UNIVERSITÄT PADERBORN

UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

Fachbereich für Elektrotechnik, Informatik und Mathematik
Institut für Informatik
Arbeitsgruppe Mensch-Computer-Interaktion

Entwicklung eines Konzeptes zur Erstellung von Aufgabenmodellen an einem interaktiven Display

Masterarbeit

Wirtschaftsinformatik

Eingereicht bei:
Prof. Dr. Gerd Szwillus

Betreut durch:
Adrian Hülsmann, M.Sc.

Eingereicht von:
Jens Eggers, cand. M.Sc.
Warburger Str. 68b
33098 Paderborn
Matr.Nr: 6598361

Paderborn, den 31.01.2013

Eidesstattliche Erklärung

Hiermit versichere ich, Jens Eggers, die vorliegende Arbeit selbstständig und unter ausschließlicher Verwendung der angegebenen Literatur und Hilfsmittel erstellt zu haben. Alle Stellen, die wörtlich oder sinngemäß veröffentlichtem oder unveröffentlichtem Schrifttum entnommen sind, habe ich als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Ort, Datum Unterschrift

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AUI	Abstract User Interface
CUI	Character User Interface
CLI	Command-Line Interface
CTT	Concurrent Task Trees
CTTE	Concurrent Task Tree Environment
GUI	Graphical User Interface
HCI	Human Computer Interaction
K-MADe	Kernel of Model for Activity Description environment
MCI	Mensch-Computer-Interaktion
MT	Multi-Touch
MTT	Multi-Touch Tabletop
NUI	Natural User Interface
OSD	On Screen Display
SE	Software Engineering
SMS	Short Message Service
TUI	Tangible User Interfaces
UI	User Interface
UML	Unified Modeling Language
VTMB	Visual Task Model Builder
WPF	Windows Presentation Framework

Kurzreferat

Diese Arbeit beschäftigt sich mit der Entwicklung eines Konzeptes zur Erstellung von Aufgabenmodellen auf einem interaktiven Display. Dabei werden bekannte und neue Formen der Interaktion betrachtet und existierende Entwicklungsumgebungen für Aufgabenmodelle evaluiert, um eine neue Art des Erstellens von Aufgabenmodellen zu etablieren. Im Verlauf der Arbeit werden Aspekte der Darstellung, Interaktion sowie Kollaboration bei der Gestaltung der Modelle mit einem Multi-Touch Tabletop diskutiert und Elemente aus all diesen Gebieten zusammengeführt, um die User Experience intuitiv zu halten und Usability sicherzustellen. Dazu finden Techniken aus den Gebieten Natural UIs und Tangbile UIs Anwendung, die sowohl die Eingabe mit Fingern und Stempeln als auch die Kopplung des MTT mit einem Smartphone zum Bearbeiten und Erstellen von Diagrammelementen beinhalten. Das Konzept rundet damit ab, dass für das gemeinsame Arbeiten Vorschläge zur Arbeitsteilung gemacht werden und für die Benutzer die Einrichtung individueller Territorien vorbehalten wird, um komplexe Sachverhalte getrennt zu behandeln und diese im Anschluss wieder zusammenzufügen oder in Zusammenhang zu stellen.

Abstract

This thesis deals with developing a concept for creating task models using an interactive display. For reaching this aim, common and new forms of interaction are being considered while existing environments for task modelling are being evaluated, to establish a totally new kind of designing task models. In the course of the thesis, aspects of presentation, interaction and collaboration being used on a multi-touch tabletop to create the models are discussed whilst elements of all given fields are brought together to keep an intuitive user experience and ensure usability. For this purpose, techniques out of the fields Natural UIs and Tangible UIs are used which provide user input by fingers or stamps and, in addition, incorporate the usage of a smartphone paired with the MTT for editing and creating functions on diagram elements. The concept concludes with proposals for dividing the work between participants and therefore enabling the arrangement of individual territories, thus handling complex issues separately and bringing them back together or putting them into context in a later step.

1 Einführung

1.1. Problembeschreibung und Motivation

Im Entwicklungsprozess von Benutzungsschnittstellen bedarf es mehrerer wichtiger Schritte, um zu einem aufgaben- und benutzergerechten Ergebnis zu kommen. Neben der Dialog- und der Benutzermodellierung ist ein wichtiger Schritt im modellbasierten Entwicklungsprozess die Aufgabenmodellierung. Diese identifiziert die Aufgaben, die durch den Benutzer mit der Schnittstelle am System durchgeführt werden können. Dieser Schritt sollte erfolgen, bevor das eigentliche Interface als Prototyp designt und implementiert wird, damit beim Entwickeln der Dialoglogik und des späteren Aussehens der Schnittstelle diese Aufgaben bereits bekannt sind. SZWILLUS beschreibt den Entwicklungsprozess wie folgt (Szwillus 2012):

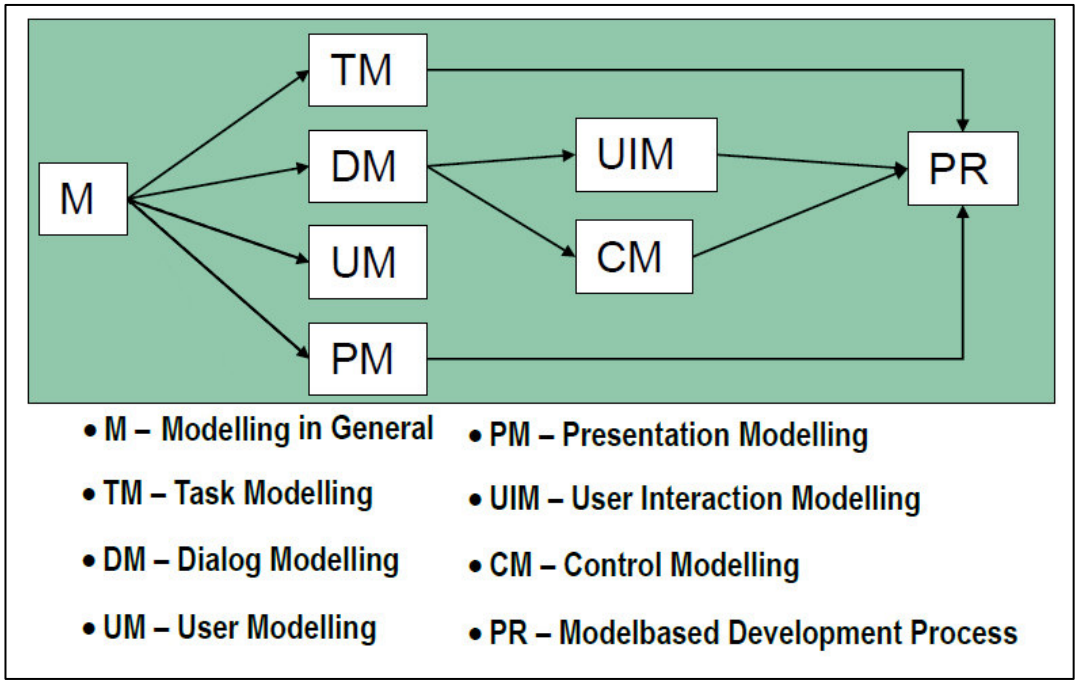


Abbildung 1.1 - Model based Development Process for User Interfaces[1]

In der Praxis wird jedoch häufig bevorzugt mit dem Designen der Schnittstelle begonnen, ohne die genauen Aufgaben des Benutzers zu kennen (Stichwort „Rapid Prototyping“[2]). Dieser Umstand sollte dahingehend verbessert werden, dass die Modellierung der Aufgaben des Benutzers attraktiver gestaltet wird, als dies bisher der Fall war und die Verantwortlichen dazu motiviert werden, vor der eigentlichen Implementierung über die vom Benutzer zu lösenden Aufgaben nachzudenken.

In den letzten Jahren hat die Nutzung von Geräten mit intuitiven Bedienungsmöglichkeiten stark zugenommen. Das Unternehmen Apple hat mit seinen Produkten iPhone und iPad den Markt für mobile Endgeräte revolutioniert, was nicht zuletzt auf die konsequente Bedienung der Geräte ohne Tasten nur mit Hilfe eines berührungsempfindlichen Bildschirms zurückzuführen ist. Diese Art der Bedienung wurde seitdem von nahezu allen Herstellern übernommen, und sie bietet daher großes Potenzial für neue Anwendungen, die sich nicht nur ausschließlich im mobilen Endkundenbereich ansiedeln müssen.

[1] Quelle: (Szwillus 2012), S. 2 [Intro-2]; modifiziert
[2] Rapid Prototyping beschreibt in der Informatik das schnelle und direkte Erzeugen von (Software-)Prototypen, um z. B. im UI Designprozess ein Gefühl für die spätere Maske und den Dialog der Software zu bekommen.

Der Konkurrent Microsoft hat dazu bereits vor einem Jahrzehnt mit der Entwicklung an einer stationären Anwendung dieser Technologie begonnen. Der im Jahre 2007 unter dem Namen „Surface“ eingeführte und heute unter dem Namen „PixelSense“ bekannte Computer erschien mit Anwendungen, die man nur mit der Hilfe seiner Hände und Finger bedienen konnte. Dabei wird wie bei mobilen Endgeräten auf eine gestenbasierte Bedienung gesetzt, die in der Literatur unter dem Oberbegriff „Natural User Interfaces“ zusammengefasst wird, da die Bedienung sehr natürlich und intuitiv mit den Fingern erfolgt (Bollhoefer et al. 2009).

Microsoft lieferte mit dem Betriebssystem Windows Vista erstmals native Unterstützung für berührungsempfindliche Steuerung aus und stellte sogleich das Windows Presentation Framework vor, das für die Entwicklung von Anwendungen für eben diese Technologie vorgesehen ist. Vor allem besticht hier der Einsatz einer größeren Fläche, wodurch beide Hände mit je mehr als 2 Fingern effektiv eingesetzt werden können oder mehrere Akteure gleichzeitig an ein und demselben Projekt arbeiten.

Die angesprochenen Technologien bieten neue Anwendungsmöglichkeiten für das Erstellen von Aufgabenmodellen, die die Grenzen der konventionellen Bedienung mit der Maus aufzubrechen versuchen – sie werden sie dennoch mit hoher Wahrscheinlichkeit niemals komplett ersetzen können. Steht man vor dem Problem, komplexe Aufgabenmodelle mit den bekannten Editoren aufzubauen, wird man irgendwann mit Beschränkungen hinsichtlich Bedienbarkeit und Usability der existierenden Entwicklungsumgebungen konfrontiert. Aus diesem Grund ist es wichtig, ein Konzept zu entwickeln, das Anwendungsentwickler zukünftig dabei unterstützt, Software für die Aufgabenmodellierung in einer gestenbasierten Umgebung zu produzieren.

Im Idealfall erstellt die Arbeit eine Art Anleitung, wie eine Software für den Einsatz auf Multi-Touch-fähigen Geräten implementiert werden sollte, um damit produktiv an der Entwicklung von Aufgabenmodellen arbeiten zu können. Einen besonderen Reiz macht dabei die Zusammenarbeit mehrerer Kollaborateure an ein und demselben Gerät aus, die es hinsichtlich der Benutzbarkeit zu untersuchen gilt.

1.2. Ziel und Aufbau der Arbeit

Das Ziel dieser Arbeit ist das Entwickeln eines Konzeptes für die Modellierung der Aufgaben eines Benutzers in Form von Modellen auf einem Multi-Touch-fähigen Tisch (interaktives Display). Um dies zu erreichen, werden bestehende Bedienkonzepte aus der einschlägigen Literatur hinzugezogen und evaluiert. Weiterhin werden bereits existierende Werkzeuge zur Erstellung von Aufgabenmodellen analysiert und dahingehend überprüft, ob eine Adaption für den Multi-Touch-Tisch sinnvoll wäre.

Außerdem soll darüber nachgedacht werden, ob bei Aufgabenmodellen, die aufgrund ihres hierarchischen Aufbaus oft durch Bäume repräsentiert werden, auch eine andere Darstellungsweise Sinn ergeben könnte (vielleicht ergibt bspw. der Einsatz des von Grafikprogrammen bekannten Konzepts der Ebenen Sinn, um die hierarchische Struktur eines Aufgabenmodells effizienter darzustellen). Ansätze für die Darstellung, Entwicklung und Notation von Aufgabenmodellen sind vorhanden, und werden im Hinblick auf eine Adaption für den Touch-Tisch analysiert.

Weiterhin soll der Frage nachgegangen werden, wie man komplexe Aufgabenmodelle sinnvoll auf einem Touch-Tisch darstellen und entwickeln kann, und ob die Erstellung auch zusammen mit weiteren Personen kooperativ sinnvoll durchführbar wäre (z. B. durch Einteilung des Aktionsraums der jeweils Agierenden). Die aufgeteilte Erstellung von Aufgabenmodellen mit unterschiedlichen Akteuren (Benutzer, System) könnte ein sinnvoller Arbeitsbereich sein.

Eine intuitive Bedienung wäre im Sinne der Natural User Interfaces wünschenswert, daher sollte auch die Auswahl der Symbolik ansprechend und leicht zu verstehen sein. Welche Symbolik durch das Windows Presentation Framework vorgegeben wird, und ob diese Darstellungsangebote ausreichen oder individuell erweitert werden sollten, soll ebenfalls innerhalb der Arbeit geklärt werden.

Ein weiteres beeinflussendes Gebiet stellen die Tangible User Interfaces dar, welche eventuell auch eine größere Rolle einnehmen können. Sie integrieren innerhalb der sogenannten erweiterten Realität (engl. „Augmented Reality“) physische Objekte in den Entstehungsprozess von Modellen und machen das Interface auf diese Weise haptisch „anfassbar“. Inwieweit es sich als unterstützend darstellt, solche Objekte auch für die Erstellung von Aufgabenmodellen zu verwenden, soll ebenfalls Teil dieser Arbeit sein.

Werden alle genannten Aspekte in Kombination gebracht, kann ein fundiertes Konzept für die Erstellung von Aufgabenmodellen auf dem Multi-Touch-Tisch entstehen. Bestehende Werkzeuge sollen dabei keinesfalls ersetzt, sondern um die Anwendung des interaktiven Displays erweitert werden, um den technischen Weiterentwicklungen der letzten Jahre auch in die Aufgabenmodellierung gerecht zu werden.

