit der Gesamtlösung muss aus diesem Grund von vornherein im Auge behalten werden, zum Beispiel durch die Wiederverwendung von Notationsdefinitionen.

Darstellung der Prozessinformation in verschiedenen Formen:
Zusätzlich zu der Darstellung der Prozesse in Form von Prozessgraphen sollen auch weitere Darstellungsformen unterstützt werden.

5.2.2 Nicht-funktionale Anforderungen

Es existieren zusätzlich zu funktionalen Anforderungen, welche die direkt erlebbaren Funktionen der Visualisierungskomponente beschreiben, weitere Anforderungen, die eher die allgemeine Beschaffenheit der Anwendung widerspiegeln. Insgesamt sind diese Anforderungen aber auch relevant für die Akzeptanz der Visualisierungskomponente.

Prozessdarstellung über Inter-/Intranet zugreifbar:
Für den Endanwender müssen die Prozessvisualisierungen einfach zugänglich sein. Am besten wird dies durch eine Integration in das bestehende Intranet erreicht. Dies bedeutet für die zu entwickelnde Prozessvisualisierungskomponente, dass die Prozessdarstellungen in einem geeigneten Format zu erstellen sind. Die Visualisierungskomponente muss gleicherweise eine Integration in das Unternehmensportal erlauben. Der minimale Installationsaufwand auf dem Klientenrechner ist ein weiterer Vorteil dieser Lösung, obwohl dieses Argument durch die immer besseren Software-Verteilungsmechanismen zunehmend an Gewicht verliert.

Zugriffsbeschränkung:
Prozessmodelle enthalten sensible Informationen, die meistens nur einem begrenzten Nutzerkreis zur Verfügung gestellt werden sollen. Durch entsprechende Maßnahmen muss folglich sichergestellt werden, dass nur authentifizierte und autorisierte Benutzer Zugriff auf die Prozessdarstellungen erhalten. Für den Fall, dass ein Benutzer zum Beispiel eine Aktivität nicht sehen darf, sind Mechanismen vorzusehen, wel-

che die entsprechenden sensiblen Daten aus dem Prozessmodell ausblenden, anstatt dass der Zugriff auf das Modell vollständig verweigert wird.

Benutzerfreundliche Gestaltung der Anwendung:
Die benutzerfreundliche Ausgestaltung der Visualisierungsapplikation ist für den Betrachter einer Prozessvisualisierung sehr wichtig. Zusätzlich zu den funktionalen Aspekten ist sie mit entscheidend für die Akzeptanz der Gesamtlösung und damit auch für die Akzeptanz prozessorientierter Systeme an sich.

Sicherheit/Datenschutz:
Person-Centric Flows beinhalten viele Informationen über das spezifische Verhalten einer Person, die geschützt werden sollten [3], [22].

Es existieren auch andere Anforderungen, die im Rahmen dieser Arbeit anzugeben sind und im Folgenden betrachtet werden.
Menschliche Akteure sind beschränkt rational und haben nur eingeschränktes Wissen über die Zukunft. Folglich sind die Modelle oft unvollkommen, da sie normalerweise nicht alle möglichen Umstände berücksichtigen. Ein Prozessunterstützungssystem muss demzufolge Laufzeitänderungen am ursprünglichen Modell ermöglichen und dem Benutzer kontextabhängige Information über den laufenden Prozess als eine Basis für Argumentation der möglichen nächsten Schritte bereitstellen. Prozesskarten, eine Repräsentation von Modellen, können als Teil solcher kontextabhängiger Information dienen. Detailliertere Informationen dazu kann man in [20] finden.
Die Spezifität der Prozessstruktur ändert sich im Laufe der Zeit. Infolgedessen sollte ein Modell von Geschäftsprozessen in der Lage sein, die Palette der Prozessspezifität (von hoch spezifizierten und routinemäßigen bis hoch unspezifizierten und dynamischen Prozessen) zu erfassen. Ein Prozessunterstützungssystem sollte Prozessmodelle mit verschiedenen Spezifitätsgraden interpretieren können. Darüber hinaus sollte es Benutzer unterstützen, wenn sich die Prozessspezifität zur Laufzeit ändert.
Bei einer Änderung versuchen die Benutzer, das Problem zu lösen, indem sie ihre Interpretation der Struktur und des aktuellen Kontextes befolgen. Die entstehende Aktivität stützt sich auf eine Form der Struktur und somit eine Form von Spezifität. Folglich sollte ein System, das die Unterstützung einer entstehenden Aktivität plant, eine Struktur als eine kontextabhängige Basis für angesiedelte Improvisation bereitstellen. Prozesskarten (analog zu geografischen Karten) können eine solche Struktur anbieten.
Außerdem sollte ein Prozessunterstützungssystem die Änderung, Anordnung und die Ausführung von Prozessen zur Laufzeit ermöglichen sowie Mittel anbieten, die in eine existierende Umgebung integriert werden müssen (z.B. Verwendung einer offenen Schnittstelle).

Das wichtigste Ziel des Unterstützungssystems ist die Bereitstellung von Kontext für den Benutzer, damit er entscheiden kann, was er als Nächstes tun muss. Ähnlich dem Task Manager (s. Kapitel 2.3) hilft deswegen das System den Benutzern, To-Do-Listen und Dokumente (Ressourcen), welche spezifisch zu dem vorhandenen Task sind, gemeinsam zu benutzen.
Aus Sicht der Implementierung sollte das System eine gemeinsame, verteilt-zugängliche, hierarchische To-Do-Liste bereitstellen, die den Benutzern erlaubt, Dateien (wie Ressourcen) an jedes der To-Do-Elemente anzufügen.

Wenn sich der Benutzer entscheidet, einige maschinenlesbare Constraints zu einem To-Do-Element hinzuzufügen, bietet das System Constraint-Überwachungsdienste. Z.B. das Anfügen einer Frist an ein To-Do-Element könnte dem System erlauben, den Benutzer aufzufordern, wenn die Frist abläuft. Je mehr Constraints durch den Benutzer angegeben sind, umso nützlicher kann das System beim Unterstützen des Benutzers zur Erreichung seiner Ziele sein. Zusammenfassend kann man sagen, dass das System den Benutzern durch die Verwaltung von Constraints zwischen Tasks und Ressourcen hilft.
Benutzer bilden Constraints auf Attribute der Aktivitäten/To-Do-Elemente oder Ressourcen in den bestehenden Prozessmodellen ab. Typische Constraint-Typen können einschließen: Zeiteinschränkungen (z.B. Fristen), Budgetgrenzen (z.B. Mitarbeiteranzahl, verfügbare Finanzmittel), externe Faktoren (z.B. kein Transport in Europa), Spezifikation von Ressourcen (z.B. Typen von Prozessoren/vorgefertigten Servern auf Lager), etc.

Wenn der Benutzer das Ziel oder die Nachbedingung einer Aktivität in seiner To-Do-Liste spezifiziert, wird das System versuchen, dem Benutzer eine Reihe von möglichen Ansätzen vorzuschlagen, damit er seine Arbeit ausführt. Das System plant Tasks und Ressourcen zur Erreichung der Ziele und lässt den Benutzer entscheiden, welchen der möglichen Wege er nehmen will. Es ordnet die Ausführung der Tasks unter Verwendung von Ressourcen an, um die Ziele zu erreichen.
Je spezifischer eine Task-Beschreibung ist, desto mehr unterstützt das System den Benutzer und nimmt ihm einen Teil des Tasks ab. Je weniger spezifisch der Task ist, desto mehr muss der Benutzer arbeiten. Folglich lenkt die Spezifität einer Prozessbeschreibung die resultierende Form von Arbeitsteilung zwischen dem menschlichen Akteur und dem System.

Eine andere Anforderung an das Prozessunterstützungssystem ist die Bereitstellung von Kontextinformationen für die Auswertung, Bestimmung und Darstellung der nächsten Schritte. Das System liefert dem Benutzer umfangreiche kontextabhängige Information (vergangene Aktivitäten im Prozesskontext, Dokumente, bezogen auf diesen Prozess, andere beteiligte Akteure, etc.), um den aktuellen Zustand des Prozesses zu verstehen.
Jedoch kann in manchen Phasen passieren, dass der Benutzer nicht genau weiß, was er als Nächstes tun soll. Die möglichen Optionen für nächste Tätigkeiten können zum Beispiel außerhalb seiner Erfahrung liegen oder ein alternatives, neuartiges Vorgehen wird notwendig. Melone et. al. [15] haben beschrieben, wie ein Repository von wiederverwendbaren Prozesskomponenten sowie vergangenen Fällen auf Organisationsprozesse angewandt werden kann und nützlich in einem Prozess-Design und im Setzen von Innovationen sein kann. Ein Prozess-Repository, das Prozessfragmente und vergangene Fälle enthält, kann den Benutzern helfen, die nächsten Schritte zu verstehen [14].

6 Konzept

In diesem Kapitel wird auf das Konzept der implementierten Prototyp-Visualisierung des Person-Centric Flows und auch auf seine Bestandteile – Tasks, deren Attribute und Constraints – näher eingegangen. In diesem Zusammenhang werden im Folgenden die Idee, das Szenario und die verwendete Prozessnotation näher erläutert.

6.1 Idee

Im Rahmen dieser Arbeit wird der Fokus auf die Visualisierung eines Person-Centric Flows in einem Gesundheitswesen-Szenario gesetzt. Die Tasks des Flows werden von einer *einzelnen* Person ausgeführt.
Das Ziel der Visualisierung eines Person-Centric Flows ist eine optimale Unterstützung bei der täglichen Arbeit eines Benutzers. Der Person-Centric Flow enthält die wichtigsten Informationen für die Ausführung der jeweiligen Tasks des Benutzers. In diesem Zusammenhang ist die Visualisierung essentiell und bringt erhebliche Vorteile mit sich, da mit ihrer Hilfe der Benutzer über die Ausführungszustände seiner Tasks durchgängig informiert ist und seine Aufgaben nicht vergisst. Zudem wird dadurch eine bessere Planung der zukünftigen Aufgaben des Benutzers ermöglicht.
Die grundsätzliche Idee besteht darin: Wenn der Benutzer (Client) auf die auf dem mobilen Gerät installierte Anwendung klickt, dann wird die Menge seiner Tasks und die zugehörigen Constraints durch den Task Manager geladen. Wie bereits in Kapitel 2.3 erwähnt, wird der Human Task Manager eingesetzt. Er ist ebenfalls für die Zuordnung der Tasks zu den Benutzern zuständig.
Hier ist aus Benutzersicht zu beachten, dass ein Klick mit der Maus auf die Anwendung als eine Berührung des Touchscreens des Gerätes auf die Anwendung zu interpretieren ist. Darüber hinaus muss auch die folgende Nebenbedingung berücksichtigt werden: Da die Visualisierung auf einem mobilen Gerät lauffähig sein soll, steht nur eine begrenzte Displayfläche zur Darstellung, Rechengeschwindigkeit und Speicher zur Verfügung.