Die Arbeit gliedert sich in fünf Kapitel. Nach dieser Einführung in Kapitel 1, werden zunächst die fachlichen Grundlagen für die Thematik in Kapitel 2 erarbeitet, bevor in Kapitel 3 bereits vorhandene Entwicklungsumgebungen zur Aufgabenmodellierung sowie Interaktionstechniken analysiert werden. Das eigentliche Konzept für ein interaktives Display wird dann schließlich innerhalb von Kapitel 4 entwickelt. Eine Übersicht zeigt die Abbildung 1.2:

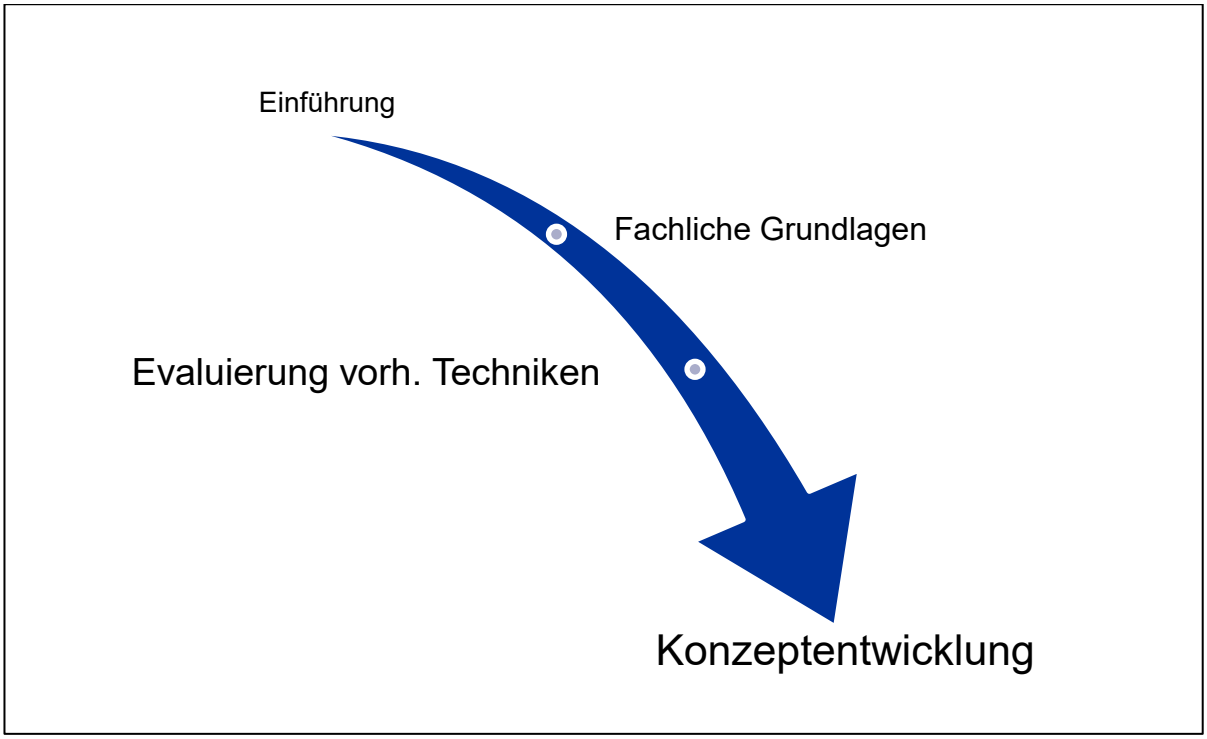


Abbildung 1.2 - Aufbau der Arbeit

Eine Zusammenfassung sowie Ausblicke in die Zukunft runden die Arbeit im Kapitel 5 ab, in dem noch einmal die Ziele der Arbeit reflektiert und mit den gewonnen Erkenntnissen zusammengebracht werden. Angesprochen wird dabei ebenfalls, wie zukünftig die Aufgabenmodellierung gestaltet werden kann.

Die Arbeit basiert größtenteils auf Forschungsergebnissen von Prof. Fabio Paternò und Prof. Dr. Gerd Szwillus im Bereich der Aufgabenmodellierung, sowie für den Bereich interaktive Displays auf dem Buch „Emotionales Interaktionsdesign“ von Rainer Dorau und diversen Forschungsergebnissen auf dem Gebiet, vorwiegend des Lehrstuhls UI&SE der Universität Magdeburg um Prof. Dr. Raimund Dachselt.

2 Grundlagen

In diesem Kapitel sollen vorab die fachlichen Grundlagen erklärt werden, um die Basis für die Entwicklung eines Konzeptes für interaktive Displays zur Erstellung von Aufgabenmodellen zu schaffen. Weiterhin wird die Untersuchung bekannter Editoren vorbereitet, da hier erklärte Begrifflichkeiten dort später aufgegriffen werden. Dazu werden zunächst die benötigten Begriffe definiert und im Anschluss die Bereiche *Aufgabenmodellierung* (s. 2.2) sowie *klassische* (s. 2.3.1) und *moderne User Interfaces* (s. 2.3.2) detailliert vorgestellt.

2.1 Definitionen

Zum Einstieg in die Thematik müssen zu Beginn ein paar grundsätzliche Definitionen erfolgen. Die hier behandelten Begriffe tauchen im weiteren Verlauf der Arbeit immer wieder auf und es ist daher sinnvoll, sie am Anfang festzulegen.

Modellierung

Im Entwicklungsprozess der Modellierung entstehen Modelle als künstliche Repräsentationen bestimmter Aspekte der Realität. SZWILLUS beschreibt Modelle als *Artefakte* (z. B. eine Zeichnung, ein Text oder ein physisches Objekt), die dazu dienen, relevante Aspekte eines realen Systems zu reflektieren oder zu repräsentieren. Modelle werden benötigt, um bestimmte Eigenschaften realer Objekte vereinfacht und abstrakt darzustellen. Das Artefakt an sich ist dabei künstlicher Natur, und kann nicht alle Aspekte der Realität verkörpern, sondern nur die (für z. B. eine zu lösende Aufgabe) wichtigen und dafür benötigten. SZWILLUS veranschaulicht dies in Abbildung 2.1 (Szwillus 2012):

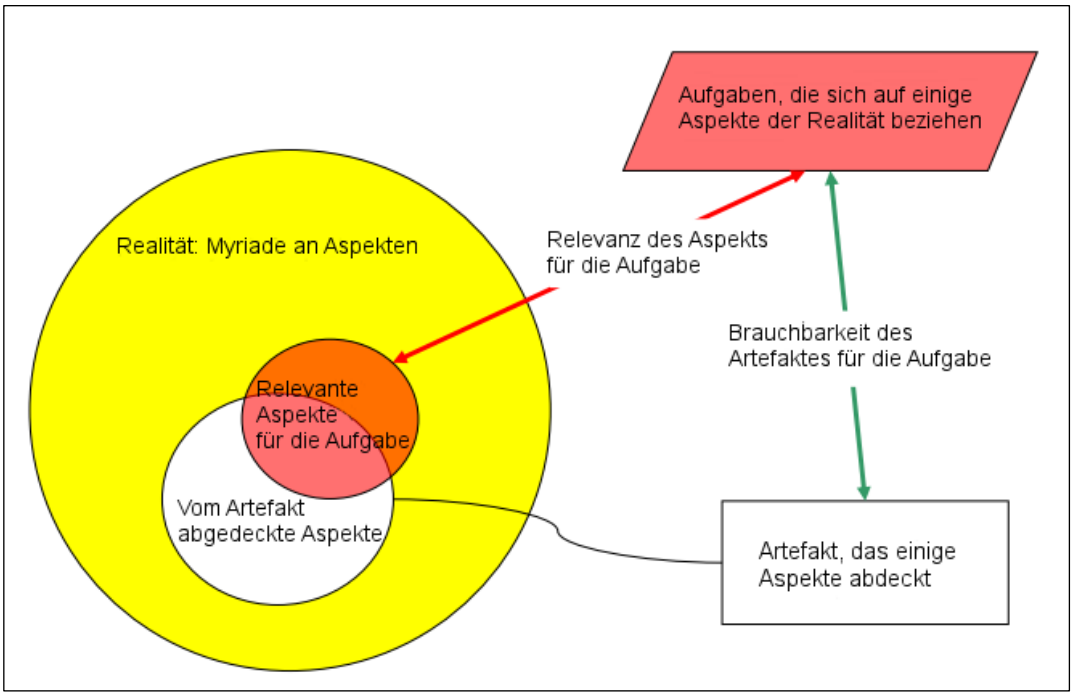


Abbildung 2.1 - Artefakt und Aspekt im Kontext[3]

Die Realität bietet eine Unmenge (Myriade) an Aspekten. Diese komplett in einem Modell abzubilden erscheint unmöglich; und entspricht auch nicht der Definition von Modellen. Innerhalb der Realität gibt es Artefakte, die einzelne Aspekte umfassen. Gleichzeitig sind aber für die Erfüllung einer Aufgabe

[3] Quelle: (Szwillus 2012), S. 8 [M-6]; übersetzt ins Deutsche

weitere Aspekte der Realität nötig. Um das richtige Artefakt zu finden, müssen sich die beiden inneren Kreise im Idealfall deckungsgleich einander annähern, um die „perfekte“ Einheit zu bilden. Dies ist in der Praxis leider oft kaum realisierbar, sodass Kompromisse eingegangen werden und das am besten für die Aufgabe geeignete Artefakt ausgewählt wird.

Aufgabenmodelle

Aufgabenmodelle (engl.: „Task Models“) beschreiben Aufgaben (z. B. eines Benutzers) in Modellform. Das Ziel von Aufgabenmodellen ist es, die eigentliche durchzuführende Aufgabe in verschiedene, meist einfach zu lösende Unteraufgaben zu zerlegen und diese dann in einer bestimmten Abfolge auszuführen. Dabei wird ähnlich dem aus der Algorithmentheorie bekannten „Divide and Conquer“[4] Prinzip verfahren, bei dem ein Problem in kleinere Teilprobleme zerlegt wird, die effizienter lösbar sind. Ein Beispiel zeigt Abbildung 2.2:

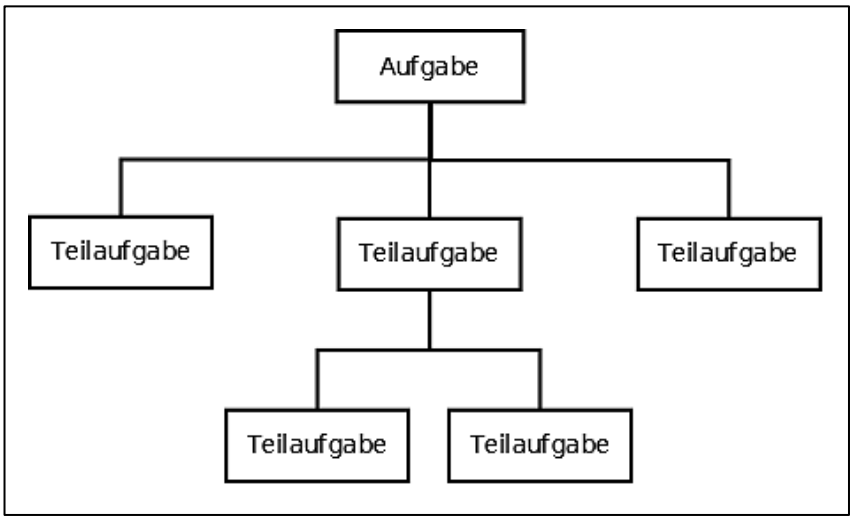


Abbildung 2.2 - Struktur eines Aufgabenmodells

Hierbei wird die hierarchische Struktur deutlich, die Aufgabenmodelle ausmacht. Die Hauptaufgabe wird durch Erbringung der Teilaufgaben gelöst. Es fällt auf, dass die entstehenden, untergeordneten Teilbäume (je nach Aufgabe) durch Rekursion beliebig komplex werden können, aber alle disjunkte, in sich geschlossene Unteraufgaben darstellen. Zum besseren Verständnis soll an dieser Stelle auf das später folgende Kapitel 2.2 verwiesen werden, in dem wesentlich detaillierter auf die Thematik eingegangen wird.

Die Abfolge oder das Weglassen bestimmter Teilaufgaben, da diese auch optional durchzuführen sein können, werden durch sogenannte *temporale Relationen* abgebildet.

Temporale Relation

Temporale Relationen beschreiben die Beziehungen zwischen den einzelnen Unteraufgaben innerhalb eines Aufgabenmodells. Sie beschreiben somit die Art, wie Menschen eine Aufgabe ausführen. Dabei geht es beispielsweise um die Frage, ob Unteraufgaben in einer bestimmten Reihenfolge ausgeführt werden müssen, ob eine Unteraufgabe eine andere erst ermöglicht, ob sie mehrfach ausgeführt werden muss oder ob Unteraufgaben optionaler Natur sind und nicht immer zwingend durchgeführt werden müssen. Diese verschiedenen Arten temporaler Relationen sind essentiell wichtig und werden benötigt, um den gesamten Ablaufprozess einer Aufgabe korrekt darzustellen.

[4] Divide and Conquer: das „Teile und Erobere“ Prinzip

PATERNÒ sowie SZWILLUS haben unterschiedliche Relationen identifiziert, die sich in ihrer Definition teilweise überschneiden. Sie sind in der folgenden Tabelle zusammengefasst (Paternò 2001) (Szwillus 2011):

PATERNÒ	SZWILLUS
Hierachie (Hierarchy)	**Sequenz (Sequence)**
Ermöglichen (Enabling)	**Zufällige Sequenz (Random Sequence)**
Auswahl (Choice)	**Parallel**
Ermöglichen plus Informationsübermittlung	**Unbeschränkt (Unrestricted)**
Nebenläufige Aufgaben (Concurrent Tasks)	**Selektion (Selection)**
Nebenläufige kommunizierende Aufgaben	**Optional**
Aufgabenunabhängigkeit (Task Independance)	**Iteration**
Deaktivieren (Disabling)	
Unterbrechen–Wiederaufnehmen (Suspend-Resume)	

Tabelle 2.1 - Arten von temporalen Relationen[5]

Eine detailliertere Auflistung beider Varianten befindet sich im Anhang der Arbeit.

User Interface

Ein User Interface (engl. für *Benutzungsschnittstelle*) verkörpert die Schnittstelle zwischen dem System und einem Benutzer. Der Benutzer erhält Informationen vom System über dessen Zustand, und kann mit Hilfe eines Eingabegerätes wie Tastatur, Maus oder Touchscreen Manipulationen am System durchführen, wodurch er weitere Rückmeldungen erhält. Sowohl Informationserfassung als auch Manipulation durch den Benutzer können an der Schnittstelle durch verschiedene menschliche Sinne erfolgen.

Am häufigsten verbreitet sind dabei audio-visuelle Rückkopplungen mit dem System über die Schnittstelle, unter Benutzung von Monitoren und Lautsprechern; aber auch haptisches Feedback (z. B. für Blinde) ist durchaus möglich, etwa durch die Benutzung einer Braillezeile, die Blindenschrift ausgibt; oder die verstärkt aus dem Gaming-Bereich bekannten *Force Feedback*[6] Eingabegeräte, die Vibrationen bei bestimmten Ereignissen erzeugen. Diese Technik ist auch bei heutigen Smartphones bekannt, bei denen der Vibrationsmotor zusätzlich zur Benachrichtigungsfunktion auch für die Rückkopplung einer Berührung oder Auswahl am Display genutzt wird. Olfaktorische oder gustatorische Formen der Interaktion sind derzeit nicht üblich bzw. bekannt.

Interaktives Display

Ein interaktives Display ermöglicht Interaktionen mit bzw. Manipulation des Systems durch den Benutzer direkt am sichtbaren Aktionsraum. Das eigentliche Eingabegerät überlagert den Bildschirm und kann durch Berührung Aktionen auslösen; Ein- und Ausgabegerät sind somit vereint. Klassisch ist hierbei der Touchscreen zu nennen, es gibt aber auch Varianten die nur auf bestimmte gekoppelte Instrumente wie bspw. einen Stift reagieren. Interaktive Displays werden noch einmal detaillierter in Kapitel 2.4 aufgegriffen.

Nachdem nun tragende Begrifflichkeiten definiert sind, werden im nächsten Schritt die für die Arbeit relevanten Themenbereiche näher betrachtet.

---

[5] Quellen: (Paternò 2001), S 8-9; (Szwillus 2011), S. 279

[6] Force Feedback: engl. für Kraftrückkopplung, ermöglicht ein haptisches Erfassen von bestimmten Events durch den Benutzer, indem beispielsweise Vibrationsmotoren durch einen speziellen Treiber angesteuert werden.

2.2 Aufgabenmodellierung

Die Aufgabenmodellierung ist ein wichtiger Vorgang im Designzyklus von (Software-)Systemen. In einem ersten Schritt müssen die Anforderungen des neu zu erstellenden Systems (*Requirements of The New System*) definiert und diese in Einklang mit dem Aufgabenmodell des existierenden Systems (*Existing System's Task Model*) gebracht werden, um die Abdeckung aller für das zukünftige System relevanten Aspekte im Prozess sicherzustellen. PATERNÒ hat den Vorgang wie folgt illustriert (Paternò 2001):

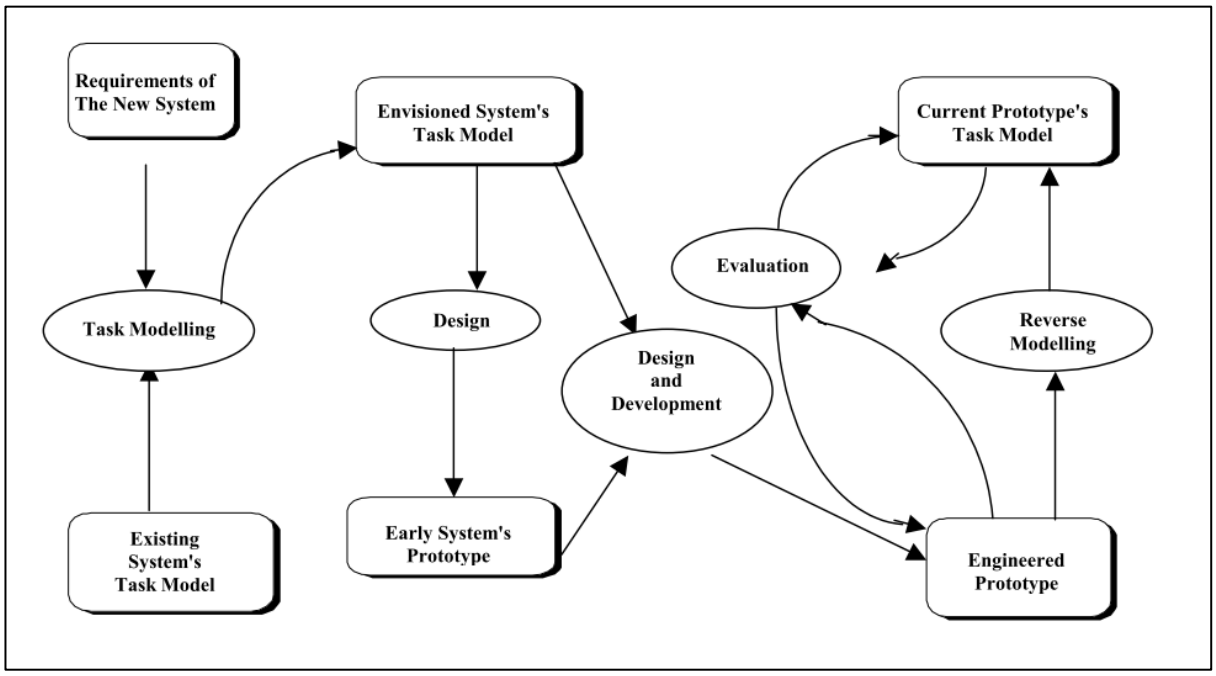


Abbildung 2.3 - Use of task models in the design cycle[7]

Die genannten Schritte vereinen sich in der Aufgabenmodellierung (*Task Modelling*), und als deren Ergebnis entsteht dann das sogenannte *Envisioned System's Task Model*, also ein Aufgabenmodell des zukünftigen Systems. Dieses Modell dient als Basis für alle weiteren Design und/oder Entwicklungsprozesse, aus denen dann verschiedene Stufen von Prototypen und am Ende das fertige Produkt hervorgehen. Bevor dieses marktreif werden kann, wird der *Engineered Prototype*, ein erster benutzbarer Prototyp, durch Evaluation bis zum fertigen Produkt weiter verbessert. Die Evaluation wird beeinflusst durch das Aufgabenmodell des momentan betrachteten Prototyps (*Current Prototype's Task Model*), welches mit Hilfe von *Reverse Modelling*[8] aus dem bestehenden Prototyp generiert wurde.

Der Prozess der Aufgabenmodellierung besteht grundsätzlich aus den zwei Phasen *Aufgabenanalyse* und der eigentlichen Entwicklung des *Aufgabenmodells.*

2.2.1 Aufgabenanalyse

Bevor ein formales Modell zu einer Aufgabe entwickelt werden kann, muss in einem ersten Schritt eine Analyse der Aufgabe erfolgen. Im Kern geht es dabei um die Erfassung der Aufgaben mit einem bereits existierenden System, und dem zukünftigen, neu zu entwickelnden System. Dazu ist es nötig, die Benutzer in diesen Prozess mit einzubeziehen, da sich diese am besten in ihrem Aufgabenumfeld auskennen.

---

[7] Quelle: (Paternò 2001), S. 4

[8] Reverse Modelling: Analog zum *Reverse Engineering* wird ein Modell aus dem betrachteten Prototyp erzeugt, ohne Kenntnis über dessen Funktionsweise oder inneren Aufbau

Nach SZWILLUS gilt es, innerhalb der Aufgabenanalyse folgende Fragen zu beantworten (Szwillus 2012):

- Wer sind die Benutzer der Anwendung? (Rollen, Erfahrungen, Nutzertypen...)
- Was sind die Ziele der Nutzer beim Benutzen des Systems?
- Woher stammen die gewonnenen Informationen?
- Welche Methoden kennen die Benutzer und welche wenden sie an?
- Was sind die Ziele des Entwicklers?

Um diese Fragen beantworten zu können, schlägt er eine Feldanalyse vor, die die Aspekte Organisation, Menschen, Meinungen, Zeitrahmen sowie (Aktions-)Räume untersucht. Ziel dieser Feldanalyse ist es, genug Informationen zusammenzutragen, um eine erste Version einer Benutzungsschnittstelle kreieren zu können. Als mögliche Quellen für diese Informationsgewinnung nennt er drei Ansätze:

- **„McKinsey Ansatz“:** Interviews, Fragebögen, zwanglose Unterhaltungen
- **Ethnografischer Ansatz:** Beobachten, „über die Schulter gucken“, beiläufige Befragungen
- **„Mingling“ (vermischender) Ansatz:** Arbeiten mit den Betroffenen, Selbstversuch

Das Problem bei diesen Ansätzen ist die Unterschiedlichkeit der Ergebnisse, da die Informationen aus verschiedenen Quellen gewonnen werden und somit schwer vergleichbar sind. Einen strukturierteren Ansatz liefern die beiden folgenden Methoden.

2.2.1.1 Aufgabenanalyse nach Rubinstein/Hersh

Bereits im Jahre 1984 widmeten sich RICHARD RUBINSTEIN und HARRY M. HERSH in einem Kapitel ihres Werks „*The human factor: Designing computer systems for people*“ der Aufgabenanalyse sowie Nutzungsmodellen. Ihr Vorgehen ist eine Frage/Antwort Technik, bei der zehn vordefinierte Fragen in Form eines Interviews oder Fragebogens beantwortet werden. Die Autoren behaupten, dass die Beantwortung dieser zehn Fragen alle relevanten Aspekte abdeckt. Eine verkürzte Übersicht zeigt Tabelle 2.2 (Szwillus 2012):

Wer sind die **Benutzer**? (Zielgruppe)	Was sind ihre **Aufgaben**?	Was **wissen** sie?	Was ist das **Umfeld** der Benutzung?	Wie ist die **Beziehung** zwischen Nutzer und Daten?
Welche anderen **Werkzeuge** werden benutzt?	**Kommunizieren** Benutzer untereinander?	Wie **oft** werden die Aufgaben durchgeführt?	Gibt es **zeitliche Begrenzungen**?	Was wenn etwas **schief** geht?

Tabelle 2.2 - Rubinstein/Hersh Fragen zur Aufgabenanalyse[9]

Eine Anpassung der zehn Fragen an den konkreten Anwendungsfall ist unvermeidbar, um der Domäne gerecht zu werden. Die Anpassung an die jeweilige Situation bzw. Organisation ist aufgrund des universellen Charakters der Fragestellungen jedoch leicht durchführbar.

Rubinstein/Hersh sehen dann eine Befragung in zwei Phasen vor.

1. Task Analysis Questions – bezogen auf die aktuellen Situation mit dem vorhandenen System
2. Use Model Questions – bezogen auf die zukünftige Situation mit dem neuen System

[9] Quelle: (Szwillus 2012), S. 27 [Task Modelling-3]

In der ersten Phase werden Informationen über die momentane Situation gesammelt (Momentaufnahme). Während der zweiten Phase werden (analog zur ersten Phase) wieder die gleichen Fragen gestellt, aber unter der Annahme, dass das neue System bereits implementiert wurde. Hauptaugenmerk liegt dabei auf der Frage, was die Veränderungen für die Benutzer bedeuten. Es ist zu klären, ob andere Aufgaben ausgeführt werden, oder die gleichen wie zuvor auf eine andere Art und Weise. Weiterhin ist interessant, in welcher Hinsicht sich die Umgebung verändert hat.

Nicht ermittelt werden soll das Aussehen des neuen Systems, da die Aufgabenanalyse keinen Designschritt darstellen sollte, sondern von vorgelagerter Natur ist. Werden alle Vorgaben eingehalten, liefert die Rubinstein/Hersh Methode konkrete Informationen darüber, was Benutzer in Bezug auf ihre Aufgaben getan haben und tun werden. Sie sensibilisiert darüber hinaus für eventuell auftretende Probleme und berücksichtigt die Anforderungen, die an die Entwickler bei Implementierung und Design des neuen Systems gestellt werden.

2.2.1.2 Aufgabenanalyse nach Grässer

Eine weitere, die Aufgabenanalyse unterstützende Fragetechnik ist die Verwendung von *Warum*- und *Wie-Fragen* nach GRÄSSER (Szwillus 2012). Das Ziel dieser Technik ist das Herausarbeiten des Umstandes, wie Benutzer ihre Aufgaben erledigen. Als hauptsächliche Strukturelemente gilt es herauszufinden, wie Aufgaben in kleinere Unteraufgaben aufgeteilt werden (=Hierarchie) und ob Aufgaben wiederholt, optional oder alternativ durchgeführt werden (Zeitverhalten, temporale Relationen). Als Ergebnis entsteht eine Übersicht, wie die handelnden Personen ihre Aufgaben strukturieren, respektive eine hierarchische Aufgabenstruktur inklusive zeitlicher Beziehungen.

Eine Befragung läuft wie folgt ab. Das Interview ist zweigeteilt. Begonnen wird mit der Standardfrage, was die Aufgabe des Interviewten ist; was er tut und wie er es tut. Nachdem der Befragte anfängt zu berichten, wird dieser immer wieder in seinem Bericht unterbrochen mit

- „Warum tun Sie …?“ ► „… ich tue X, weil …“
- „Wie tun Sie …?“ ► „… ich führe Y durch Z aus …“

Auf diese Art sind Rückschlüsse auf die Struktur der Aufgabe möglich. *Warum*-Fragen geben Aufschluss über übergeordnete Schichten der Hierarchie, während *Wie*-Fragen Informationen über untergeordnete Schichten liefern. Um die temporalen Beziehungen zwischen den Aufgaben zu identifizieren, muss besonders auf Ausführungen wie

- „... ich kann dann entweder X1 oder X2 tun, abhängig von ...“ (Alternativen)
- „... ich starte mit Z1 und führe dann danach Z2 aus ...“ (Reihenfolge)
- „... manchmal muss ich Y tun, aber nur wenn ...“ (Optionalität)
- „ ... ich wiederhole dies, bis ...“ (Iteration)

geachtet werden.

Die Ergebnisse der beiden genannten Fragetechniken der Aufgabenanalyse bilden die Grundlage für das Erstellen von Aufgabenmodellen, welche im nächsten Kapitel detailliert betrachtet werden.

2.2.2 Aufgabenmodell

Die Erstellung eines Aufgabenmodells ist ein synthetischer Vorgang. Nachdem innerhalb der Aufgabenanalyse alle relevanten Facetten der Aufgabe identifiziert wurden, können nun die Informationen systematisch zusammengetragen werden, um ein Aufgabenmodell zu bilden. Zunächst

wird die durchzuführende Aufgabe abstrakt definiert und im Wurzelknoten verankert. Dann werden sukzessive die ermittelten Schritte als Kinderknoten hinzugefügt.

Ein einfaches Beispiel für ein Aufgabenmodell zeigt Abbildung 2.4. Es wird die Aufgabe „Ein Getränk an einem Automaten kaufen“ modelliert. Die auf den Knoten selbst bezogenen temporalen Relationen werden in Grün dargestellt, die für die Subknoten geltenden in Rot. Weiterhin sind die vom System durchzuführenden Aufgaben durch gepunktete Linien und Boxen repräsentiert, weil diese zur Bearbeitung der Aufgabe für den Anwender nur eine untergeordnete Rolle spielen.