Wie bereits erwähnt, ist im Rahmen dieser Arbeit der Begriff Task gleichbedeutend mit dem Begriff Aktivität.
Der Benutzer muss sowohl die verfügbaren geladenen Tasks beanspruchen, starten, beenden und bearbeiten, als auch sich die Details der Tasks anzeigen lassen können.
Jeder Task hat die folgenden im Task Manager festgesetzten Attribute:

- Name des Tasks: der Name beschreibt kurz das Ziel des Tasks (*Title*);

- Patientenname: Name des Patienten, der der Krankenschwester zugewiesen ist (*Name of the patient*);

- Raum: dieses Attribut gibt den Raum an, in dem sich der Patient befindet (*Room*) (optional);

- Medikamentenname: Name des Medikaments, welches die Krankenschwester einem Patienten überreichen soll (*Name of medicine*) (optional);

- Zustand des Tasks: hier werden vier verschiedene Zustände eines Tasks unterschieden (*State*):

  - **Ready:** der Task ist aktiv und kann ausgeführt werden.

  - **Reserved:** wenn im Task Manager ein Task instanziiert wird, kann es einige mögliche Benutzer geben, die diesen Task ausführen können. Wenn einer der Benutzer ihn ausführen möchte, muss er ihn beanspruchen, damit dieser Task seiner Worklist und seinem Flow hinzugefügt wird. Innerhalb der Anwendung gilt Folgendes: Wenn der Benutzer auf einen sich im Zustand „Ready“ befindenden Task „Task claim“ klickt, ist dann dieser Task im Zustand „Reserved“. D.h. der Benutzer hat den Task beansprucht und kann ihn danach oder zu einem späteren Zeitpunkt ausführen.

  - **InProgress:** der Task, der sich in diesem Zustand befindet, wird gerade ausgeführt. Der Benutzer hat auf einen „reserved“ Task „Task start“ angeklickt. Der Benutzer kann „Task start“ auch auf einen Task, der im Zustand „Ready“ ist, anklicken. Das bedeutet, dass der Benutzer sofort mit der Ausführung dieses Tasks anfängt.

  - **Completed:** der entsprechende Task wurde erfolgreich ausgeführt und beendet. Der Benutzer hat auf einen sich im Zustand „InProgress“ befindenden Task „Task complete“ angeklickt.

Abbildung 6.1 veranschaulicht die Zustandsübergänge eines Tasks [32]. In dieser Arbeit wird die Aufmerksamkeit auf die oben genannten Zustände konzentriert.

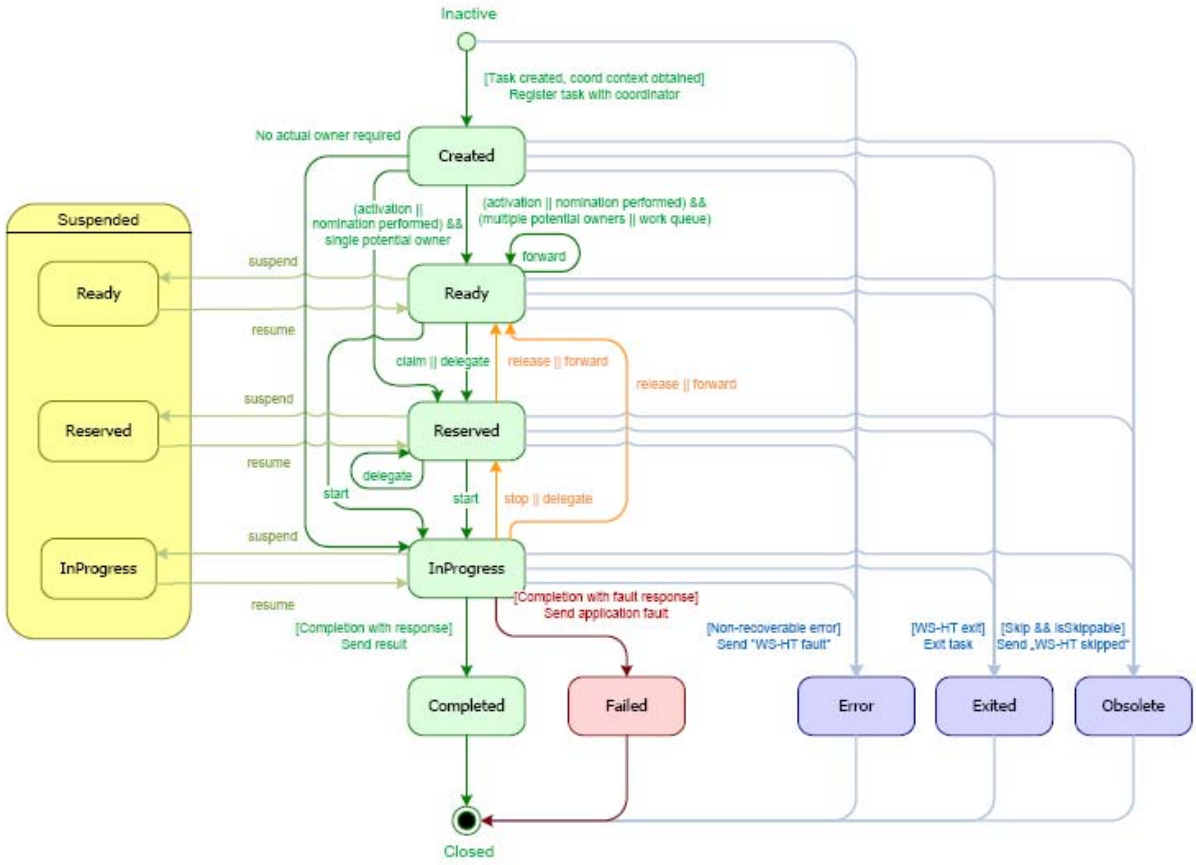

Abbildung 6.1: Zustandsübergänge eines Tasks

Bei der Visualisierung eines Person-Centric Flows eines Benutzers kann der Benutzer auswählen, welchen Task er ausführen möchte. Dabei sind jede zwei Tasks direkt miteinander verbunden, um die bestehenden Constraints bezüglich der Reihenfolge der Tasks zu erfüllen. Wird auf einen Task geklickt, der von einem anderen noch nicht ausgeführten Task abhängt, wird dementsprechend eine Fehlermeldung angezeigt. Wenn der Benutzer beispielsweise auf seinen zweiten Task „Task start" klickt, während sein erster Task noch im Zustand „Ready" ist, wird die Fehlermeldung *„You must first execute task 1 before you start task 2"* auf dem Display erscheinen, wobei 1 und 2 die Namen der Tasks bezeichnen.

Wenn der Benutzer auf den anklickbaren Task länger klickt, kann er unter vier Optionen auswählen: *„Details*", *„Task claim"*, *„Task start"* und *„Task complete"*.
Wird auf „Details" (d.h. die Attribute) geklickt, werden dem Benutzer die Attribute des jeweiligen Tasks angezeigt.
Die Optionen „Task claim", „Task start" und „Task complete" sind Zustandsoperationen, mit denen einzelne Human Tasks gesteuert werden können. Diese Operationen bewirken Veränderungen im Zustand eines Tasks und steuern ihn somit durch seinen Lebenszyklus (s. Abbildung 6.1).
Wenn der Benutzer auf einen Task „Task claim" klickt, wird der Zustand dieses Tasks in „Reserved" und dementsprechend seine Farbe in Orange geändert. Wählt er „Task start" aus, ändert sich der aktuelle Zustand des jeweiligen Tasks in „InProgress" und die Farbe in Grün. Möchte der Benutzer einen Task beenden, klickt er auf „Task complete" und der Zustand dieses Tasks wird in „Completed" und entspre-

chend in Lila angezeigt. Details in Bezug auf die Hintergrundfarbe eines Tasks sind in Kapitel 6.3 angegeben.

Innerhalb des Task Managers gilt im Allgemeinen: wenn die Krankenschwester auf einen Task „Task claim“ klickt, verbindet sich die Anwendung mit dem Server und es wird überprüft, ob dieser Task ausgewählt oder beansprucht bzw. angefordert werden kann, und wenn ja – wird er als „Ready“ oder „Reserved“ markiert.
Der Zustand eines Tasks kann in “Reserved” geändert werden, nur wenn die Anwendung mit dem Task Manager verbunden ist, da ansonsten zwei Krankenschwestern denselben Task haben könnten.
In dieser Arbeit wird folgende Annahme getroffen: Jeder Task wird im Task Manager *eindeutig* einer Krankenschwester zugewiesen. Damit werden in der Visualisierung eines Person-Centric Flows genau die Tasks einer bestimmten Krankenschwester angezeigt, die der Task Manager liefert, abgesehen davon, in welchem Zustand sie sind.
Zur Laufzeit können neue Tasks zu dem Person-Centric Flow bzw. zu der To-Do-Liste der Krankenschwester hinzugefügt werden.

Wenn der Zustand eines Tasks geändert wird, ändert sich dadurch auch die entsprechende Farbe des Tasks. Bei einer Änderung des Zustandes muss sofort eine Synchronisierung stattfinden und damit ein Refresh der Visualisierung ausgelöst werden. Dadurch wird erreicht, dass selbst bei einem eventuellen bzw. unerwarteten Abbau der Verbindung zwischen der Anwendung und dem Task Manager, beim nächsten Verbindungsaufbau die zum Zeitpunkt des Verbindungsabbaus aktuellen Zustände der Tasks eines Benutzers geladen werden. Es besteht also eine dauerhafte Verbindung zwischen der Anwendung und dem Task Manager. Diese Verbindung muss auf jeden Fall bestehen bleiben, wenn der Benutzer den Zustand eines seiner Tasks ändert.

Der Benutzer sieht auf dem Display auch einen Refresh-Button, mit dessen Hilfe er die angezeigte Darstellung seines Flows aktualisieren kann.
Generell gibt es zwei Möglichkeiten, eine Aktualisierung der Visualisierung zu realisieren. Die eine Möglichkeit ist, wie bereits erwähnt, mithilfe eines Refresh-Buttons, der durch den Benutzer angeklickt werden kann.
Die zweite Möglichkeit ist ein Autorefresh. Dabei wird die Visualisierung, wie der Name auch andeutet, automatisch nach einem bestimmten Zeitintervall aktualisiert.