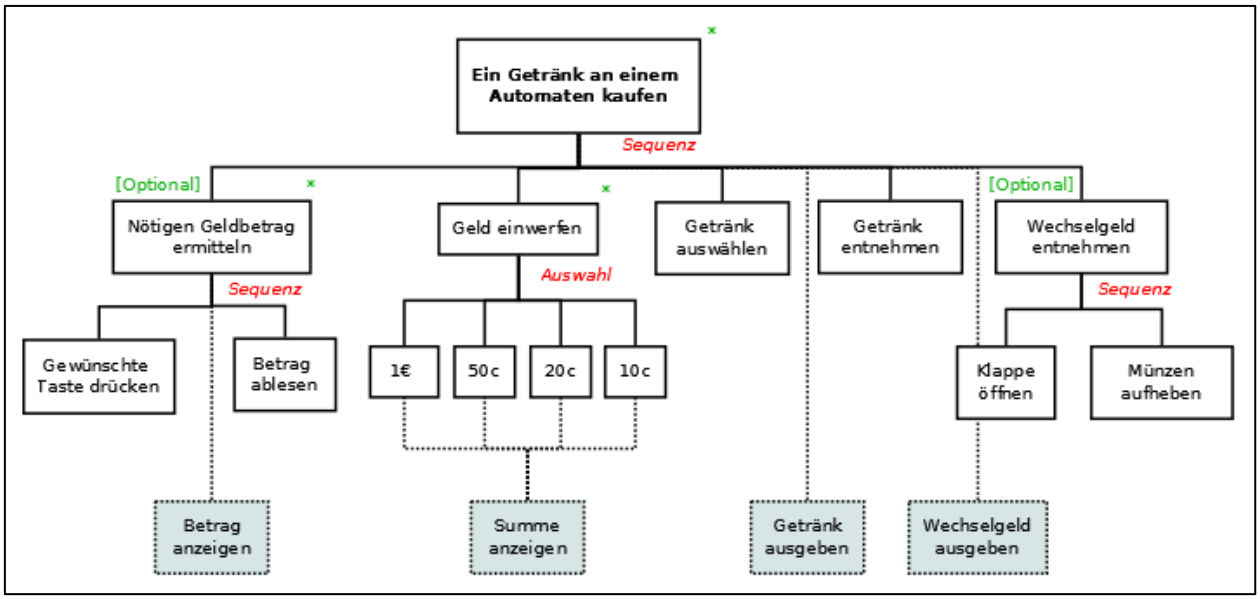


Abbildung 2.4 - Beispiel Aufgabenmodell

Wie man unmittelbar erkennen kann, wird am Wurzelknoten die mit dem Modell zu lösende Hauptaufgabe „Ein Getränk an einem Automaten kaufen“ platziert. Diese besteht hierarchisch aus den fünf Schritten „Nötigen Geldbetrag ermitteln“, „Geld einwerfen“, „Getränk auswählen“, „Getränk entnehmen“ und „Wechselgeld entnehmen“, die rekursiv nach unten weiter verfeinert werden. Diese Schritte sind durch eine *Sequenz* angeordnet, was bedeutet, dass sie in der angezeigten Reihenfolge von links nach rechts ausgeführt werden müssen. Ein Getränk kann z. B. nicht entnommen werden, wenn es weder bezahlt noch ausgewählt wurde. Ähnlich verhält es sich mit dem Wechselgeld. Der erste sowie der letzte Schritt sind dabei *optional*, da man den benötigten Geldbetrag für sein Wunschgetränk schon kennen kann, bzw. man gar kein Wechselgeld bekommt, weil man den Betrag passend eingezahlt hat.

Kennt man den nötigen Geldbetrag nicht, muss man diesen zunächst ermitteln. Dazu sind die Schritte „Gewünschte Taste drücken“ und „Betrag ablesen“ durch den Anwender zu erfüllen. Das System muss diesen zwischenzeitlich *anzeigen*. Das Sternchen am Knoten symbolisiert *Iteration*, d. h. der Anwender kann diesen Schritt beliebig oft durchführen, um die verschiedenen Preise zu ermitteln. Hat er den Preis seines Wunschgetränkes erfasst, muss das nötige Geld eingezahlt werden. Dazu können an diesem Automaten *1 €, 50 Cent, 20 Cent* oder *10 Cent* eingeworfen werden. Als temporale Relation wurde hier die *Auswahl* gewählt, da der Benutzer entscheiden kann, welche Münze er eingeben möchte. Für ein eventuelles Einzahlen mehrerer Münzen steht wieder die Iteration zur Verfügung. Der Automat muss zur Orientierung nach jedem Einwerfen die eingezahlte *Summe anzeigen*.

Ist genug Geld eingezahlt, kann das gewünschte *Getränk ausgewählt* werden. Der Automat wird dann veranlasst, das *Getränk auszugeben*, damit man es *entnehmen* kann. Ist ein offener Restbetrag vorhanden, muss der Automat nun das *Wechselgeld ausgeben*. Der Anwender kann dieses dann

entnehmen. An dieser Stelle wird deutlich, dass man das Modell dahingehend verfeinern könnte, die letzten beiden Schritte unter einem neuen Knoten „Vorgang abschließen“ anzuordnen, und diesen dann mit einer *zufälligen Sequenz* zu belegen (vgl. Abbildung 2.5). Dies wäre deshalb sinnvoll, weil es gleichgültig ist, ob der Anwender zuerst das Getränk oder zuerst das Wechselgeld entnimmt, wichtig ist nur das beides geschieht. Im Gegensatz dazu steht wieder der letzte Knoten „Wechselgeld entnehmen“, der in einer Sequenz angeordnet ist, weil man die *Münzen* nicht *aufheben* kann, wenn man zuvor nicht die *Klappe geöffnet* hat.

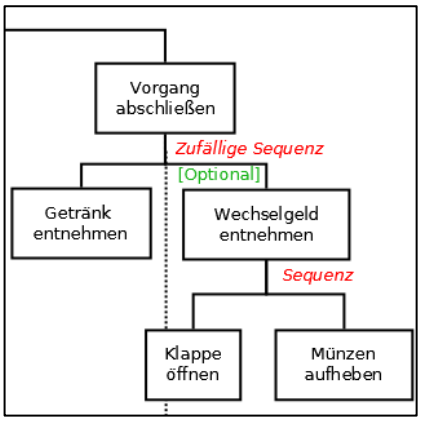


Abbildung 2.5 - Verändertes Aufgabenmodell mit zufälliger Sequenz

Dieses kleine Beispiel zeigt die Ausdruckskraft von Aufgabenmodellen, die durch die anderen identifizierten temporalen Beziehungen (s. Kapitel 2.1, Abschnitt „temporale Relation“) noch starke Erweiterungen bzw. andere Anwendungsmöglichkeiten erfahren. Am Wurzelknoten ist ebenfalls eine Iteration angedeutet, da ein Anwender durchaus auch mehrere Getränke einkaufen möchte. Des Weiteren bietet selbst dieses simple Beispiel noch Potenzial, einzelne Schritte detaillierter weiter zu differenzieren. So könnte auch für den Schritt „Getränk entnehmen“ eine Klappe vorhanden sein, die das Getränk schützt und erst geöffnet werden muss. Zwar würde die Granularität damit immer weiter zunehmen, aber man erkennt sehr schnell auf welche Level der Komplexität das Modell bei schwierigeren Aufgaben anwachsen kann.

2.3 User Interfaces

Dieses Kapitel der Arbeit ist den Benutzungsschnittstellen (engl. *User Interfaces)* gewidmet. Die Art der Gestaltung von Benutzungsschnittstellen spielt die entscheidende Rolle für die Akzeptanz bei den Benutzenden und sie kann – bei richtiger Umsetzung – einen hohen Grad an Usability und Komfort erreichen. Benutzungsschnittstellen verkörpern den „Unterbau“ der später (Kap. 2.4) betrachteten interaktiven Displays.

2.3.1 Klassische User Interfaces

Dieses Unterkapitel beschäftigt sich mit klassischen User Interfaces, wie sie seit Jahrzehnten für die Bedienung von Computern eingesetzt werden. Diese Art der Bedienung ist weit verbreitet und es ist noch heute möglich, mit Hilfe dieser Schnittstellen mit modernen Betriebssystemen zu interagieren.

2.3.1.1 Kommandozeilenbasierte User Interfaces

Die kommandozeilenbasierten User Interfaces (engl.: *Command Line Interface,* kurz CLI) stellen die erste Generation der Benutzungsschnittstellen dar. Die intuitiven Interaktionsmöglichkeiten mit dieser Schnittstelle sind stark eingeschränkt. Nach dem Start findet sich der Benutzer in einer in Abbildung 2.6 dargestellten Ansicht wieder:

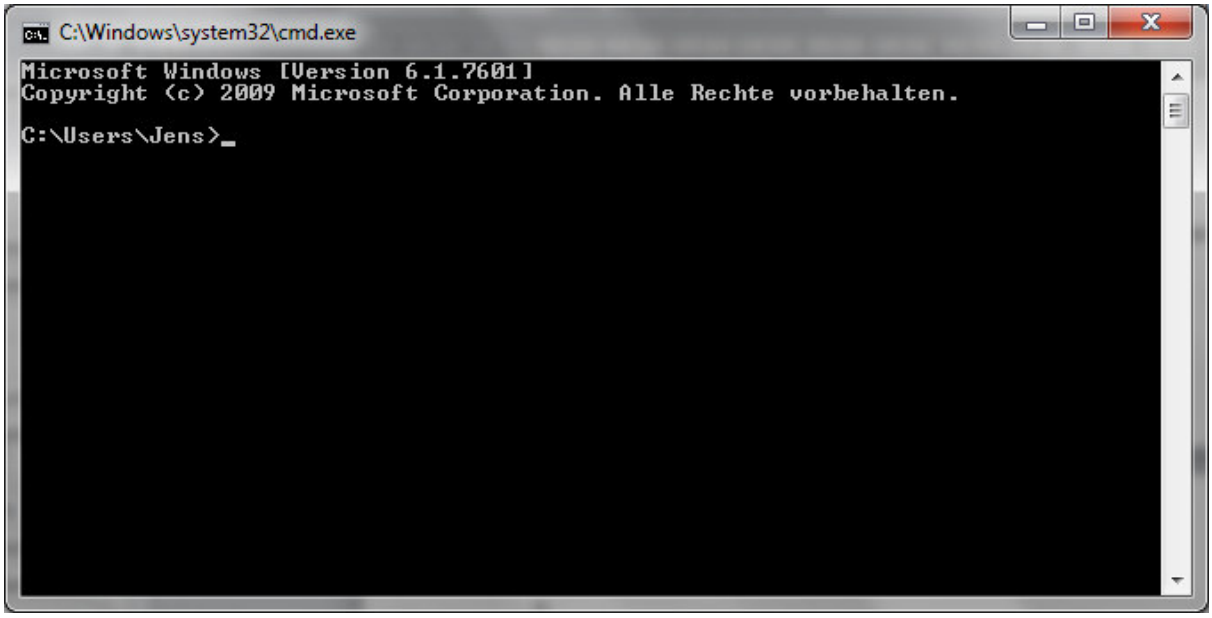


Abbildung 2.6 - Kommandozeilenbasiertes User Interface (Windows 7)

Die einzige Information, die der Benutzer erhält, ist welcher User gerade angemeldet ist. Jegliche grundlegenden Befehle müssen vom Benutzer aus dem Gedächtnis abgerufen werden, um mit dem System zu interagieren. Der Dialog gleicht einem Frage/Antwort Spiel, bei dem das System den Benutzer fragt, was es als nächstes tun soll, und der Benutzer das gewünschte Kommando als Antwort eingibt. Ob der Nutzer eine Rückmeldung über sein eingegebenes Kommando erhält, ist vom ausgeführten Programm abhängig.

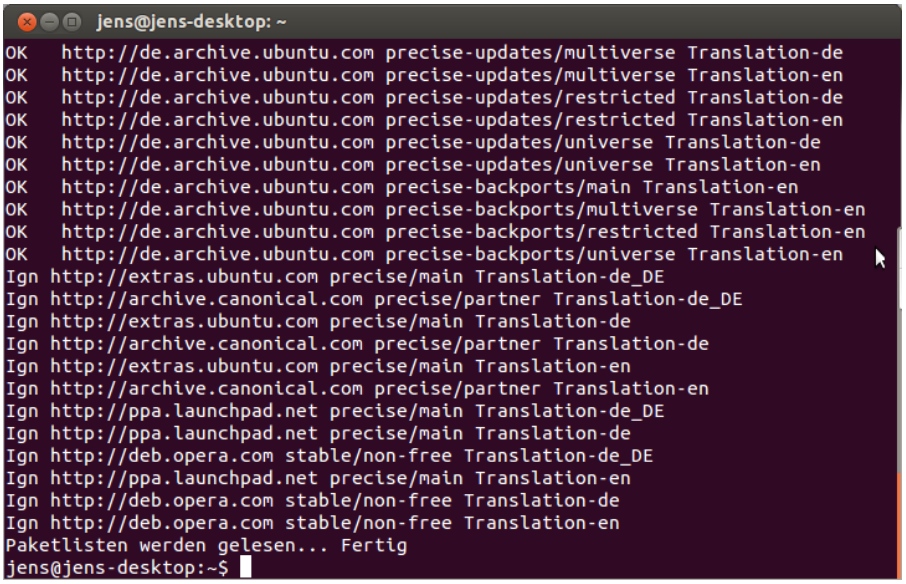


Abbildung 2.7 - Kommandozeile in Ubuntu Linux (Terminal)

Noch heute werden mit Hilfe dieser Eingabemethode viele Kommandos unter aktuellen Unix-basierten Distributionen durchgeführt, wie Abbildung 2.7 bespielhaft für Ubuntu Linux zeigt. Zu den wichtigsten Funktionen gehören das Installieren von Software sowie das Ausführen von Programmen mit Root-Berechtigungen, die auf Betriebssystemebene Uneingeschränktheit zur Programmlaufzeit erlauben. Eine weitere wichtige Anwendung ist das Übergeben von Parametern zum Programmstart. Erfahrene Benutzer schätzen diese Art der Interaktion, weil so die direkte Eingabe von Kommandos ermöglicht wird – es wird ausgeführt, was eingegeben wurde.

Da diese Art der Bedienung für die meisten Menschen dennoch nicht sehr ansprechend respektive intuitiv ist und nicht zu unterschätzendes Expertenwissen voraussetzt, haben sich im Laufe der Zeit stärker grafisch orientierte Oberflächen durchgesetzt, die eine bessere Assoziation mit der realen Welt realisieren.

2.3.1.2 Grafische User Interfaces

Grafische Benutzeroberflächen (engl.: *Graphical User Interface*, kurz GUI) sind frühestens seit der Entstehung von Apple, spätestens aber seit der Einführung von Microsoft Windows de facto Standard heutiger Betriebssysteme. Charakteristisch für diese Art von Oberflächen ist die Verwendung von manipulierbaren Objekten, mit denen Interaktionen durchgeführt werden können. Erreicht werden diese Objekte durch einen sogenannten *Pointer* (engl. für Zeiger), der üblicherweise mit der Computermaus in einem zweidimensionalen Aktionsraum bedient wird. Dabei wird die mechanische Bewegung in ein elektrisches Signal umgesetzt, das analog zu der Bewegungsrichtung der physischen Maus den Pointer auf dem Bildschirm bewegt. Ein Beispiel für eine grafische Schnittstelle zeigt Abbildung 2.8:

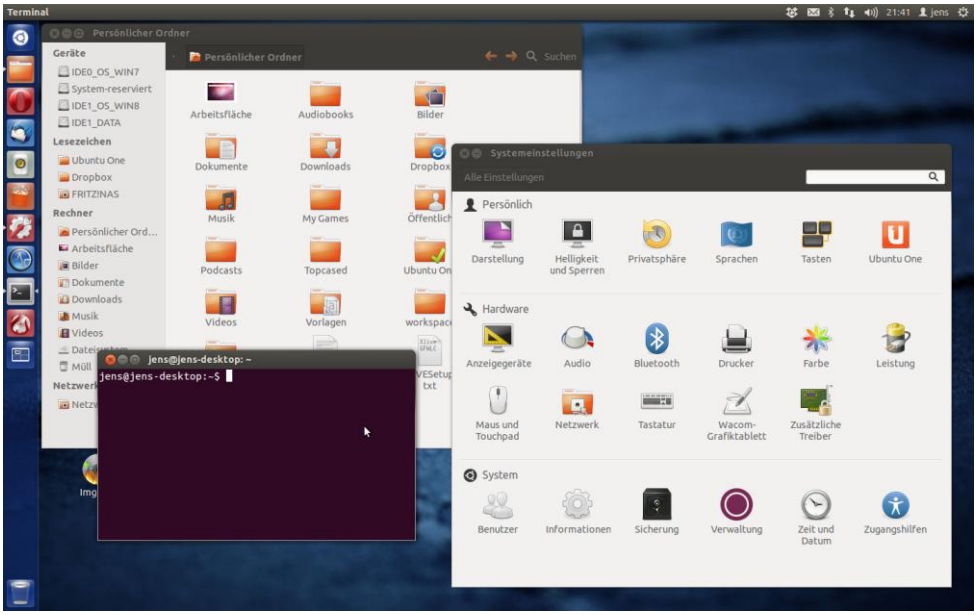


Abbildung 2.8 - Graphical User Interface (Ubuntu Unity Desktop)

Typische manipulierbare Objekte sind Fenster, die mit Hilfe des Pointers verschoben, in ihrer Größe verändert, am Bildschirm ausgerichtet und geöffnet oder geschlossen werden können. Die unterste Ebene dieser Fenster bildet die Arbeitsfläche, der sogenannte *Desktop,* der metaphorisch die Oberseite eines Schreibtisches symbolisiert[10]. Weiterhin kann es verschiedene virtuelle Objekte geben, die unterschiedliche Arten von Dateien repräsentieren. Diese Repräsentation wird durch die Benutzung sogenannter *Icons* (griechisch für Bild) ermöglicht, die eine Assoziation mit der realen Welt ermöglichen, und auf dem Desktop oder innerhalb von Fenstern verankert sind. So gibt es zum Beispiel den virtuellen Papierkorb, in dem Objekte zum Löschen abgelegt werden können, indem sie mit dem Pointer darauf gezogen werden. Oder virtuelle Ordner helfen beispielsweise dabei, auf der Festplatte befindliche Dateien sinnvoll zu strukturieren – analog zur realen Welt.

Wenn das Betriebssystem grafische Unterstützung bietet, arbeitet jede moderne Software mit GUIs. Mit ihrer Hilfe werden häufig benutzte Kommandos wiedererkennbar und leicht ausführbar im Aktionsraum

[10] Die Desktopmetapher wurde maßgeblich durch den Informatiker Alan Kay in den 1970er Jahren am Palo Alto Research Center begründet. Erster Einsatz war auf einem Apple Macintosh Computer.

des Benutzers platziert, oder mit Hilfe von ausklappbaren Menüs strukturiert zugänglich gemacht. Dabei gibt es bestimmte Icons, die softwareübergreifend dieselben Funktionen repräsentieren. Klassisch zu nennen ist hier das so gut wie überall bekannte Diskettensymbol, mit dem im Programm geöffnete Dateien gespeichert werden. Gleichermaßen verhält es sich mit dem Symbol einer leeren Seite für das Erzeugen einer neuen, leeren Datei oder dem Ordnersymbol zum Öffnen bereits vorhandener Dateien. Diese und viele weitere klassische Metaphern, wie das Druckersymbol oder der Brief, veranschaulicht Abbildung 2.9:

Abbildung 2.9 - Typische Symbolik am Beispiel Libre Office 3.4

2.3.2 Moderne User Interfaces

Seit einigen Jahren hat sich ein Wandel der Benutzungsschnittstellen vollzogen. In vielen Lebensbereichen haben berührungsempfindliche Displays, sogenannte „Touchscreens“, Einzug in das tägliche Leben erhalten. Sie bilden die nächste Generation von Benutzungsschnittstellen und begegnen uns als Auskunfts- oder Buchungsterminals auf Flughäfen, Bahnhöfen oder Messen, bei der Benutzung von Geldautomaten oder als Informationsgeber in Supermärkten. Auch der Markt für mobile Endgeräte wie Smartphones oder Tablet Computer bewegt sich weg von der konventionellen Bedienung mit Tasten hin zur direkten Manipulation mit den Fingern.

Dieser Abschnitt stellt aus diesem Grund moderne Formen der Interaktion vor, die den de facto Standard der Bedienung, wie er im vorherigen Abschnitt erläutert wurde, abzulösen versuchen. Sie werden eine tragende Rolle bei der Entwicklung des späteren Konzepts übernehmen.

2.3.2.1 Natural User Interfaces

> *„Ein Natural User Interface (NUI) beschreibt ein Interface, welches unmittelbar durch einen oder mehrere Sinne des Nutzers bedient wird. Es bildet somit einen Oberbegriff für viele Arten der Interaktion an der Mensch-Maschine-Schnittstelle.“* (Bollhoefer et al. 2009)

Diese Definition von BOLLHOEFER, MEYER ET AL. beschreibt sehr gut das weite Spektrum, das Natural User Interfaces abdecken. Während traditionell Computer mit Hilfe diverser Eingabegeräte bedient wurden, ermöglichen Natural User Interfaces den unmittelbaren Zugang zum System ohne weitere Hilfsmittel wie Maus, Tastatur oder anderer Fernbedienungen. Die Facetten reichen von Touchscreen-Terminals in Banken oder Bahnhöfen, Spracherkennung bei Callcenter-Services, Bewegungserkennung in den neuen iPods zum Skippen von Tracks, Fingerabdruck- oder Iris-Scans bei Zugangskontrollen, Multi-Touch-Bedienung des Apple iPhone bis hin zur zukünftigen Systemsteuerung durch direkte Impulse der Synapsen im menschlichen Gehirn. (Bollhoefer et al. 2009)

Die Autoren unterscheiden dabei *aktive* und *passive* Natural User Interfaces. Bei den aktiven NUIs tritt der User direkt in Interaktion mit dem System, z. B. durch seine Finger am Multitouch-Bedienfeld. Mit passiven NUIs tritt der Benutzer jedoch nur indirekt in Interaktion, wie bspw. beim Iris-Scan, wo nur ein Abdruck der Augen eingescannt wird. Da es in dieser Arbeit um die Erstellung von Aufgabenmodellen geht, die einen aktiven Vorgang repräsentiert, wird der Schwerpunkt auf aktive NUIs gelegt.

2.3.2.2 *Tangible User Interfaces*

Tangible User Interfaces (etwa *greifbare Benutzungsschnittstellen*) erweitern den Aktionsraum der mit dem System interagierenden Benutzer um eine neue Komponente. Sie machen die Objekte innerhalb der Schnittstelle real anfassbar, indem diese in einem definierten Bereich durch Platzieren mit den Händen integriert werden können. Diese Art der Benutzungsschnittstellen kommt vorwiegend auf den sogenannten Tabletops (s. Kapitel 2.4.3) zum Einsatz, da sie meist eine ebene Fläche für das Verwenden von Objekten erfordern.

Einfach ausgedrückt bedeutet das, dass beispielsweise kleine Figuren auf der berührungsempfindlichen Oberfläche des Tabletops arrangiert werden können. Der in den Tabletop eingebettete Computer kann dann über die vorhandene Sensorik (z. B. Kameras) diese Figuren auch als solche wahrnehmen und erkennen. Die Erkennung erfolgt durch sogenannte *Fiducials* (s. Abbildung 2.10), die unter den realen Objekten (Figuren, Formen) angebracht werden. Softwareseitig werden diese durch das Framework reacTIVision des TUIO Protokolls erkannt, dass in den gängigsten Programmiersprachen verfügbar ist (Kaltenbrunner & Bencina 2007) (Kaltenbrunner 2009).

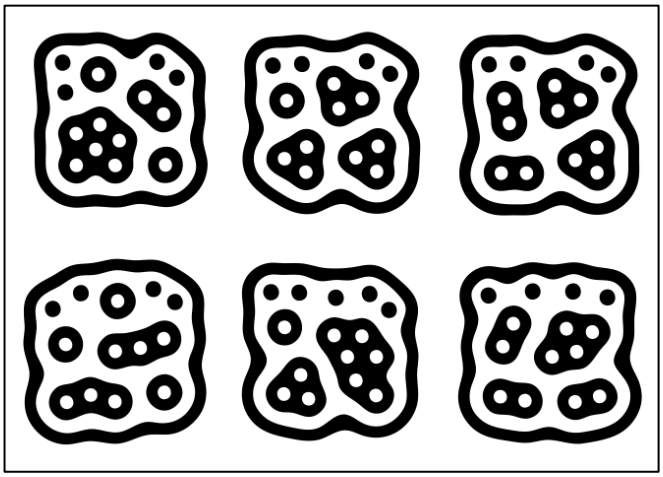


Abbildung 2.10 - Beispiel Fiducials „amoeba“ des reacTIVision Frameworks[11]

Auf diese Weise kann die Position der Figur und seine Art, also um was für eine Figur es sich handelt, bestimmt werden. Bewegungen der Figuren können interpretiert oder aufgezeichnet werden, solange der Kontakt der Objekte mit der Oberfläche nicht verloren geht. Die Fiducials fungieren dabei wie eine Art zweidimensionaler Barcode, der von den optischen Sensoren des Tisches gelesen und zugeordnet werden kann. Der Abgleich erfolgt im Hintergrund über eine Datenbank, in der die Eigenschaften des durch das Fiducial gekennzeichnete Objekt hinterlegt sind. Somit ist die Benutzung mehrere verschiedener Objekte möglich, oder die Nutzung mehrere gleicher bzw. ein Kombination aus beidem.

Metaphorisch entsteht auf diese Weise eine direkte Brücke zwischen realen und virtuellen Objekten, da eine Handlung an einem realen Objekt die gleiche Handlung (innerhalb der Funktionalität des Aktionsraums) am virtuellen Objekt impliziert. Wird beispielsweise die Position des realen Objekts verändert, ändert sich diese entsprechend auch für das virtuelle Objekt; dreht man das physische Objekt, dreht sich auch das digitale.

[11] Quelle: (Reactable Systems S.L. 2009), S. 1

2.4 Interaktive Displays

Hier soll nun eine Auswahl interaktiver Displays vorgestellt werden, die Interaktion und Anzeige in einem Gerät vereinen.

2.4.1 Touchscreen

Touchscreens sind die bekanntesten Vertreter der interaktiven Displays. Sie waren die ersten Geräte, die Eingabe direkt am Bildschirm ohne weitere Hilfsmittel ermöglichten. Dabei werden die Eingaben eines Menschen durch den Einsatz von resistiver, kapazitiver oder induktiver Technik erkannt. Die beiden ersten Varianten ermöglichen die Eingabe nur mit Hilfe der Finger, die induktive Variante braucht weiterhin einen speziellen Stift o. Ä. um Eingaben zu erkennen. Durch Einblenden einer softwarebasierten Maske wird Zeicheneingabe sowie die Auswahl von Optionen ermöglicht. Die Entwicklung dieser Technik war essentiell und bildet die Grundlage vieler weiterer technischer Entwicklungen auf dem Gebiet der Eingabegeräte.

2.4.2 Smartphone / Tablet

Der Markt mobiler Endgeräte wechselt in der jüngsten Zeit stark von den klassischen, mit Tasten bedienten Mobiltelefonen hin zu sogenannten Smartphones, die weit mehr Funktionalität als das Telefonieren und Versenden von Kurznachrichten (SMS) bieten. Mit ihnen wird es möglich, das Internet ständig verfügbar mit sich zu führen (solange es eine Datennetzverbindung gibt). Über einen angeschlossenen Onlinemarkt können Apps (kurz für *Applications*) zur nahezu „unbegrenzten“ Funktionserweiterung nachinstalliert werden.

So wird das Telefon (bei passender Installation in den Gebäuden) fähig, elektronische Geräte aus der Ferne zu steuern (Strom, Licht, Kaffeemaschine) oder den Status von Anlagen zu überwachen (z. B. Photovoltaik, Heizungsanlage). Es kann Bestellungen aufgeben, dient zur Navigation im Terrain oder gibt Informationen zu mit der integrierten Kamera gemachten Fotos.

Abbildung 2.11 - Smartphones (Quelle: Apple/Google)

Charakteristisch für ein solches Smartphone ist, dass es durchweg weitestgehend ohne Tasten auskommt. In den meisten Fällen ist lediglich eine Taste zum Einschalten des Gerätes vorhanden, und die komplette Steuerung erfolgt direkt am Display des Gerätes, bzw. mit Hilfe von Sensortasten, die aber

keine erkennbare Erhebung besitzen und somit im ausgeschalteten Zustand nicht als Tasten identifizierbar sind. Tablets hingegen stellen eine physisch vergrößerte Variante der Smartphones dar und sind in erster Linie nicht zum Telefonieren konzipiert. Da auf Ihnen jedoch dieselben Plattformen (Google Android, Apple iOS, Microsoft Windows 8 Phone / Windows RT) laufen, gelten für sie die gleichen Erläuterungen wie für Smartphones.

2.4.3 Multi-Touch Tabletops

Unter Multi-Touch Tabletops (kurz: MTT) werden Computer verstanden, die das Multi-Touch-Bedienkonzept[12] auf einen großflächigen Bildschirm bringen, der in eine Tischplatte eingebettet ist. Auf diese Weise können eine oder mehrere Personen die auf dem Computer laufenden Anwendungen direkt auf dem Tisch durch ihre Hände bedienen (Müller-Tomfelde 2010).

Abbildung 2.12 - Multitouch Tabletop an der Universität Paderborn: MISTER T[13]

Durch Gestensteuerung lassen sich dargestellte Objekte drehen, ihre Position oder ihre Größe verändern, in die Arbeitsfläche hinein oder aus ihr hinauszoomen etc. Auch die Benutzung von realen Objekten wird durch die Verwendung von TUIs möglich, wobei die horizontale Lage des Bildschirms die entscheidende Rolle spielt. So lassen sich etwa an Partner Objekte durch Verschieben von Figuren übergeben, die diese dann weiter verwenden können und bspw. nach Veränderung wieder zurückgeben; oder schnell und direkt verschiedene Anordnungen oder Gruppierungen von Objekten erstellen, indem jene Figuren real auf dem Tisch angeordnet werden.

Nachdem nun die fachlichen Grundlagen für die weiteren Untersuchungen hinreichend beschrieben wurden, werden im nächsten Abschnitt der Arbeit bereits vorhandene Ansätze zur Aufgabenmodellierung evaluiert, bevor im Anschluss mit der Konzeption begonnen werden kann. Viele Aspekte können sicherlich in ihrer Grundidee übernommen werden; idealerweise entsteht im späteren Konzept eine Kombination aller genannten Vorteile der untersuchten Editoren innerhalb einer Anwendung, die in Verbindung mit aktuellen technologischen Neuerungen den Entwicklungsprozess von Aufgabenmodellen verstärkt kollaborativ in Szene zu setzen vermögen.