Die Constraints stellen die Reihenfolge der auszuführenden Tasks dar. Die vom Task Manager geladenen Tasks werden in einer Reihenfolge dargestellt, die semantisch inkonsistent sein kann. Beispielsweise kann ein Task, der im Zustand „InProgress“ ist, vor einem Task, der sich im Zustand „Reserved“ befindet, dargestellt sein. Dies wird hier nicht als Fehler wahrgenommen, da dadurch der Benutzer in der Lage sein wird, sich besser an seinen Aufgaben zu orientieren, welche er als Nächstes ausführen soll.

Nachdem der Benutzer seine Tätigkeit ausgeführt hat, gelangt er wieder in die ursprüngliche Darstellung der Visualisierung.

Eine andere Funktion, die hier berücksichtigt werden muss, ist die sogenannte History-Funktion. Dabei kann festgelegt werden, ob und wenn ja – welche (d.h. in welchem Zustand) und wieviele Tasks auf dem mobilen Gerät gespeichert werden. Als

Beispiel für eine History-Funktion kann man gmail auf einem mobilen Telefon angeben. Z.B. hat der Inbox eine Größe von 1 GB, aber das Mobiltelefon speichert nur die Emails seit der letzten Woche. Es gibt dafür eine Telefoneinstellung, bei der man angeben kann, wieviele Tage die Emails gespeichert werden müssen.
Wenn diese Funktion in Bezug auf die Arbeit einer Krankenschwester betrachtet wird, kann die Krankenschwester beispielsweise in den Einstellungen des mobilen Geräts angeben, dass ihre letzten 100 ausgeführten bzw. abgeschlossenen Tasks (d.h. im Zustand „Completed“) gespeichert werden sollen. Sie könnte auch als eine Einstellung angeben, dass alle ihren ausgeführten Tasks gespeichert werden oder alle ihren Tasks, egal in welchem Zustand sie sind.

Eine weitere getroffene Annahme im Rahmen dieser Arbeit ist, dass der Benutzer direkt auf ein Ereignis reagiert und auf dem Objekt navigiert, indem er direkt den Touchscreen berührt.

Es existieren neben den Zustandsoperationen, also Operationen, die zu einem Zustandswechsel in einem Human Task führen, auch Taskeigenschaftsoperationen, sowie Struktur- und Navigationsoperationen.
Durch die Taskeigenschaftsoperationen wird der Zugriff auf die Eigenschaften eines Tasks ermöglicht. Man kann diese Operationen grob in Zuweisungs-, Lösch- und Leseoperationen unterteilen.
Die Strukturoperationen machen das Strukturieren der Human Tasks und den Zugriff auf die Tasks innerhalb einer Struktur ebenso wie den Zugriff auf Daten, die mit der Strukturierung zusammenhängen, möglich.
Bestimmte Tasks können mit Hilfe der Navigationsoperationen anhand ihrer Position innerhalb einer Taskstruktur ausgewählt werden.
Weitere Informationen über die Operationen auf Tasks können [18], [36] entnommen werden.

6.2 Szenario

Um zu zeigen, dass das Konzept der flow-basierten pervasiven Anwendungen universell einsetzbar ist, wird ein Anwendungsszenario im Folgenden skizziert.

Das hier vorgestellte Szanario ist im Bereich des Gesundheitswesens angesiedelt.

Im Hinblick auf ein Pflegeheim müssen mehrere Flows regelmäßig ausgeführt werden. Flows wie „Überreichen von Medikamenten“, „Bestellung von Arzneimitteln“, „Erstellung von Dokumentation“, „Beobachtung der Patienten“, „Wechsel der Bettwäsche“, „Unterstützung bei der Hygiene“, „Messen der Vitalparameter“, „Wechsel der Bandagen“ uvm. sind von essentieller Bedeutung für ein erfolgreich funktionierendes Pflegeheim.
Doch nicht nur die Bereitstellung von täglichen Dienstleistungen ist für die Bewohner von Pflegeheimen wichtig, sondern auch die zwischenmenschliche Betreuung und Tasks sind essentiell. Das alles erfordert Zeit, die oft von administrativen und organisatorischen Tasks verbraucht wird.
Aus diesem Grund besteht eine gewisse Notwendigkeit zur Verbesserung der täglichen Workflows.

Grundsätzlich werden alle Tasks in der Patientenversorgung in zwischenmenschlicher Interaktion zwischen der Krankenschwester und den Patienten durchgeführt, die dieser bestimmten Krankenschwester zugeordnet sind.
Ein Person-Centric Flow einer Krankenschwester besteht hauptsächlich aus den folgenden Tasks:

- **Medication:** tägliche Verabreichnung von Medikamenten
- **Washing:** Waschen eines Patienten
- **Documentation:** Erstellung von Dokumentation

Wenn zum Beispiel angenommen wird, dass zwei Patienten, die sich in verschiedenen Räumen befinden, einer Krankenschwester zugeordnet sind, könnte diese Krankenschwester zuerst dem einen Patienten die Medikamente verabreichen und ihn waschen und nachher dem anderen Patienten die Medikamente verabreichen und ihn waschen. In einer anderen Situation könnte die Krankenschwester entscheiden, zuerst die Medikamente den beiden Patienten zu verabreichen und sie dann waschen. Das kann als eine Anpassung ihres Flows betrachtet werden.
Das Szenario zeigt, dass ein Person-Centric Flow nicht einfach einem statischen Flow-Modell folgt. Es ist hochdynamisch infolge des Kontextes und der Änderungen der Worklist. Die Krankenschwester gestaltet implizit ihren Flow entsprechend der aktuellen Situation.

Dieses Szenario ist sehr menschenorientiert, mit hohen Anforderungen an die Kontexterkennung sowie die menschliche Interaktion [3], [17].

6.3 Verwendete Prozessnotation

Im Folgenden wird auf die graphische Adaption der Prozessvisualisierung durch Definition und flexible Anwendung einer Prozessnotation näher eingegangen. Alle graphischen Aspekte einer Prozessvisualisierung werden dabei abgedeckt, vom „Aussehen“ der Symbole für die verschiedenen Prozesselemente bis hin zur Beschreibung des Verwendungskontextes dieser Symbole.

Die Tasks eines Benutzers sind das zentrale Element einer Visualisierung eines Person-Centric Flows.
Für die einzelnen Tasks werden bestimmte Attribute in der Visualisierung des Person-Centric Flows benötigt.
Für den *Medication*-Task muss die Krankenschwester aus der Visualisierung erkennen können, welches Medikament sie welchem Patienten überreichen soll und auch wo sich der jeweilige Patient befindet, also *Patientenname*, *Medikamentenname* und *Raum*.
Für den *Washing*-Task muss ersichtlich werden, welchen Patienten die Krankenschwester waschen soll und wo er sich befindet, d.h. *Patientenname* und *Raum*.
Bei dem *Documentation*-Task muss der *Patientenname* und die entsprechenden *überreichten Medikamente* dokumentiert werden. Ferner soll auch der *Raum* in dem Documentation-Task enthalten sein, da es Patienten mit demselben Namen geben könnte.

Die Pfeile zwischen den einzelnen Tasks eines Benutzers stellen den Kontrollfluss graphisch dar.

Abbildung 6.2 veranschaulicht die für die Visualisierung benötigten Daten.

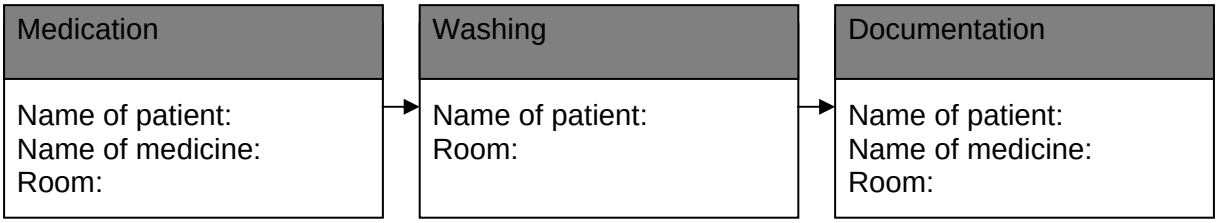

Abbildung 6.2: Daten, die in der Visualisierung benötigt werden

Hintergrundfarbe
Im Rahmen dieser Arbeit wird die folgende Annahme getroffen: Dem aktuellen Zustand eines Tasks wird eine Hintergrundfarbe folgendermaßen zugeordnet:

- **Blau:** Der Task ist aktiv und kann ausgeführt werden.

- **Orange:** Ein orange dargestellter Task bedeutet, dass der Benutzer diesen Task beansprucht hat und ihn ausführen wird. Der Benutzer hat „Task claim“ angeklickt. Der Task ist in seiner To-Do-Liste und ist im Zustand „Reserved“.

- **Grün:** Grün kennzeichnet einen Task, der gerade ausgeführt wird. Der Benutzer hat „Task start“ angeklickt, der Task ist im Zustand „InProgress“.

- **Lila:** Diese Farbe bedeutet, dass der Task schon erfolgreich ausgeführt und beendet wurde. Der Benutzer hat „Task complete“ angeklickt und der Task ist im Zustand „Completed“.

Der Zustand und folglich die Farbe des Tasks kann durch den Benutzer geändert werden.
Diese getroffene Annahme macht die Visualisierung benutzerfreundlicher und hilft dabei dem Benutzer, sich besser am Ausführungszustand seiner Tasks zu orientieren und seine Aufgaben zu erkennen.

7 Implementierung

Nachdem im vorigen Kapitel das Konzept für die Visualisierung des Person-Centric Flows erläutert wurde, wird in diesem Kapitel die Implementierung des Prototyps vorgestellt. Dafür wird im Folgenden zunächst auf den eigentlichen Ablauf und das User Interface, und dann auf die Architektur der prototypischen Visualisierung des Person-Centric Flows als eine Anwendung, deren Schichten und die eingesetzten Technologien eingegangen.
Das Ziel des Prototyps ist es, ungelernte bzw. unerfahrene Krankenschwestern anzuleiten. Der Prototyp wird auch zur Verbesserung der Fehlerfreiheit der Aktivitäts- und Positions-Sensierung und der Kontexterkennung verwendet.

7.1 Ablauf und User Interface

Damit die Anwendung möglichst benutzerfreundlich ist, wurde der folgende Ablauf bzw. das folgende User Interface ausgewählt.
Die Anwendung wird gestartet, indem eine Krankenschwester auf die Anwendung selbst klickt.
Beim Starten der Anwendung sollte zunächst auf dem Display ein Login-Dialog erscheinen, in dem sich die Krankenschwester identifizieren kann. Dazu meldet sich die Krankenschwester mit ihrem Benutzernamen und ihrem Passwort an.
Die folgende Abbildung 7.1 veranschaulicht ein Beispiel für einen Login-Dialog auf einem mit Android betriebenen Mobiltelefon.