---

[12] Unter Multi-Touch versteht man die Eigenschaft, einen Touchscreen mit mehr als nur einem Finger zu bedienen, und ihn so für weitaus mehr als nur einfaches Klicken oder Schreiben zu benutzen. Diese Technik ist elementar für die Verwendung von Gestik auf heutigen Consumer-Endgeräten.

[13] Internetquelle: MAGIC - http://www.cs.uni-paderborn.de/uploads/RTEmagicC_DSC_1138.JPG.jpg

3 Untersuchung vorhandener Werkzeuge und Techniken

Für die Entwicklung eines eigenen Konzeptes ist es unerlässlich, bereits erbrachte Leistungen anderer Wissenschaftler in die eigenen Betrachtungen mit einzubeziehen. Weder die Aufgabenmodellierung, noch die Benutzung natürlicher Benutzungsschnittstellen sind völliges Neuland; jedoch steigt die Zahl der Veröffentlichungen auf beiden Gebieten immer weiter an – besonders im Bereich Multi-Touch, da es dort nahezu unendlich viele Anwendungsmöglichkeiten zu geben scheint.

Dieser Abschnitt der Arbeit untersucht die wichtigsten Entwicklungen aus beiden Gebieten und bewertet sie im Hinblick auf Funktionalität und Usability, um aus den gewonnenen Erkenntnissen eine ideale Grundlage für das Konzept in Kapitel 4 aufzubauen. Zunächst wird die Aufgabenmodellierung näher behandelt, im Anschluss Methoden der Interaktion sowie der Diagrammbearbeitung auf MTTs analysiert und die daraus resultierenden Ergebnisse in einem Zwischenfazit einander gegenübergestellt.

3.1 Evaluation von Werkzeugen zur Aufgabenmodellierung

Es sind bereits verschiedene interessante Ansätze für das Erstellen von Aufgabenmodellen öffentlich bekannt und haben mehr oder weniger Beachtung erhalten. In diesem Kapitel soll es darum gehen, ausgewählte Ansätze auf ihre Vor- und Nachteile zu untersuchen und im Hinblick auf die Adaption für das interaktive Display zu bewerten. Dabei geht es ebenso um Techniken der Visualisierung und Präsentation wie um die Art und Funktionalität, auch komplexe Aufgabenmodelle mit bestmöglichem Blick auf die Realität zu gestalten. Die bei der Untersuchung gemachten Erfahrungen werden dokumentiert und kritisch bewertet, sodass für das in Kapitel 4 entstehende Konzept wichtige Elemente identifiziert und genutzt werden können. Allen Ansätzen gemein ist das Grundverständnis von Aufgaben als eine Hierarchie untergelagerter Subaufgaben. Jedoch bieten die Ansätze unterschiedliche Erweiterungen, um die Realität besser abbilden zu können. Die Bewertung der einzelnen Werkzeuge erfolgt komprimiert am Ende eines jeden Unterkapitels, und in Kapitel 3.3 werden die Ergebnisse dann noch einmal reflektierend im Zusammenhang dargestellt.

3.1.1 Visual Task Model Builder

Der Visual Task Model Builder (kurz VTMB) entstand im Rahmen der Dissertation von Matthias Biere im Jahre 1999 an der Universität Paderborn. Der hervorgegangene Editor bietet ein einfaches Objekt-Modell, das auf einer Klassenhierarchie basiert und auf diese Weise die Effekte von Aufgaben durch Manipulation der Objektattribute ermöglicht. Weiterhin werden Vor- und Nachbedingungen unterstützt, welche bspw. Umwelteinflüsse (die der Benutzer nicht beeinflussen kann) simulieren können. Ein Rollenmodell fehlt, sodass VTMB keine verschiedenen Rollen unterscheiden kann. VTMB konzentriert sich auf die Zerlegung von Aufgaben und deren Ausführungsregeln. Mitgeliefert wird ein Simulator, mit dem man die Ausführung der Aufgabe auf Richtigkeit überprüfen kann (Szwillus 2011). Aktuell findet keine aktive Weiterentwickelung von VTMB statt, somit erfolgt die Evaluation auf Basis der zuletzt verfügbaren Version 1.2, Build 761.

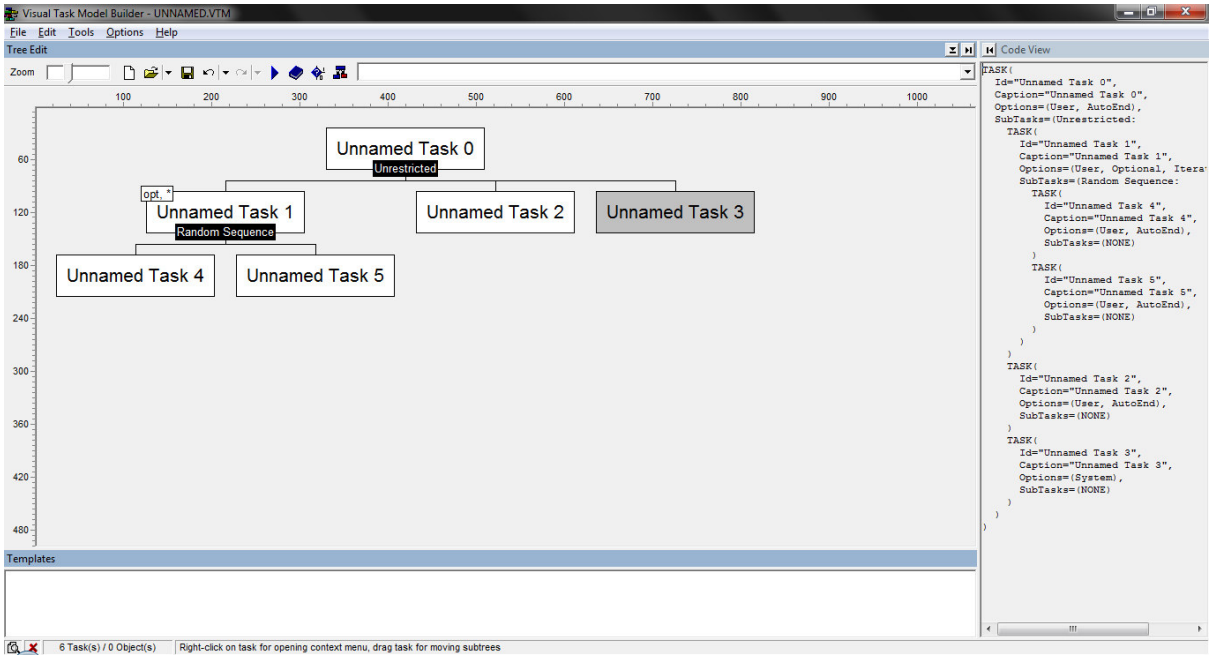


Abbildung 3.1 - VTMB Beispiel

Die Oberfläche von VTMB gestaltet sich recht einfach. Bei Programmstart enthält der Aufgabenbaum nur den Wurzelknoten, den man durch umbenennen zu der zu modellierenden Aufgabe umdefinieren kann. Über das Kontextmenü des Knotens lassen sich nun Subaufgaben hierarchisch unterhalb im Baum erzeugen. Auffällig ist, dass hierfür keine Schaltflächen mit entsprechender Symbolik bereitstehen, die den Benutzer unterstützen. So müssen immer mindestens zwei Klicks, begleitet von einer Menüauswahl für das Anlegen neuer Knoten verwendet werden. Nach dem Anlegen eines neuen Knotens öffnet sich das folgende Fenster, in dem die Eigenschaften des Knotens definiert werden können:

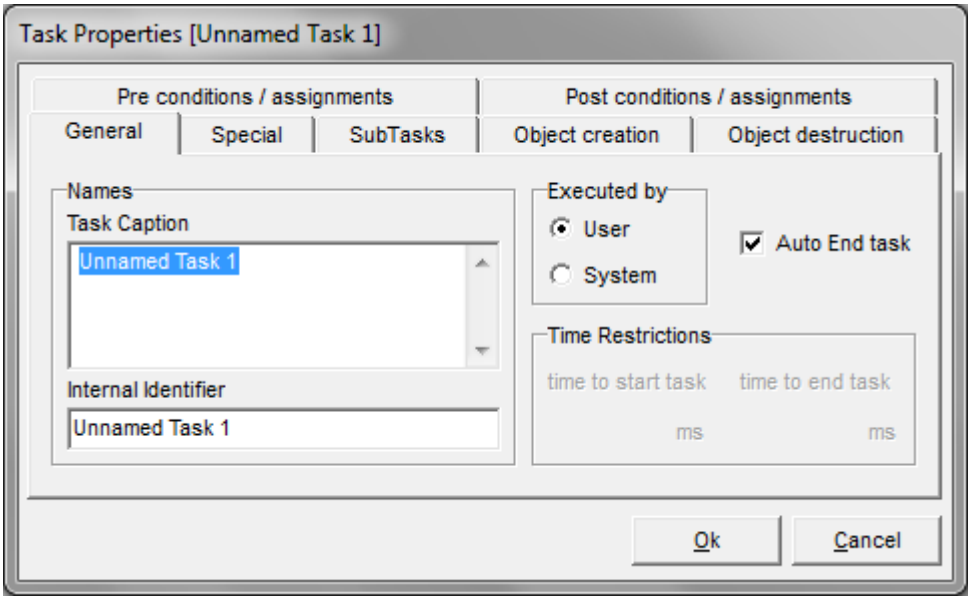


Abbildung 3.2 - VTMB Task Properties

Hier können für die Aufgabe mögliche Parameter konfiguriert werden. VTMB unterscheidet dabei zwischen Useraufgaben, die automatisch bei Durchführung beendet werden können, sowie Systemaufgaben, die einer Zeitbeschränkung unterliegen können. Systemaufgaben können nicht durch den User beeinflusst werden und dauern meist eine bestimmte Zeit, die das System zum Antworten benötigt; sie werden im Baum entsprechend durch graue Hinterlegung gekennzeichnet.

Weiterhin können die temporalen Relationen für die Aufgabe unter den Reitern *Special* und *Subtasks* festgelegt werden, wobei ersteres für spezielle Parameter direkt am Knoten genutzt wird und letzteres für die zeitliche Beziehung der untergeordneten Aufgabenschritte verantwortlich ist. So kann für die Aufgabe bestimmt werden, ob sie mehrfach ausgeführt werden kann (Iteration) und wie oft, oder ob sie optional durchführbar ist. Auch können Bedingungen für den Austritt aus der Iterationsschleife oder das Überspringen der Aufgabe festgelegt werden. Allgemein fällt auf, dass diese Optionen auch für den Wurzelknoten eingestellt werden können, was zunächst wenig sinnvoll erscheint und auch zu Fehlern führen würde. Setzt man die Parameter für den Wurzelknoten, wird man jedoch direkt durch einen im Hintergrund laufenden Parser auf seinen Fehler hingewiesen, was sich als sehr positiv darstellt. Jedoch würde ein Verbieten der Optionen am Wurzelknoten erst gar keine Fehler entstehen lassen.

Im Reiter *Subtasks* werden die untergeordneten Aufgaben aufgelistet. Hier kann die Art der Ausführung dieser Subaufgaben festgelegt werden. Möglich sind die temporalen Relationen *Unrestricted*, *Sequence*, *Random Sequence*, *Parallel* oder *Selection*. Auch befindet sich an dieser Stelle die einzige Möglichkeit, die angezeigte Reihenfolge der Subaufgaben durch Ziehen zu verändern. Die festgelegten Eigenschaften werden im Graphen unmittelbar am Knoten angezeigt, was dem Benutzer bei der Orientierung hilft. Weiterhin muss keine Symbolik für die temporalen Relationen gelernt werden, wie dies bei anderen Umgebungen der Fall ist (bspw. CTTE, s. nächstes Kapitel).

Schlussendlich können in den Eigenschaften Objekte bei Erreichen der Aufgabe erzeugt oder zerstört, sowie Vor- und Nachbedingungen des Modells angepasst werden. Diese Eigenschaften stellen die erweiterten Funktionen von VTMB dar. Zuvor erschaffene Objekte können nun bei Beendigung dieser Teilaufgabe zerstört werden, oder neue Objekte bestimmter Klassen ebenso erzeugt werden. Dazu bedarf es der vorherigen Festlegung dieser Klassen und ihrer Eigenschaften in dem dafür vorgesehenen *Class Manager.* Bei Erzeugung von Objekten können die Initialwerte der im Class Manager definierten Objektattribute festgelegt werden. Außerdem können, wie bereits erwähnt, Vor- bzw. Nachbedingungen gepflegt werden, die dazu führen, dass Teilaufgaben nur unter bestimmten Umständen gestartet oder beendet werden können. Gleichzeitig können aber auch Zuweisungen an Variablen vorgenommen werden, um Bedingungen zu ändern. Diese Eigenschaften grenzen VTMB von anderen Umgebungen ab und sollten unter keinen Umständen im Konzept fehlen, da sie fundamentale Funktionen in den Modellierungsprozess integrieren.

Das Anordnen der Unteraufgaben im Baum erfolgt sequentiell, das heißt das zuerst angelegte Kind erscheint links, alle weiteren Geschwister werden zunächst horizontal rechts in Anlegereihenfolge auf gleicher Ebene angeordnet. Ein nachträgliches Verändern der Reihenfolge der Knoten ist jedoch nur versteckt über das Eigenschaften-Menü des übergeordneten Knotens möglich, was den Benutzer fast schon dazu zwingt, sich bereits im Vorfeld Gedanken über die Schritte einer Aufgabe und deren Abfolge zu machen – was nicht unbedingt einen Nachteil darstellt. Problemlos hingegen gestaltet sich das Unterordnen von Blättern oder ganzen Teilbäumen zu übergeordneten Knoten direkt im Baum. Für das Arbeiten an einem Multi-Touch-Tisch sollten jedoch beide Varianten ohne Zwischenwege im Baum durchführbar sein. Auch das Verschieben des Graphen bzw. der Arbeitsfläche ist nur über die Scroll-Balken am Rand möglich; dies sollte auf dem Touchscreen durch entsprechende Gestik ersetzt werden.

Das Ein- und Ausklappen der Teilbäume erfolgt problemlos und ist eine gute Möglichkeit, schnell einen abstrakteren Überblick über das Modell zu bekommen. Diese Funktion wird mit Hilfe des Cursors angezeigt, der sich in einen gelben, nach oben gerichteten Pfeil verwandelt, wenn man mit der Maus den Ast entlang fährt. Diese Methode ist ungeeignet für Touchscreens, weil keine Zeiger zur optischen Unterstützung zur Verfügung stehen. Besser wäre hier eine bekannte Symbolik am Knoten wie ein

Minuszeichen, um die Ansicht zu reduzieren. Diese Symbolik wird dann aufgegriffen, wenn der Baum eingeklappt ist: Nun sind an jedem eingeklappten Knoten Pluszeichen, um die Bäume wieder zu erweitern. Auch die Farbe der gewählten Relation ändert sich dabei, was den Benutzer visuell unterstützt, eingeklappte Knoten zu erkennen. Diese Methodik ist sehr für den Touchscreen geeignet, sollte jedoch konsequent umgesetzt werden. Auch wäre eine Schaltfläche z. B. am Rand optimal, mit der man alle Knoten einer Ebene gleichzeitig einklappen könnte. Diese Überlegungen werden später in das Konzept einfließen.

Zoomen im Baum ist über einen Schieberegler realisierbar, jedoch arbeitet die Funktion nicht intuitiv so wie man es von heutiger Software gewöhnt ist. Die Ausrichtung des Graphen ist relativ zur oberen linken Ecke der Fläche, so dass man unter Umständen beim Hinein- oder Hinauszoomen verstärkt nachjustieren muss, da sich der Fokus ständig verschiebt. Weiterhin ist das Hinauszoomen in eine Stufe möglich, in der man zwar einen Überblick über die Komplexität des Modells bekommt, aber keine Beschriftungen mehr wahrnehmen kann. Besser wäre hier die Verwendung einer Miniaturkarte, die die Position im Graphen anzeigt und den Überblick über den Graphen permanent sichtbar macht. Auf diese Weise könnte auf Zoomstufen der eben beschriebenen Art verzichtet werden, sodass die Lesbarkeit gewahrt bleibt. Für eine Multi-Touch Anwendung könnte die Miniaturkarte beispielsweise während des Zoomens kurzzeitig eingeblendet werden. Idealerweise sollte man sie aber auch fixieren können.

Positiv ist noch zu erwähnen, dass das Modell im Anschluss in einer Simulationsumgebung, die auf dem Interpreter MODSN aufsetzt, auf Richtigkeit überprüft werden kann. Es können die Teilaufgaben basierend auf ihren temporalen Relationen Schritt für Schritt durchsimuliert werden. Wiederholungen sowie das Überspringen von Aufgaben sind (nach entsprechender Kennzeichnung im Modell) auswählbar, ebenso wird die zeitliche Simulation von Systemaufgaben unterstützt. Vor- und Nachbedingungen werden separat angezeigt. Kommt man in der Simulation an einem Schritt nicht mehr weiter, ist das Modell nicht schlüssig und kann dann auf Unstimmigkeiten untersucht werden. Bei Erfolg konnte man jeden einzelnen Schritt bis zum Ende der Simulation einwandfrei durchführen. Diese Art der Simulation eignet sich durchaus für eine interaktive Anwendung und kann so oder in ähnlicher Form in das Konzept Einzug erhalten.

Am Ende eine kurze Zusammenfassung der gewonnenen Erkenntnisse:

Pro	Contra
+ Klassenmanager für das Objektmodell + Simulator MODSN für das Aufgabenmodell + Hintergrundparser zur Vermeidung von Benutzerfehlern + Ein- und ausklappbare Äste im Baum + Templates nutzbar zur schnellen Erstellung sich ähnelnder Teilaufgaben + Aufgabenmodellquellcode permanent sichtbar	– Keine permanente Gesamtübersicht über den Baum (Miniatur) – Zoomen nur eingeschränkt möglich – Keine Symbolik zur Steuerung, meist nur über Kontextmenüs durchführbar

Tabelle 3.1 - Bewertung: VTMB für interaktives Display

3.1.2 ConcurTaskTrees Environment

Den wohl bekanntesten Ansatz der Aufgabenmodellentwicklung stellt das ConcurTaskTree Environment dar, welches von Fabio Paternò am CNR-ISTI in Pisa, Italien initiiert wurde. Diese Umgebung und die dazugehörige Art der Visualisierung und Definition von Aufgabenmodellen haben sich bis in die Industrie etabliert und werden in der Praxis zur Modellierung eingesetzt (Szwillus 2011). Bedeutend ist die Methodik der kooperativen Aufgabenmodelle, bei der Aufgaben unterschiedlicher Akteure im Modell in einen weiteren, übergeordneten Metamodell miteinander in Beziehung gesetzt werden. Diese Eigenschaft ist besonders für das gemeinsame Arbeiten an einem MTT interessant, da dort dann Aufgaben, die zusammen durchzuführen sind, auch gemeinsam modelliert werden können. CTTE wird kontinuierlich am CNR-ISTI durch die Fachgruppe um Fabio Paternò weiterentwickelt, die Version 2.6.0 dient als Grundlage der folgenden Evaluation.

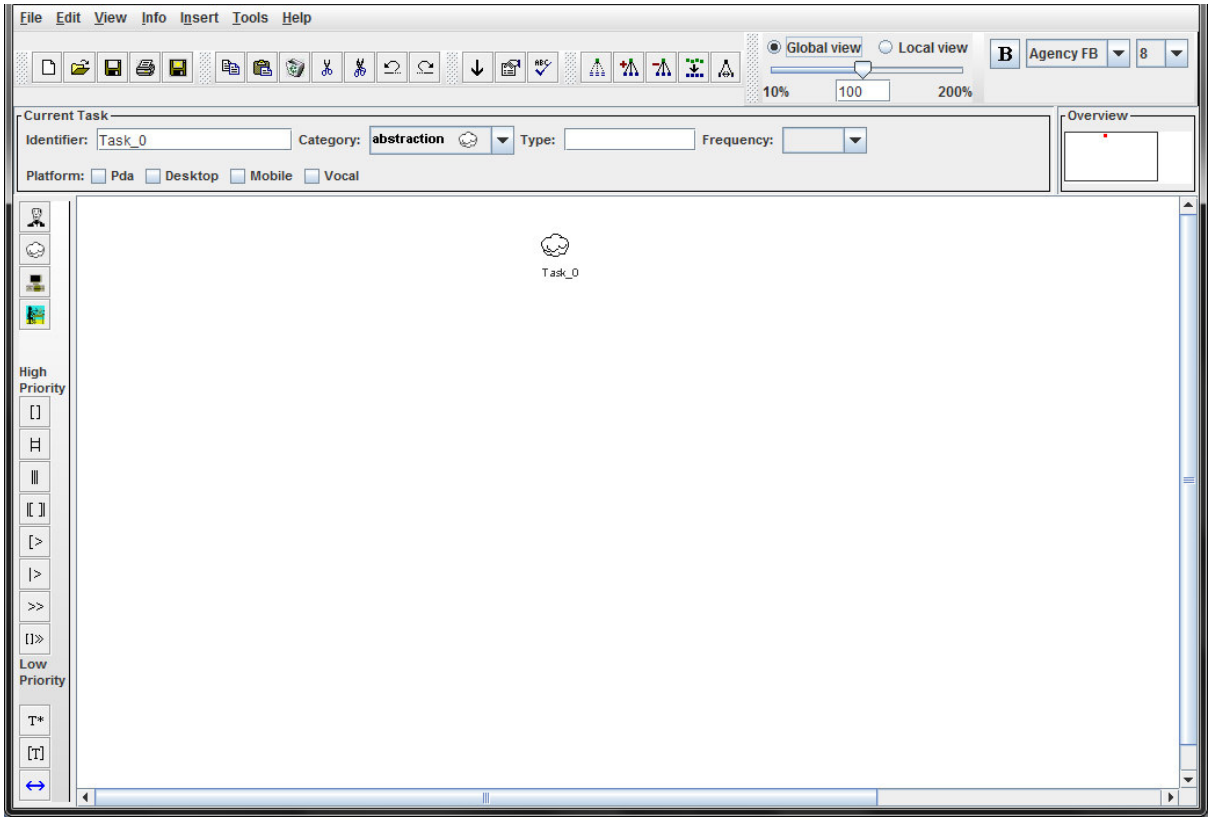


Abbildung 3.3 - CTTE Beispiel

CTTE startet in der Single-User Umgebung, in der Aufgaben eines einzelnen Benutzers mit einem System modelliert werden können – analog zu VTMB. Der kooperative Modus, der CTTE auszeichnet, ist nur versteckt über das View-Menü zu erreichen. Hinweise darauf erhält man nicht direkt, sodass auch der integrierte Rollenmanager im Verborgenen bleibt; er wird erst dann aktiviert, wenn man sich im kooperativen Modus befindet. Sehr viel besser wäre ein kurzer Dialog zu Beginn des Programms, der den Benutzer fragt, welche Art von Aufgabe er oder sie modellieren möchte. Auf diese Weise würde aktiv auf die Möglichkeit der kooperativen Tasks aufmerksam gemacht; ein Umstand, der im Konzept Beachtung finden wird.

Ein Vorteil von CTTE ist die verwendete Symbolik innerhalb des Diagramms. Zu Beginn enthält es lediglich eine Wolke (s. Abbildung 3.3). Diese stellt eine abstrakte Aufgabe dar, also die eigentliche Aufgabe auf oberster Ebene, die man mit dem Modell näher beschreiben will. Dieser „Supraaufgabe“ können im nächsten Schritt weitere Teilaufgaben hierarchisch untergeordnet werden. Hier hat man die

Wahl zwischen Nutzeraufgaben, Aufgaben des Systems, Interaktionsaufgaben (Nutzer mit dem System oder mit anderen Nutzern) oder weiterer abstrakter Aufgaben. In der Standardansicht werden farbliche Icons verwendet, die eine einfache Assoziation ermöglichen (vgl. Abbildung 3.4): Eine Person für die User-Aufgabe, ein Computer für Aufgaben des Systems oder eine Zeichnung eines Menschen an einem PC für Interaktionsaufgaben. Auf diese Weise kann man direkt entscheiden, wer die Teilaufgabe real ausführt. Gleichzeitig wird ein breiteres Spektrum an Möglichkeiten vorgegeben wie im Vergleich zu VTMB. Diese Art der Darstellung ist ansprechend und wirkt intuitiv benutzbar. Als Zusatz sind die verwendeten Icons auch austauschbar.

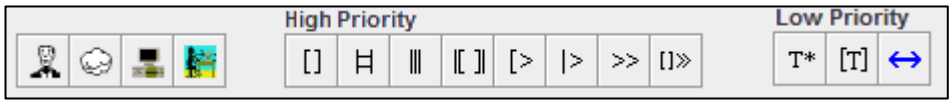


Abbildung 3.4 - CTTE Symbolik

Für die temporalen Relationen verfolgt CTTE ein leicht anderes Konzept als die anderen untersuchten Anwendungen. Beziehungen zwischen den Unteraufgaben können direkt innerhalb des Baumes erzeugt und ihre Ausprägung selektiert werden. Um das zu erreichen, werden im Baum Querverbindungen von einem Kind zum nächsten angelegt, die mit einem bestimmten Operator ausgestattet werden können, welcher die Art der Beziehung repräsentiert (vgl. Abbildung 3.4). Hier zeigt sich ein Nachteil von CTTE, da eine sehr abstrakte resp. sich sehr ähnlich sehende Darstellung der temporalen Relationen verwendet wird; diese muss vom Benutzer gelernt werden, um effektiv Arbeiten und das spätere Diagramm auch lesen zu können. Eine eindeutige oder leichter zu unterscheidende Repräsentation (vielleicht durch Textform) sucht man vergeblich – VTMB bietet in dieser Hinsicht die besser verständliche Variante an. Dass jedoch die Verwendung von Querverbindungen im Modell den Horizont der verwendbaren Relationen stark erweitert bzw. andere Relationen möglich macht, muss man diesem Ansatz dennoch zu Gute halten.

Für das Manipulieren des Aufgabenbaums stehen verschiedene Optionen zur Verfügung. Zunächst lässt sich über einen Schalter festlegen, an welcher Stelle neue Knoten angelegt werden: hierarchisch *unterhalb* des ausgewählten Knotens oder *links* davon auf gleicher Ebene. Dafür wählt man den jeweiligen Knoten aus und klickt dann eins der Aufgabensymbole an, welches dann entsprechend angeordnet wird. Nachträgliches Verändern des Baumes ist einfach möglich. Nachdem ein Knoten mit Linksklick ausgewählt wurde, kann er mit der rechten Maustaste per Drag&Drop an anderen Knoten angeordnet werden. Dazu werden je nach Berührungspunkt bei Kontakt mit dem Zielknoten Richtungspfeile angezeigt, die dem Benutzer eine Zuweisung links, rechts oder unterhalb des angepeilten Knotens suggerieren. Diese Methode ist sehr intuitiv lässt sich sehr gut auch für Touch-Anwendungen einsetzen. Teilbäume können ein- und ausgeklappt werden, jedoch nicht direkt am Knoten sondern nur über eine Schaltfläche. Zur besseren Übersicht können die Abstände zwischen den Ebenen erhöht oder verringert werden, und eine Ausrichtungsfunktion für die Knoten ist ebenfalls integriert, um das Modell bei chaotischer Anordnung wieder übersichtlich zu machen.

Über das Kontextmenü des einzelnen Knotens können dessen Eigenschaften aufgerufen werden (s. Abbildung 3.5). Im General-Reiter können allgemeine Einstellungen vorgenommen werden, wie die Art der Teilaufgabe, ihre Häufigkeit, die Plattform auf der sie ausgeführt wird, sowie auf sich selbst bezogene temporale Operatoren und Vor- und Nachbedingungen; Parallelen zu VTMB sind an dieser Stelle deutlich erkennbar. Dass im Modell die Plattform gespeichert werden kann, hilft beim späteren erneuten Öffnen oder bei Weiterbearbeitung des Modells durch Dritte bei der Einordnung in den Kontext der Aufgabe. Es besteht die Wahl zwischen *PDA*, *Desktop*, *Mobile* und *Vocal,* die durch

Checkboxen auch parallel markiert werden können. Als temporale Operatoren können *Iteration* und *Optionalität* ausgewählt werden. Einen Spezialoperator stellt die Kennzeichnung als Teil einer kooperativen Aufgabe dar, der wie die Möglichkeiten der Vor- und Nachbedingungsdefinition noch genauer beschrieben wird.