Abbildung 7.1: Beispiel für einen Login-Dialog, entnommen aus [35]

Bei erfolgreicher Anmeldung sollte dann die Visualisierung des Person-Centric Flows dieser Krankenschwester auf dem Display ihres Mobiltelefons angezeigt werden. Die Tasks sind in den jeweiligen Farben, entsprechend ihren Zuständen, dargestellt (s. Abbildung 7.2).

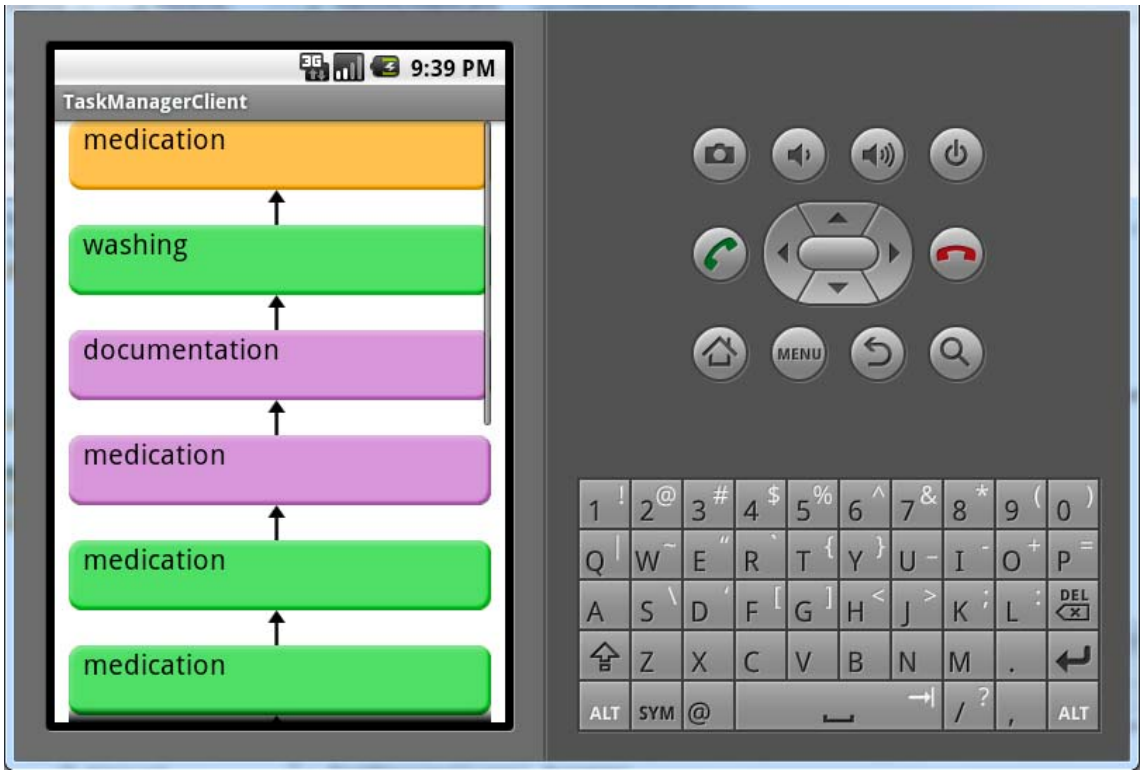

Abbildung 7.2: Beispiel für eine Visualisierung eines Person-Centric Flows einer Krankenschwester

In dieser Ansicht der Tasks, wenn der Benutzer bzw. die Krankenschwester die Taste „Menu“ anklickt, erscheinen 2 Optionen:

1. Die erste Option ist **„Refresh“**; beim Anklick werden die Tasks mit dem Task Manager synchronisiert. (Diese Option ist nur bei einer dauerhaften Verbindung zwischen dem Task Manager und der Anwendung notwendig.)

2. Die zweite Option ist **„Settings“**. Hier werden die Einstellungen des Programms angegeben. Als eine optionale Einstellung kann man z.B. das Zeitintervall bestimmen, in welchem die Visualisierung automatisch synchronisiert werden muss. Anderes Beispiel für eine Einstellung kann eine Legende mit den möglichen Zuständen eines Tasks und den zugehörigen Hintergrundfarben sein, die bei Bedarf als Nachschlagewerk für den Benutzer nützlich sein kann.

Wie bereits im Konzept erläutert, wenn die Krankenschwester auf einen aktiven Task länger klickt, erscheint ein neues Fenster mit den vier Optionen, die mit diesem Task verbunden sind:

1. **„Details“**
2. **„Task claim“**
3. **„Task start“**
4. **„Task complete“**

Abbildung 7.3 zeigt graphisch die vier möglichen Optionen, wenn ein Task angeklickt wird.

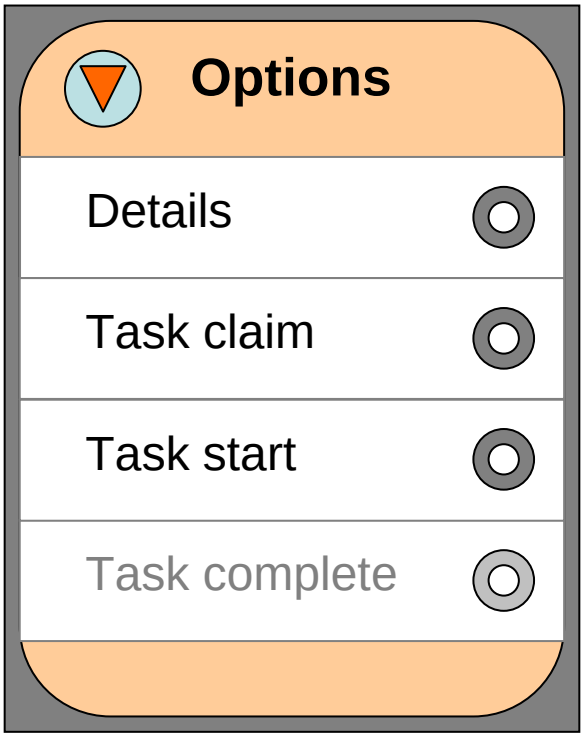

Abbildung 7.3: Mögliche Optionen beim Anklick eines Tasks

Dabei werden einige dieser Optionen ausgeblendet bzw. ausgegraut sein, je nachdem in welchem Zustand der angeklickte Task ist.
Wenn z.B. der Task im Zustand „Ready“ ist, werden zwar beim langen Anklick alle vier Optionen angezeigt, aber die Option „Task complete“ kann nicht ausgewählt werden. Diese Option wird ausgegraut dargestellt.
Ist der Task im Zustand „Reserved“, stehen beim Anklicken des Tasks die Optionen „Details“ oder „Task start“ zur Auswahl, die anderen zwei Optionen „Task claim“ und „Task complete“ sind dagegen ausgegraut und damit nicht klickbar.
Für in Ausführung befindliche Tasks (d.h. im Zustand „InProgress“) kann der Benutzer nur zwei der Optionen – „Details“ und „Task complete“ – auswählen.
Bei abgeschlossenen Tasks (d.h. im Zustand „Completed“) kann nur die Option „Details“ angeklickt werden, um die Details dieser Tasks anzeigen zu lassen (s. Abbildung 7.4).

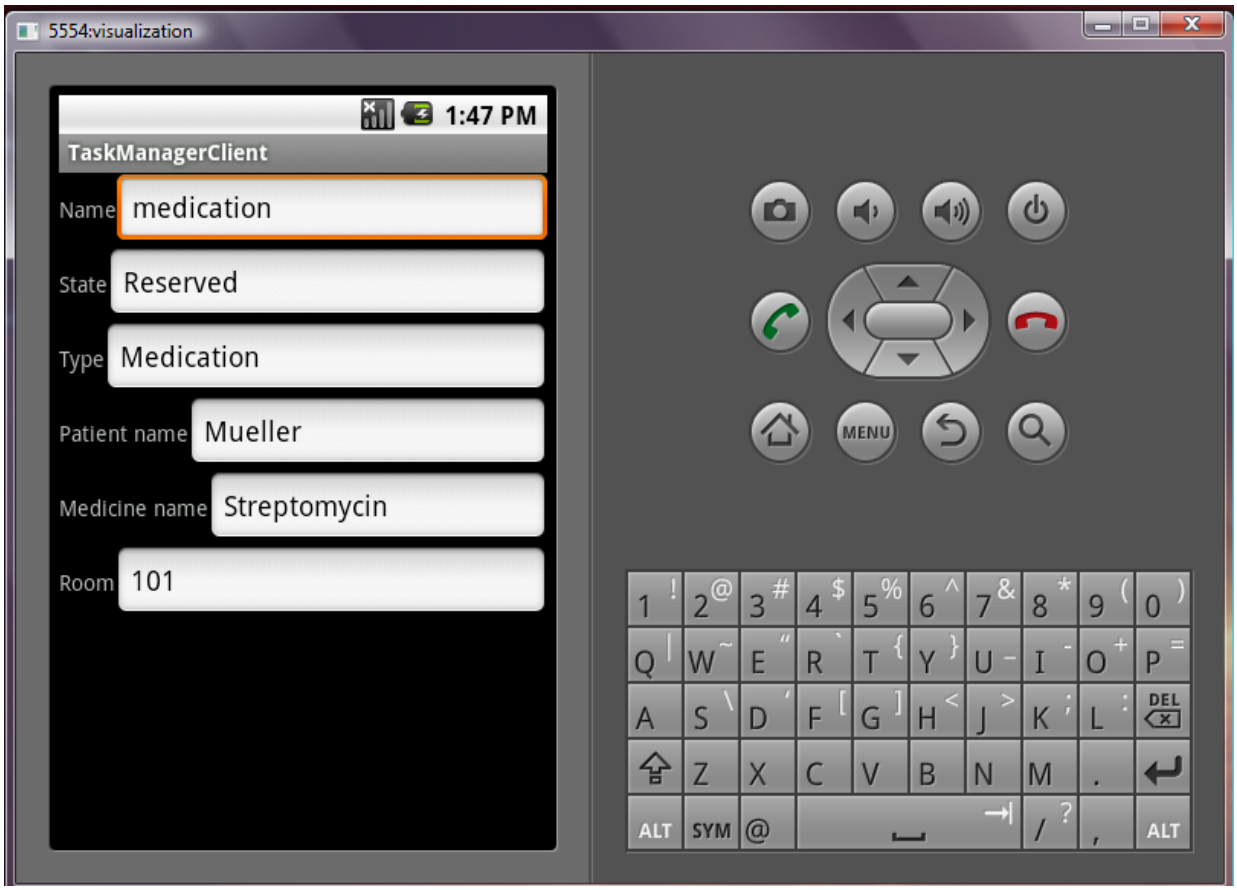

Abbildung 7.4: Ansicht der Details eines Tasks des Person-Centric Flows einer Krankenschwester

Aus Platzgründen auf dem Display wird die Visualisierung des Person-Centric Flows einer Krankenschwester in Form einer Liste von Tasks dargestellt, da ansonsten die Visualisierung weder horizontal noch vertikal auf dem Display passen kann. Wie bereits erwähnt, sind in der Liste nur die Tasks der angemeldeten Krankenschwester zu sehen. Jeden noch nicht beanspruchten Task, der auf dem Display erscheint, muss die Krankenschwester beanspruchen oder sofort starten. Die Tasks werden nacheinander in der Reihenfolge dargestellt, die genau der in einer selbst erstellten XML-Datei (im Task Manager) festgelegten Constraints entspricht.
Der Übersicht halber und damit die angezeigte Visualisierung des Person-Centric Flows ähnlich einem Prozessgraphen dargestellt wird, werden kleine Pfeile zwischen den Tasks dargestellt. Diese Pfeile entsprechen also den konkreten Kontrollflusskanten in einem Prozessgraphen.

Die abgeschlossenen Tasks könnten auf zwei Arten implementiert werden. Die erste ist: sobald die Tasks in Zustand „Completed“ geändert werden, werden sie aus der Liste der Tasks entfernt. Sie lassen sich nicht mehr auf dem Display anzeigen und werden in dem Mobiltelefon gespeichert.
Im Rahmen dieser Arbeit wird die zweite Art implementiert: Die Tasks, die sich im Zustand „Completed“ befinden, können nicht mehr aktiv werden. Deren Zustand kann sich nicht mehr ändern. Sie bleiben jedoch in der Visualisierung des Person-Centric Flows zu sehen. Dies erleichtert die Übersicht aller Tasks für die Krankenschwester. Sie kann auf diese Weise schneller erkennen, welche Tasks sie schon ausgeführt hat und welche sie noch ausführen muss.

Der Person-Centric Flow einer Krankenschwester kann beliebig viele Tasks enthalten. Eine Krankenschwester könnte einige Tasks zum Beispiel im Zustand „InProgress“ oder in einem anderen Zustand haben.

Bezüglich der Verbindung der Anwendung mit den benötigten Daten wird folgende Annahme getroffen: Da zu diesem Zeitpunkt keine Schnittstelle zwischen dem Task Manager und der Anwendung existiert, wird bei der Prototyp-Implementierung offline mit lokalen Daten gearbeitet, die auf einem Wechseldatenträger gespeichert sind. Die Daten, d.h. die Tasks, ihre Attribute und Zustände, sowie die Constraints bezüglich der Reihenfolge der Tasks sind lokal in einer XML-Datei gespeichert.

In Bezug auf die History-Funktion, die im Konzept beschrieben worden ist, treffe ich folgende Annahme: Es werden im Rahmen dieser Arbeit die aktuellen Tasks des Benutzers gespeichert, weil sich diese Funktionalität zu diesem Zeitpunkt nicht implementieren lässt.
Die Tasks werden in einer lokalen Datenbank gespeichert und müssen in regelmäßigen Zeitintervallen gelöscht werden, damit sie nicht soviel Speicherplatz verbrauchen.
Bei einer Änderung des Zustandes eines Tasks findet sofort eine Synchronisation mit der lokalen Datenbank statt, d.h. ein (automatischer) Refresh der Visualisierung wird ausgelöst. Damit sieht der Benutzer die aktualisierten Ausführungszustände seiner Tasks.
Die erstellte Anwendung ist in Englisch implementiert worden.

7.2 Architektur

Im Folgenden wird die Architektur der prototypischen Visualisierung eines Person-Centric Flows als eine Anwendung beschrieben. Dabei wird zunächst auf die abstrakte Architektur mit ihren Schichten und dann auf die konkrete Detailarchitektur näher eingegangen.

7.2.1 Abstrakte Architektur der prototypischen Visualisierung

Man kann die Architektur in drei Schichten abstrakt gliedern: Visualisierungschicht, Kommunikationsschicht und lokale Datenbank.

Die *Visualisierungsschicht* umfasst Methoden zur Darstellung des Person-Centric Flows eines Benutzers auf dem Display seines mobilen Gerätes. Dazu greift diese Schicht auf die lokale Datenbank zu, um die entsprechenden Daten auszulesen.

Die *Kommunikationsschicht* ermöglicht die Kommunikation zwischen der Anwendung und dem Benutzer. Sie stellt die Funktionen, die dem Benutzer beim Anklicken eines Tasks zur Verfügung stehen, bereit. Wenn der Benutzer den Zustand eines Tasks ändert, wird die Visualisierung seines Flows aktualisiert und die Zustände der Tasks werden auch mit der lokalen Datenbank synchronisiert.

In der *lokalen Datenbank* werden die für die Darstellung des Person-Centric Flows benötigten Daten – Tasks, ihre Attribute, Zustände und die zugehörigen Constraints – in Form einer eigenen XML-Datei gespeichert.

Die Java-Plattform unterstützt verschiedene Mechanismen, um mit XML auf dem Android Betriebssystem und Software Developers Kit (SDK) zu arbeiten. Beispielsweise stehen Simple API für XML (SAX) von Java und das Document Object Model (DOM) auf Android zur Verfügung. Nähere Informationen darüber, wie man genau mit XML auf Android arbeiten kann, findet man in [37].

Zum besseren Verständnis der Architektur dient Abbildung 7.5.

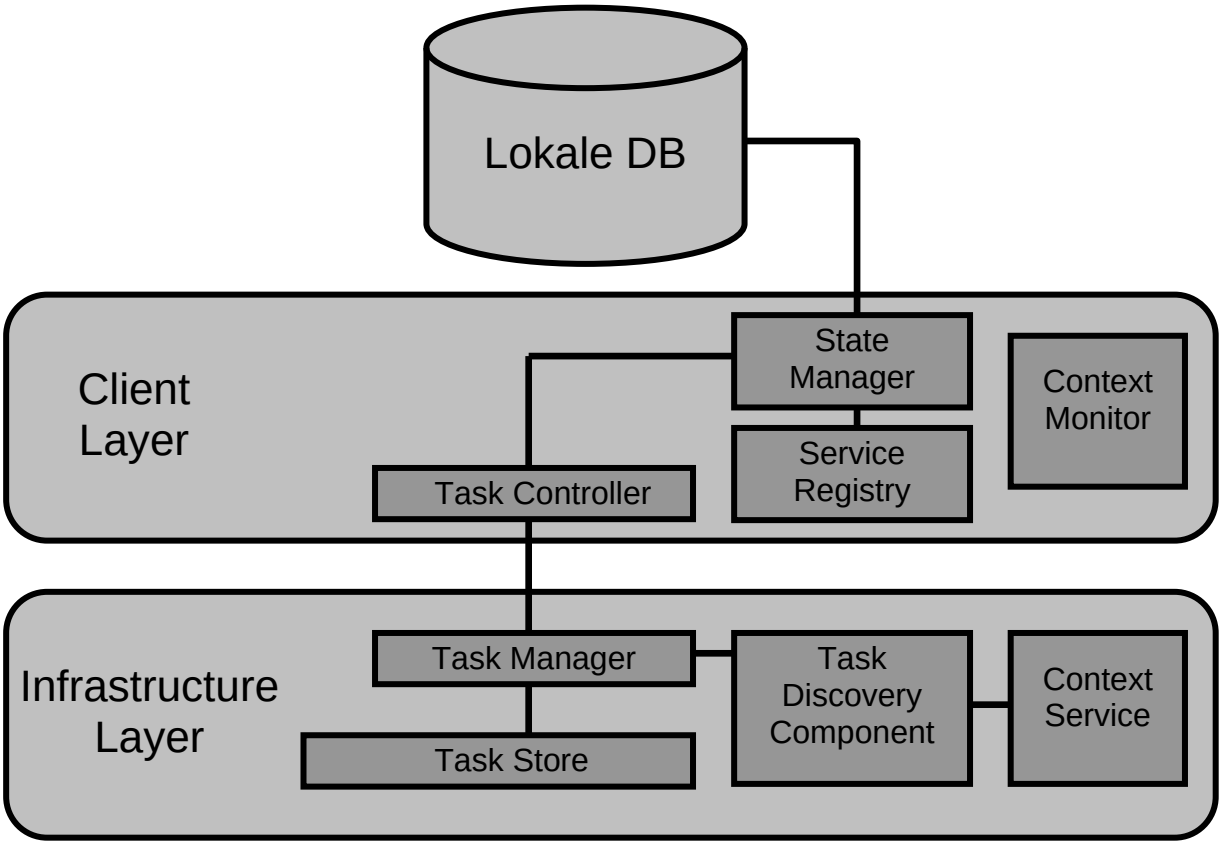

Abbildung 7.5: Abstrakte Architektur der prototypischen Visualisierung

Der „Task Store“ handhabt die Persistenz der Tasks durch die Bereitstellung einer Schnittstelle zum Erstellen, Löschen und Beziehen der Tasks.
Der „Task Manager“ verwaltet das Laufzeitverhalten eines Tasks, indem ermöglicht wird, dass Tasks erstellt, initialisiert, beansprucht, gestartet und abgeschlossen werden.
Der „Task Controller“ ist die Verbindung zwischen dem Client (d.h. einem einzelnen Gerät) und dem Server.
Die „Service Registry“ funktioniert wie eine lokale Service-Entdeckungskomponente auf dem lokalen Client.
Der „State Manager“ handhabt den Zustand für einen Task.
Der „Context Monitor“ ist für die Überwachung des Kontextes auf dem Gerät und seine Verwendung verantwortlich. Die Kontextinformation wird durch den „Context Service“ gesammelt, der mit der „Task Discovery Component“ (TDC) kooperiert. Die TDC hat eine Menge von Regeln, wodurch sie in der Lage ist, entweder die Menge der Kontext-Eregnisse an einen existierenden Task anzupassen, oder neue Tasks zu erstellen, die in einem spezifischen Kontext wichtig scheinen. Der „Context Monitor“, der „Context Service“ und die TDC unterstützen die in dem Task vorliegenden Infor-

mationen. Da der Kontext für diese Arbeit nicht von entscheidender Bedeutung ist, werden diese drei Komponenten im Rahmen dieser Arbeit auch nicht detailliert betrachtet [38], [39].

7.2.2 Konkrete Architektur der prototypischen Visualisierung

Im Folgenden wird die konkrete Implementierung der prototypischen Visualisierung eines Person-Centric Flows als eine Android-Anwendung erläutert.

Die Anwendung kommuniziert mit einem cloud-basierten Dienst und synchronisiert ihre Daten mit den Daten, die lokal in einem Content Provider gespeichert sind. Dafür werden zwei in Beziehung stehende Bestandteile des Android Frameworks – der „Account-Manager“ und der „Synchronization Manager“ (durch einen „Synchronisationsadapter“) – verwendet.

Für die Implementierung des Login-Dialogs könnte der „Account-Manager“ verwendet werden.
Die Klasse „Account-Manager“ bietet Zugriff auf ein zentralisiertes Register der Online-Konten der Benutzer und ermöglicht die gemeinsame Nutzung von Anmeldeinformationen über mehrere Anwendungen und Dienste. Benutzer geben die Anmeldeinformationen (Benutzername und Passwort) für jedes Konto nur einmal ein – Anwendungen mit der USE_CREDENTIALS-Erlaubnis können dann den „Account-Manager“ zum Erhalten eines auth-Tokens (Authentifizierungs-Kurznamen) für das Konto abfragen. Ein Authentifikator (eine Zusatzkomponente von „Account-Manager“) fordert die Anmeldeinformationen des Benutzers an, validiert sie mit einem Authentifizierungs-Server, der in dem Cloud läuft, und speichert sie dann auf dem „Account-Manager“. Der Authentifikator für einen Dienst wird durch die Erweiterung der neuen abstrakten „AbstractAccountAuthenticator“-Klasse implementiert.

Der „Synchronisationsadapter“, eine grundlegende Komponente des Synchronisationsdienstes, teilt dem Synchronisations-Manager den Accounttyp und die Content Provider-Berechtigung mit. Eigene Synchronisierungsadapter können durch die Erweiterung der abstrakten „AbstractThreadedSyncAdapter“-Klasse und die Implementierung der „onPerformSync()“–Methode geschrieben werden. Diese Methode wird aufgerufen, immer wenn der Synchronisations-Manager eine Synchronisierungsoperation für diesen Synchronisationsadapter erteilt [35].

Zur Visualisierung des Person-Centric Flows bzw. der Daten in einer Liste auf dem Display des mit Android betriebenen Mobiltelefons wird die Funktionalität der Klasse „ListView“ verwendet. Das ist eine Ansicht, die die Elemente in einer vertikalen scrollbaren Liste zeigt. Die Elemente kommen von dem „ListAdapter“, der mit dieser Ansicht verbunden ist. D.h. die Listenelemente werden automatisch in die Liste unter Verwendung des „ListAdapters“ eingefügt.
Das Interface „ListAdapter“ ist ein erweiteter „Adapter“, der die Verbindung zwischen einer „ListView“ und den Daten ist, und der die Liste unterstützt. Die „ListView“ zeigt die zur Verfügung gestellten Daten, die von einem „ListAdapter“ umhüllt sind. Zum Zeichnen der Pfeile zwischen den Elementen in der Liste, d.h. zwischen den Tasks des Person-Centric Flows, werden die Methoden von „ListView“ – „getDevider()“ und „setDevider()“ – verwendet.

Ein „Adapter“-Objekt selbst dient als eine Verbindung zwischen einer „AdapterView“ und den zugrundeliegenden Daten für diese View. Das Interface „Adapter“ ermöglicht Zugriff auf die Datenelemente und ist verantwortlich für die Erstellung einer „View“ für jedes Element in der Datenmenge. Eine „AdapterView“ ist eine Klasse und eine View, deren Kinder durch einen „Adapter“ bestimmt werden. Die Klasse „View“ stellt den Grundbaustein für die Komponenten der Benutzeroberfläche dar und ist für das Zeichnen und das Event-Handling (Ereignisbehandlung) zuständig. „View“ ist die Basisklasse für Widgets, die zur Erstellung von interaktiven User Interface-Komponenten (Buttons, Textfelder, etc.) verwendet werden.
Weitere Informationen über diese Klassen, Interfaces und Methoden sind in [40], [41] zu finden.

Wie bereits erwähnt, dient die selbst erstellte XML-Datei zum Auslesen der für die Visualisierung des Person-Centric Flows benötigten Daten. Sie wird für die Prototyp-Implementierung auf einem Wechseldatenträger gespeichert, da eine Verbindung zum Task Manager zu diesem Zeitpunkt nicht realisierbar ist. Alle Daten, die in der XML-Datei enthalten sind, werden auf das Mobiltelefon der Krankenschwester geladen und angezeigt. Generell besteht die XML-Datei aus Tasks, deren Attributen und Constraints. Aufgrund der fehlenden Verbindung zum Task Manager existiert kein Online-System zur Verifikation der Anmeldedaten der Benutzer.
Beim Starten der Anwendung wird die XML-Datei ausgelesen und es werden Objekte generiert. Diese Objekte werden an die Liste, die die Tasks visualisieren wird, angebunden. Wenn ein bestimmter Task modifiziert wird, wird auch das entsprechende Objekt modifiziert. Nach der Modifikation werden die Daten aus dem Objekt zurück in die XML-Datei geschrieben und gespeichert. Somit ist beim nächsten Starten der Anwendung eine aktualisierte Visualisierung des Person-Centric Flows gewährleistet und der Benutzer sieht die aktuellen Zustände seiner Tasks.

Man könnte Dateien direkt auf dem internen Speicher des Geräts speichern. Standardmäßig sind Dateien, die auf dem internen Speicher gespeichert sind, nur für die Anwendung zugänglich und damit nichtöffentlich und weder andere Anwendungen noch der Benutzer können auf sie zugreifen. Wenn aber der Benutzer die Anwendung deinstalliert, werden diese Dateien entfernt. Aus diesem Grund wurde die Entscheidung getroffen, im Rahmen dieser Arbeit einen externen Speicher für die Speicherung der XML-Datei zu verwenden. Damit kann die XML-Datei manuell auf einen Wechseldatenträger kopiert werden, und die Anwendung wird nach der Datei immer an derselben Stelle suchen.
Jedes Android-kompatible Gerät unterstützt einen gemeinsamen "externen Speicher", den man verwenden kann, um Dateien zu speichern. Dies kann ein Wechseldatenträger (wie z.B. eine SD-Karte) oder ein interner (nicht entfernbarer) Speicher sein. Dateien, die auf dem externen Speicher gespeichert sind, sind für alle lesbar und können durch die Benutzer geändert werden, wenn sie einen USB-Massenspeicher zur Übertragung von Dateien auf einem Computer aktivieren [42].

Abbildung 7.6 zeigt einen Auszug aus der XML-Datei. Jeder Task des Person-Centric Flows hat eine eindeutige ID. Die Constraints bezüglich der Reihenfolge der Tasks entsprechen einem Prozessgraphen, wobei „src“ die ID des betreffenden Tasks bezeichnet und „target“ auf die ID des nachfolgenden Tasks in dem Person-Centric Flow verweist.

```
<person-centric-flow>
    <tasks>
        <task>
          <id>1</id>
          <name>medication</name>
          <state>Ready</state>
          <type>medication_task</type>
          <patient_name>Mueller</patient_name>
          <medicine_name>Streptomycin</medicine_name>
          <room>101</room>
        </task>
        <task>
          <id>2</id>
          <name>washing</name>
          <state>Ready</state>
          <type>washing_task</type>
          <patient_name>Mueller</patient_name>
          <room>101</room>
        </task>
        <task>
          <id>3</id>
          <name>documentation</name>
          <state>Ready</state>
          <type>documentation_task</type>
          <patient_name>Mueller</patient_name>
          <medicine_name>Paracetamol</medicine_name>
          <room>101</room>
        </task>
        .....
        .....
        <task>
          <id>12/id>
          <name>medication</name>
          <state>InProgress</state>
          <type>medication_task</type>
          <patient_name>Wagner</patient_name>
          <medicine_name>Penicillin</medicine_name>
          <room>204</room>
        </task>
    </tasks>
    <constraints>
        <link src="1" target="2" />
        <link src="2" target="3" />
        <link src="6" target="7" />
        <link src="9" target="10" />
    </constraints>
</person-centric-flow>
```

Abbildung 7.6: Auszug aus der XML-Datei

7.3 Technologien

Für die Implementierung der Visualisierung des Person-Centric Flows werden verschiedene Technologien eingesetzt, die alle öffentlich verfügbar sind.
Die Implementierung erfolgte in Eclipse 3.5 (Galileo)[1] als Entwicklungsumgebung für Java EE (Enterprise Edition) Developers mit Java 5.0[2] als Programmierungssprache.
Wie bereits schon erwähnt, soll die Visualisierung des Person-Centric Flows auf einem mobilen Gerät lauffähig sein. Aus diesem Grund wird Android eingesetzt. Android ist eine von Google entwickelte freie Software und eine Open Source-Plattform für mobile Geräte, die ein Betriebssystem, Middleware und die wichtigsten Anwendungen umfasst. Die Architektur von Android baut auf dem Linux-Kernel 2.6 auf.
Android SDK[3] ist eine Umgebung zum Entwickeln javabasierter Applikationen für die Mobilplattform Android. Es bietet die notwendigen Tools und Bibliotheken zur Entwicklung von Anwendungen, die auf Android-basierten Geräten laufen. Android SDK enthält die Dalvik Virtual Machine (VM) zum Emulieren von mobilen Geräten und ein Debugging-System. Detaillierte Informationen zum Android sind in [34] zu finden.

[1] www.eclipse.org
[2] www.java.com
[3] http://developer.android.com/index.html

8 Zusammenfassung und Ausblick

Aufgabe dieser Diplomarbeit ist es, die Visualisierung von Person-Centric Flows zu beschreiben und ein konkretes Szenario der Visualisierung prototypisch zu realisieren. Dabei soll die Implementierung auf einem mit Android betriebenen mobilen Gerät eines Benutzers umgesetzt werden. Ferner soll berücksichtigt werden, dass nur eine begrenzte Displayfläche, Speicher und Rechengeschwindigkeit zur Verfügung stehen.
In diesem Kapitel werden die wichtigsten Themenfelder, die im Rahmen dieser Diplomarbeit behandelt wurden, zusammengefasst. Weiterhin wird ein Ausblick über weitere Verbesserungen, die mit dem Paradigma der Person-Centric Flows zusammenhängen, gegeben.
In Kapitel 1 der Diplomarbeit wird das Konzept eines Person-Centric Flows, der von einer einzelnen Person ausgeführt wird, eingeführt sowie auf die Motivation für die Implementierung der Visualisierung von Person-Centric Flows eingegangen. Des Weiteren werden Forschungsfragestellungen hinsichtlich der Anpassung der Visualisierung an die Bedürfnisse der Benutzer dargestellt.
In Kapitel 2 werden Person-Centric Flows formal definiert sowie die Begriffe Tasks, Aktivitäten und Constraints als deren Bestandteile erläutert. Darüber hinaus wird der Task Manager vorgestellt, der die verteilte Arbeit in Tasks organisiert und für die Zuordnung der Tasks zu den Benutzern zuständig ist. Person-Centric Flows zeichnen sich dadurch aus, dass sie in der realen Welt angesiedelt sind, indem sie an Entitäten wie Artefakte und Menschen angefügt werden und sich mit ihnen durch verschiedene Kontexte bewegen. Außerdem sind sie kontextorientiert. Sie sind also in der Lage, den Kontext bezüglich der aktuellen Umgebung und auch der Aktivitäten ihrer Entitäten zu erkennen. Eine andere Eigenschaft von Person-Centric Flows ist, dass sie sich anpassen können, um die Änderungen in der realen Umgebung widerzuspiegeln.
Kapitel 3 beschreibt zwei Beispiele für den Einsatz der Person-Centric Flows in der Praxis – im Bereich des Gesundheitswesens (Krankenhäuser) und innerhalb einer Unternehmensstruktur. Durch diese Einsetzbarkeit in der Realität wird die Notwendigkeit und der Nutzen von Flows besonders hervorgehoben, was insbesondere die wesentliche Entlastung des Personals angeht. Ferner werden hier zwei Möglichkeiten für die Verteilung von Anleitungsinformationen in einer realen Umgebung dargestellt: Aktivitäts-Erkennung und direkte Navigation. Bei der Aktivitäts-Erkennung oder auch „Activity Recognition“ genannt, werden Sensoren, Organizer, Projektoren und tragbare Geräte verwendet, um die Flows und die Task-Informationen in der umliegenden realen Arbeitsumgebung zu erkennen. Damit werden die Benutzer mit wichtigen Informationen bezüglich ihrer Arbeit versorgt. Es existieren in diesem Zusammenhang mobile, in die reale Umgebung eingebettete Anzeigeschnittstellen zur Anzeige der Task-Informationen sowie Ansteuerungsstrategien. Diese Strategien dienen der Entscheidung, welche Information zugänglich sein soll und wann, wo und wie sie in der Umgebung darzustellen ist. Bei der direkten Navigation dagegen re-

agiert der Benutzer direkt auf ein Ereignis und navigiert direkt auf dem Objekt, indem er über Eingabefelder wie z.B. Tastatur oder Touchscreen in den Prozess eingreift. Die Interaktion mit dem Benutzer erfolgt also durch sein mobiles Gerät, das er mit sich trägt.
Definitionen des Begriffs Visualisierung werden in Kapitel 4 eingeführt. Hier werden zwei Möglichkeiten betrachtet, die die Anpassung der Visualisierung an die Bedürfnisse der Benutzer behandeln. Zum einen wird die graphische Anpassung dargestellt, bei der die Notation durch Form- und Farbveränderungen der Symbole sowie das Layout angepasst werden können, ohne dass dabei inhaltlich etwas geändert wird. Zum anderen wird die strukturelle Anpassung der Visualisierung beschrieben. Dabei kann die Menge der dargestellten Information durch sogenannte Prozess-Views reduziert werden. Eine andere Möglichkeit, die Struktur der Darstellung von Prozessen zu verändern, bieten die Darstellungsformen der Informationen. Bei den verschiedenen Darstellungsformen wie z.B. Prozessgraphen, Swimlanes, Tabellen, Gantt- und Interaktionsdiagrammen werden dieselben Informationen verwendet, aber auf eine andere Art aufbereitet. In dieser Arbeit wurde die Aufmerksamkeit auf die Prozessgraphen als Darstellungsform eines Person-Centric Flows gerichtet.
Durch diese Freiheitsgrade der Gestaltung einer Visualisierung kann die Visualisierung auf die Bedürfnisse des Betrachters zugeschnitten werden.
Jede Visualisierung setzt bestimmte Daten voraus, deswegen wird den Daten eine besondere Rolle im gesamten Visualisierungsprozess zugewiesen.
Des Weiteren wird in diesem Kapitel auf den Ablauf der Visualisierung von Daten insbesondere in der Domäne der Prozessvisualisierung eingegangen. Der Visualisierungsprozess besteht aus vier Schritten, die sich in einer sogenannten Visualisierungspipeline anordnen lassen – Datenbeschaffung, Datenaufbereitung, Abbildung auf Darstellungselemente und Erzeugung der Darstellung (Rendering).
Kapitel 5 umfasst die Anforderungsanalyse, die sowohl die Anforderungen an eine graphische Konfiguration als auch an die Inhalte von Prozessvisualisierungen einbezieht. Bei den Anforderungen an die Konfiguration kann man zwischen funktionalen und nicht-funktionalen Anforderungen unterscheiden. Die funktionalen Anforderungen wie beispielsweise „Personalisierung“ oder „Einblendung von Detailinformationen“ betrachten die funktionalen Eigenschaften einer Prozessnotation, um möglichst hilfreiche Prozessvisualisierungen zu erzeugen. Die nicht-funktionalen Anforderungen wie zum Beispiel „Skalierbarkeit der Graphiken“ oder „Verwendung von standardisierten Formaten“ sind für die Einsetzbarkeit einer Visualisierungskomponente in der Realität wichtig.
Die Anforderungen an die Inhalte von Prozessvisualisierungen lassen sich auch in funktionale und nicht-funktionale Anforderungen gliedern. Die funktionalen Anforderungen beschreiben die Funktionen der Visualisierungskomponente, die direkt erlebt werden können. Es gibt einerseits allgemeine Anforderungen wie beispielsweise „Visualisierung von systemübergreifenden Prozessen“ und „Einbindung von Applikationsdaten in die Darstellung“ und andererseits spezifische Anforderungen, die die Personalisierung betreffen, wie z.B. „Flexible Definition der Prozessnotation“ und „Darstellung von Prozessinformation in verschiedenen Formen“.
Die nicht-funktionalen Anforderungen an die Inhalte von Prozessvisualisierungen spiegeln die allgemeine Beschaffenheit der Anwendung wider und sind für die Akzeptanz der Visualisierungskomponente relevant. Als solche Anforderungen kann man „Zugriffsbeschränkung“, „Benutzerfreundliche Gestaltung der Anwendung“ und „Sicherheit“ angeben.
Kapitel 6 stellt das Konzept der implementierten prototypischen Visualisierung des Person-Centric Flows eines einzelnen Benutzers in einem Gesundheitswesen-Sze-

nario vor. In diesem Kontext wird auf die Tasks und deren Attribute – Name des Tasks, Patientenname, Raum, Medikamentenname und Zustand des Tasks – als Bestandteile eines Person-Centric Flows eingegangen. Hier werden insbesondere die vier Zustände eines Tasks – „Ready“, „Reserved“, „InProgress“ und „Completed“ – berücksichtigt. Beim Anklick eines Tasks stehen dem Benutzer vier Optionen zur Verfügung – „Details“, „Task claim“, „Task start“ und „Task complete“. Bei einer Auswahl von „Details“ kann sich der Benutzer die Details eines Tasks (d.h. seine Attribute) anzeigen lassen. Die Option „Task claim“ ermöglicht es, einen Task zu beanspruchen. Mit der Option „Task start“ startet der Benutzer die Ausführung eines Tasks. Durch die Option „Task complete“ wird die Ausführung eines Tasks nach dessen erfolgreichem Abschluss beendet.
Als wichtiger Bestandteil eines Person-Centric Flows werden auch die Constraints beschrieben, die die Ausführungsreihenfolge der Tasks darstellen. Diese Reihenfolge kann semantische Inkonsistenz aufweisen.
Das menschenorientierte Anwendungsszenario ist im Bereich des Gesundheitswesens angesiedelt und als Benutzer wird eine Krankenschwester festgelegt, deren Person-Centric Flow zu visualisieren ist. In diesem Zusammenhang werden als Tasks „Medication“, „Washing“ und „Documentation“ bestimmt.
Behandelt werden hier auch Funktionalitäten wie „Refresh“, d.h. Aktualisierung der Visualisierung des Person-Centric Flows zur Laufzeit und „History-Funktion“, mit deren Hilfe die Speicherung der Tasks auf dem mobilen Gerät realisiert wird.
In diesem Kapitel wird auch auf die verwendete Prozessnotation eingegangen. Es werden hier die Attribute, die für die einzelnen Tasks in der Visualisierung notwendig sind, sowie die Hintergrundfarbe der Tasks, die sich bei einer Zustandsänderung eines Tasks auch ändert, beschrieben.
Kapitel 7 widmet sich der Implementierungphase und somit dem praktischen Teil dieser Diplomarbeit. Es werden diesbezüglich der Ablauf und das User Interface im Detail beschrieben. Ferner wird hier zum einen die abstrakte Architektur der prototypischen Visualisierung des Person-Centric Flows als eine Android-Anwendung, die sich grob in Visualisierungs-, Kommunikationsschicht und lokale Datenbank gliedern lässt, erläutert. Zum anderen wird auch auf die konkrete Architektur und die bei der Implementierung eingesetzten Technologien eingegangen.
Zusammenfassend kann man sagen, dass das Hauptziel von Person-Centric Flows die Unterstützung einer Person bei der Ausführung ihrer Tasks ist, z.B. durch das Reduzieren der kognitiven Belastung, die notwendig für die Planung ihrer Tasks ist.

Ausblick

Durch die angemessene Visualisierung von Person-Centric Flows werden diese Flows verständlicher für die Benutzer. Die Visualierung bringt viele Vorteile und eine benutzerfreundliche Unterstützung bei der Ausführung der Tasks durch ihre bessere Erkennung. Es gibt dennoch auch Verbesserungen, die hinsichtlich der Weiterentwicklung von Person-Centric Flows wünschenswert sind.
Person-Centric Flows haben kein vordefiniertes Flow-Modell. Tasks erscheinen und verschwinden ständig. Aus diesem Grund müssen die Person-Centric Flows sehr häufig angepasst werden. Folglich muss eine Sprache gefunden werden, die mit diesem hohen Grad an Flexibilität umgehen kann. Zudem muss die Auffassung von einer Instanz neu überdacht werden. Eine Instanz eines allgemeinen Flow-Modells hat einen definierten Anfang und ein definiertes Ende. Im Vergleich zu traditionellen Workflows hat ein Person-Centric Flow keinen definierten Anfang und kein definier-

tes Ende. Menschen führen ständig Tasks aus, obwohl es im Flow auch Lücken geben könnte, wo keine Tasks ausgeführt werden.
Des Weiteren sind geeignete Voraussagealgorithmen zur Bestimmung der Reihenfolge der Tasks notwendig, da der Person-Centric Flow, den eine Person im Kopf hat, nicht erfasst werden kann. Die Menschen teilen Tasks in eine Menge von Unter-Tasks ein (Granularität von Tasks). Grund dafür ist, dass die Tasks in Geschäftsprozessen oft auf einer höheren Ebene modelliert sind als sie tatsächlich von den Menschen ausgeführt werden. Daher sollen die Algorithmen zur Voraussage der Person-Centric Flows die Tatsache berücksichtigen, dass die Menschen ihre Tasks umstrukturieren.
Darüber hinaus muss diskutiert werden, ob es sinnvoll ist, Parallelität in Person-Centric Flows zuzulassen. In Bezug auf die Granularität der Tasks würde das Verbieten von Parallelität bedeuten, dass Tasks in Unter-Tasks eingeteilt werden, wenn sie überschneidend ausgeführt werden.
Der Datenschutzaspekt ist ebenfalls ein sehr wichtiges Thema, da in den Person-Centric Flows viele Informationen über das spezische Benehmen einer Person enthalten sind, die vertraulich sind und deswegen geschützt werden sollen. Dafür sind entsprechende Sicherheitsmechanismen zu entwickeln: es müssen Einschränkungen für verschiedene Granularitäten angewandt werden, von der Beschränkung des Zugriffs auf den gesamten Prozess, über spezielle Typen von Elementen oder Attributen, die nicht angezeigt werden sollen, bis zur Beschränkung des Zugriffs auf eine spezifische Ebene von Details.

Literaturverzeichnis

[1] Kortuem, G., Kawsar, F., Al Takrouri, B.: Flow-Driven Ambient Guidance, Eighth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom 2010), Mannheim, Deutschland, März 29-April 02, 2010

[2] Leymann, F., Unger, T., Wagner, S.: On Designing a People-oriented Constraint-based Workflow Language, Proceedings of the 2nd Central-European Workshop on Services and their Composition, ZEUS 2010, Berlin, Deutschland, Februar 25-26, 2010

[3] Unger, T., Eberle, H., Leymann, F.: Research Challenges on Person-centric Flows, Proceedings of the 2nd Central-European Workshop on Services and their Composition, ZEUS 2010, Berlin, Deutschland, Februar 25-26, 2010

[4] Zhou, J.: Workflow Driven Visualization, Diplomarbeit-Nr.: 2912, Universtität Stuttgart, Institut für Architektur von Anwendungssystemen, Deutschland, 2009

[5] McCormick, B. H., DeFanti, T. A., Brown, M. D.: Visualization in Scientific Computing, ACM SIGGRAPH Computer Graphics, Vol. 21(6), November 1987

[6] Lenz, R., Reichert, M.: IT support for healthcare processes – premises, challenges, pespectives, Data & Knowledge Engineering, Vol. 61(1), Elsevier Science Publishers B. V., 2007, S. 39-58

[7] Williams, J. G., Sochats, K. M., Morse, E.: Visualization, Annual Review of Information Science and Technology (ARIST), Vol. 30, 1995, S. 161-207

[8] Robertson, P. K.: A Methodology for Choosing Data Representations, IEEE Computer Graphics & Applications, Vol. 11(3), 1991, S. 56-67

[9] Schumann, H., Müller, W.: Visualisierung: Grundlagen und allgemeine Methoden, Springer-Verlag Berlin Heidelberg, 2000

[10] Reichert, M., Kuhn, K., Dadam, P.: Prozeßreengineering und -automatisierung in klinischen Anwendungsumgebungen, 41. Jahrestagung der Dt. Gesellschaft für Medizinische Informatik, Biometrie und Epidemiologie (GMDS), Bonn, Deutschland, September 1996, S. 219-223

[11] Marconi, A., Pistore, M., Sirbu, A., Leymann, F., Eberle, H., Unger, T.: Enabling Adaptation of Pervasive Flows: Build-in Contextual Adaptation, 7th International Joint Conference, ICSOC-ServiceWave 2009, Stockholm, Schweden, November

24-27, 2009, Service-Oriented Computing, LNCS, Vol. 5900/2009, Springer-Verlag Berlin Heidelberg, 2009, S. 445-454

[12] Bucchiarone, A., Lafuente, A. L., Marconi, A., Pistore, M.: A formalisation of Adaptive Pervasive Flows, Proceedings of the 6th International Workshop on Web Services and Formal Methods (WS-FM'09), Bologna, Italien, September 04-05, 2009

[13] Herrmann, K., Rothermel, K., Kortuem, G., Dulay, N.: Adaptable Pervasive Flows – An Emerging Technology for Pervasive Adaptation, Proceedings of the 2008 Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops (SASOW 2008), Italien, Oktober 20-24, 2008, IEEE Computer Society, 2008, S. 108-113

[14] Bernstein, A.: How can cooperative work tools support dynamic group processes? Bridging the specifity frontier, Proceedings of the 2000 ACM conference on Computer supported cooperative work (CSCW), Philadelphia, Pennsylvania, United States, Dezember 02-06, ACM, 2000, S. 279-288

[15] Malone, T. W., Crowston, K., Lee, J., Pentland, B., Dellarocas, C., Wyner, G., Quimby, J., Osborn, C., Bernstein, A., Herman, G., Klein, M. and O´Donnell, E.: Tools for inventing organizations: Toward a handbook of organizational processes, Management Science, Vol. 45, 1999, S. 425-443

[16] Kreifelts, T., Hinrichs, E., Woetzel, G.: Sharing To-Do Lists with a Distributed Task Manager, ECSCW Proceedings of the third conference on European Conference on Computer-Supported Cooperative Work, Mailand, Italien, September 13-17, 1993, Kluwer Academic Publishers, 1993, S. 31-46

[17] ALLOW, URL: http://www.allow-project.eu/

[18] Wagner, S.: A Concept of Human-oriented Workflows, Diplomarbeit Nr.: 2987, Universität Stuttgart, Institut für Architektur von Anwendungssystemen, Deutschland, Januar 2010

[19] Staudenecker, A.: Verfeinerung menschlicher Aktivitäten in Workflows, Diplomarbeit Nr.: 2961, Universität Stuttgart, Institut für Architektur von Anwendungssystemen, Deutschland, 2009

[20] Bardram, J. E.: Plans as Situated Action: An Activity Theory Approach to Workflow Systems, ECSCW Proceedings of the Fifth European Conference on Computer-Supported Cooperative Work, Lancaster, UK, September 07-11, 1997, Kluwer Academic Publishers, 1997, S. 17-32

[21] Vicari, J.: Die Planwirtschaftlerin, brand eins, Jahrgang 12, Heft 01, 2010, S. 98-101

[22] Bobrik, R.: Konfigurierbare Visualisierung komplexer Prozessmodelle, Dr. Hut, München, 2008

[23] Burkhard, R. A., Meier, M.: Tube Map: Evaluation of a Visual Metaphor for Interfunctional Communication of Complex Projects, Proceedings of the 4th International Conference on Knowledge Management (I-KNOW´04), Graz, Österreich, Juni 30-Juli 02, 2004

[24] Burkhard, R. A., Meier, M.: Tube Map Visualization: Evaluation of a Novel Knowledge Visualization Application for the Transfer of Knowledge in Long-Term Projects, Journal of Universal Computer Science, Vol. 11(4), 2005, S. 473-494

[25] Burkhard, R. A., Meier, M., Rodgers, P., Smis, M., Stott, J.: Knowledge Visualization: A Comparative Study between Project Tube Maps and Gantt Charts, Proceedings of the 5th International Conference on Knowledge Management (I-KNOW´05), Juni 29-Juli 01, 2005, S. 388-395

[26] Burkhard, R. A.: Impulse: Using Knowledge Visualization in Business Process Oriented Knowledge Infrastructures, Journal of Universal Computer Science, Vol. 0(2), 2005, S. 170-188

[27] Purchase, H. C., Hoggan, E. E., Görg, C.: How Important Is the "Mental map"? – An Empirical Investigation of a Dynamic Graph Layout Algorithm, Proceedings of the 14th international conference on Graph drawing, Karlsruhe, Deutschland, September 18-20, 2006, Graph Drawing, LNCS, Vol. 4372/2007, Springer-Verlag Berlin Heidelberg, 2007, S. 184-195

[28] Telea, A.: Data visualization : Principles and practice, AK Peters, 2008

[29] Zhao, X.: Automatisches Layout von Prozessgraphen, Masterarbeit, Universität Ulm, Institut für Datenbanken und Informationssysteme, Deutschland, Juni 2007

[30] Sugiyama, K., Tagawa, S., Toda, M.: Methods for Visual Understanding of Hierarchical System Structures, IEEE Transactions on Systems, Man and Cybernetics, Vol. 11(2), 1981, S. 109-125

[31] Hollingsworth, D.: The Workflow Reference Model, Technischer Bericht WFMC-TC-1003, Workflow Management Coalition, Januar 1995

[32] OASIS: Web Services – Human Task (WS-HumanTask) Specification Version 1.1, Committee Draft 06, 2009

[33] WebSphere Integration Developer, URL: http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/index.jsp?topic=/com.ibm.wbit.help.prodovr.doc/topics/chumantsk.html

[34] Becker, A., Pant, M.: Android 2: Grundlagen und Programmierung, 2. Auflage, dpunkt, Heidelberg, 2010

[35] Google: Android Dev Guide, URL: http://developer.android.com/resources/samples/SampleSyncAdapter/index.html

[36] Eckhardt, O.: Menschen-zentrierte Restrukturierung von Human Tasks, Diplomarbeit Nr.: 2999, Universität Stuttgart, Institut für Architektur von Anwendungssystemen, Deutschland, 2010

[37] IBM: Working with XML on Android, Technischer Bericht, URL: http://www.ibm.com/developerworks/opensource/library/x-android/index.html

[38] Bardram, J. E.: Activity-based computing: support for mobility and collaboration in ubiquitous computing, Personal and Ubiquitous Computing, Vol. 9(5), Springer-Verlag London, 2005, S. 312-322

[39] Bardram, J. E.: Activity-based computing for medical work in hospitals, ACM Transactions on Computer-Human Interaction (TOCHI), Vol. 16(2), ACM New York, 2009, Artikel 10

[40] Google: Android Dev Guide, URL: http://developer.android.com/reference/android/widget/ListView.html

[41] Google: Android Dev Guide, URL: http://developer.android.com/reference/android/widget/Adapter.html

[42] Google: Android Dev Guide, URL: http://developer.android.com/guide/topics/data/data-storage.html

Alle URLs wurden zuletzt am 15.08.2010 geprüft.

Erklärung

Hiermit versichere ich, diese Arbeit selbständig verfasst und nur die angegebenen Quellen benutzt zu haben.

Unterschrift:

Stuttgart, den 17.08.2010