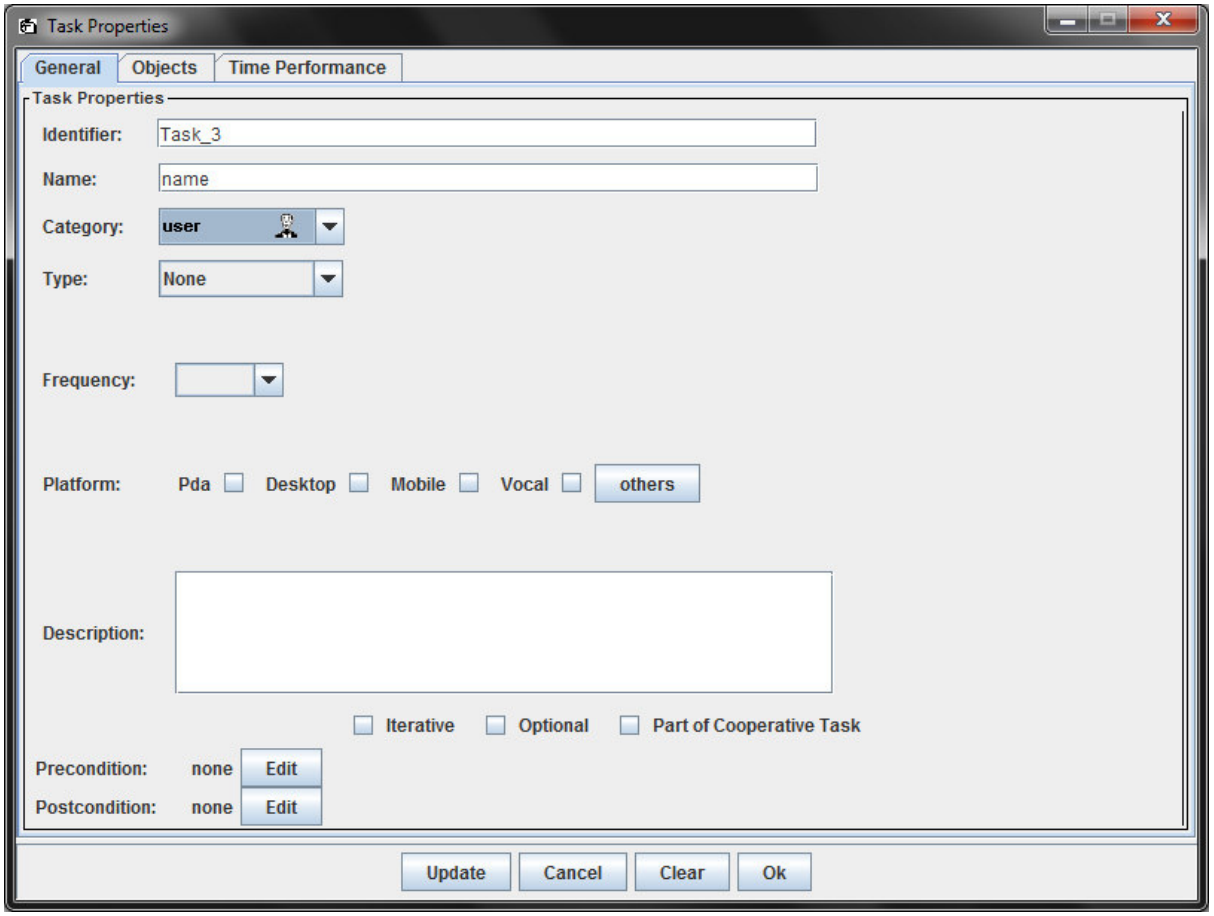


Abbildung 3.5 - CTTE Task Properties

Im Reiter *Objects* können für die Aufgabe benötigte Objekte angelegt werden. Dabei liegt nicht wie bei VTMB eine Klassenhierarchie zu Grunde, die Objekte können einfach direkt angelegt werden. Ihre Klassifizierung lässt sich in CTTE über den Datentyp festlegen. Für die verschiedenen weiteren Einstellmöglichkeiten liegt keine Beschreibung vor, sodass es für den Benutzer schwierig ist, die Optionen und ihre Auswirkungen zu erfassen. Im Konzept sollte eventuell darauf geachtet werden, Beschreibungen zu den Optionen einblendbar machen zu lassen. Zu guter Letzt lassen sich noch die Ein- und Ausgabeoptionen für verschiedene Objekte unterhalb dieser Liste definieren. Abschließend bietet der Reiter *Time Performance* die Funktion, das Zeitverhalten der Teilaufgabe festzulegen. Es kann die minimale und maximale Zeit für das Durchführen einer Teilaufgabe definiert werden, sowie der dafür ansetzbare Durchschnittswert.

Zoomen und Navigieren innerhalb des Diagramms zeigt ähnliche Probleme wie zuvor VTMB. Zwar gibt es die bereits angesprochene Miniaturkarte des Baumes (*Overview*), jedoch ist das Vergrößern und Verkleinern der Ansicht ebenfalls nur durch einen Schieberegler möglich, für den man jedes Mal das Diagramm verlassen muss. Bei großen Diagrammen verliert man beim Zoomen ebenso den Fokus, und man muss sich über die Scroll-Balken am Rand zur gewünschten Position zurückbewegen. Weiterhin werden wesentliche Informationen im Baum nicht direkt angezeigt. Zwar gibt es verschiedenartige Knoten im Baum und ihre Beziehungen zueinander sind gekennzeichnet, aber es werden keinerlei Informationen über Bedingungen, erzeugte Objekte oder Zeitverhalten angezeigt. Diese Informationen

sind nur über das Eigenschaftenfenster einsehbar; eine entsprechende Kennzeichnung sollte im Konzept bedacht werden.

Ein stark ausgeprägtes Element von CTTE ist der integrierte Regeleditor für Bedingungen (s. Abbildung 3.6). Während ältere Versionen der Software keine oder lediglich Vorbedingungen einbeziehen konnten, ist es nun möglich, Pre- und Post-Conditions eines Schrittes im Aufgabenmodell zu simulieren. Dazu können zunächst Regelgruppen definiert werden, die die Bedingungen durch AND, OR oder XOR aufnehmen und miteinander verknüpfen. Hier ist eine Erweiterung in Bezug auf VTMB zu erkennen, da VTMB zwar auch das Abfragen mehrerer Bedingungen ermöglicht, jedoch dort ausschließlich die UND-Verknüpfung realisiert; d. h. alle Bedingungen müssen gleichzeitig zutreffen, damit nicht blockiert wird. CTTE befähigt den Benutzer, die Regelgruppen in einem hierarchischen Baum anzuordnen, sodass auch intermodale Kombinationen denkbar sind (z. B. die AND Verknüpfung einer OR und einer XOR Gruppe von Regeln).

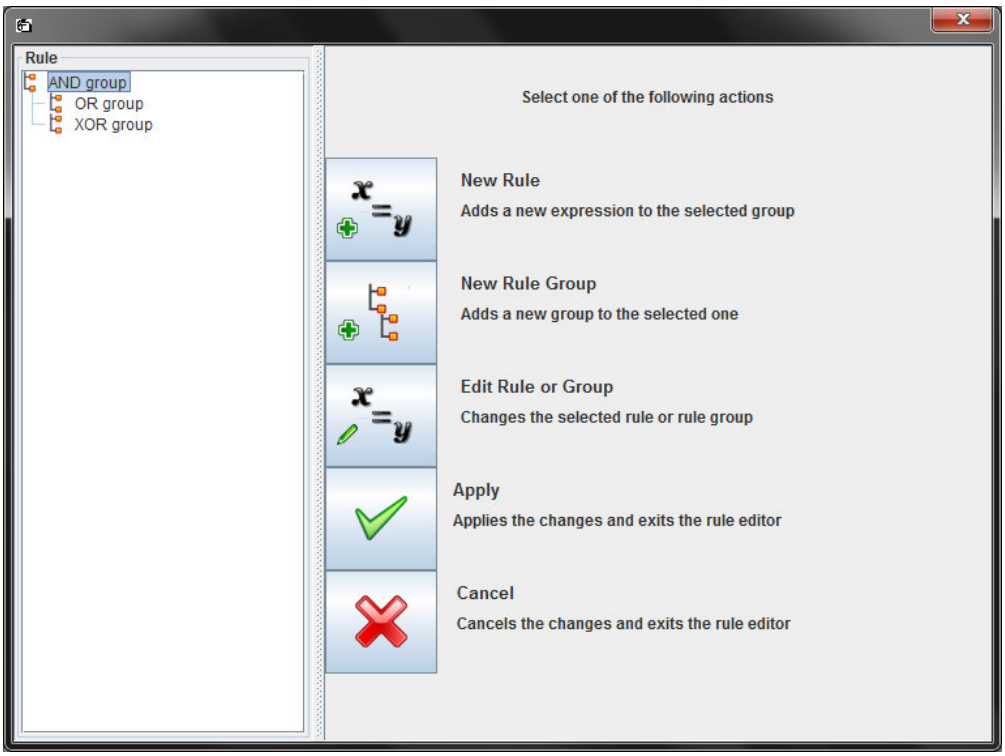


Abbildung 3.6 - CTTE Bedingungseditor

Zur Definition einer Regel müssen zuvor *Task Objects*, also zur Teilaufgabe gehörige Objekte, vom Benutzer angelegt werden (wie ein paar Absätze zuvor beschrieben). Eins dieser Objekte kann nun für die Regel ausgewählt und mit Hilfe der Vergleichsoperatoren

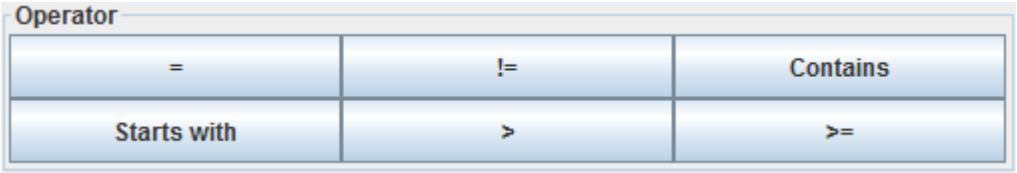


Abbildung 3.7 - CTTE Vergleichsoperatoren für Bedingungen

entweder mit einem weiteren Task Objekt oder einem Literal verglichen werden. Diese Operatoren verändern sich dynamisch (je nach verwendeter Klasse der erzeugten Aufgabenobjekte), sodass zu Laufzeitfehlern führende Vergleichsoperationen präventiv vermieden werden. Dieser Regeleditor ist der bisher umfangreichste und ein weiterer Pluspunkt für die Verwendung von CTTE.

Der vielleicht wichtigste Unterschied zu anderen Werkzeugen ist der – wenn auch versteckte – Kooperationsmodus. Wird dieser ausgewählt, wird das Hauptfenster um neue Elemente ergänzt. Es wird zunächst ein Reiter *Cooperative* angelegt, auf dem ein Metaaufgabenbaum erstellt wird. Dieser verknüpft die Aufgaben von verschiedenen Akteuren miteinander. Startelement dieses Baumes ist eine sogenannte *Cooperative Task*, also eine Aufgabe die gemeinsam durchgeführt wird. Dieses Symbol erscheint neu unter den auswählbaren Schaltflächen; dafür wird die Möglichkeit, eine Connection Task zu setzen für diesen Baum deaktiviert, da diese dafür gedacht ist, die Aufgabenbäumen der einzelnen Akteure mit dem kooperativen Baum zu verbinden.

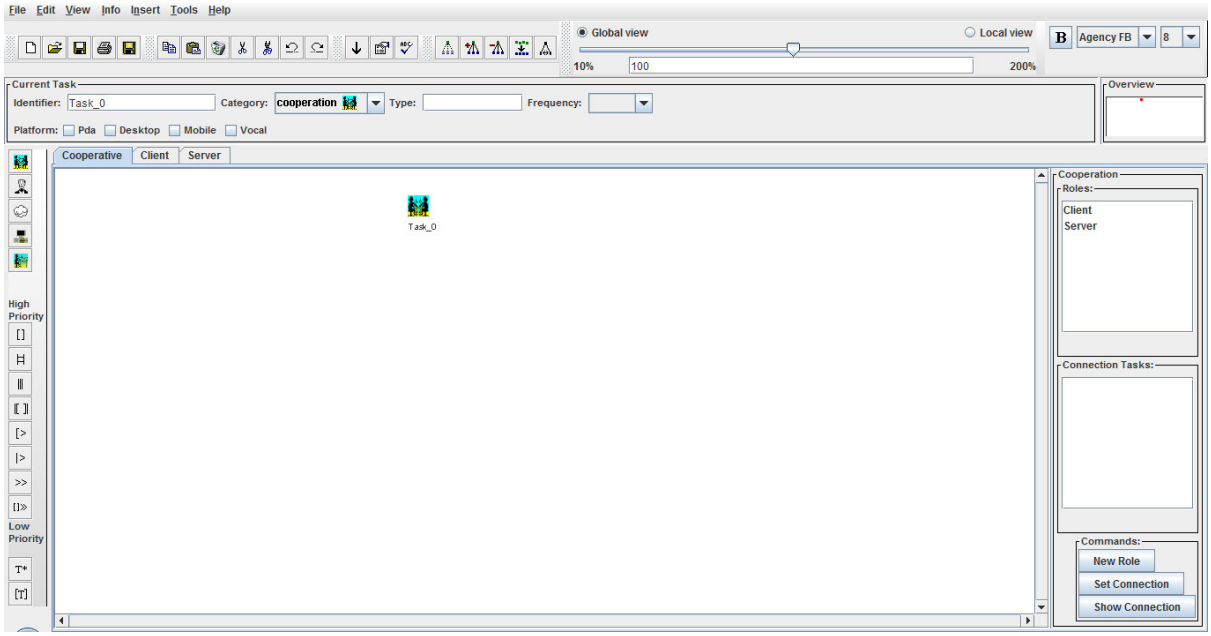


Abbildung 3.8 - CTTE kooperativer Modus

Zusätzlich wird auf der rechten Seite des Bildschirms der Rollenmanager eingeblendet. In diesem können nun die verschiedenen Rollen der Akteure definiert werden, und für jede angelegte Rolle wird ein weiterer Reiter erzeugt, innerhalb welchem die jeweiligen Aufgabenbäume der Rollen modelliert werden können. Auf diesen Reitern steht dann auch wieder die Option der Connection Task zur Verfügung, um Verbindungen mit dem Metabaum herzustellen. Diese Eigenschaft und dieses Verhalten sind höchst interessant für den MTT, da hier separat simultan an den verschiedenen Bäumen gearbeitet werden kann, ohne Reiter wechseln zu müssen. Diese Methodik wird eine zentrale Rolle innerhalb des Konzepts spielen.

Schlussendlich sei noch erwähnt, dass zur Überprüfung des Modells ein *Model-Checker* in die Software implementiert ist, sowie ein *Simulator*, mit dessen Hilfe die mit dem Modell beschriebene Aufgabe direkt entlang des Baumes ausgeführt werden kann. Der Model-Checker deckt Fehler in der Struktur des Modells auf und leitet den Benutzer unmittelbar zu problematischen Stellen darin, um diese manuell zu beheben. Erst wenn alle Fehler behoben sind, kann das Modell simuliert werden. Der Simulator öffnet in einem neuen Fenster, und lässt sukzessive alle Schritte des Modells nacheinander durchführen. Dabei kann das Diagramm beobachtet werden. Die Steuerung über die nächsten auszuführenden Schritte bzw. Aufgabenobjekte, Bedingungen etc. erfolgt in einem eigenen Bereich. Es ist vorgesehen, dass verschiedene Szenarios zu einem Modell erstellt und gespeichert werden können. Ein weiterer interessanter Aspekt für kollaborierendes Arbeiten am MTT.

Am Ende die wesentlichen Eigenschaften von CTTE im Kurzüberblick:

Pro	Contra
+ Kooperativer Modus mit interagierendem Metabaum + Symbolik zur Steuerung vorhanden, meist intuitiv + Erweiterte temporale Relationen + Eigener Formeleditor für Vor-/Nach-bedingungen + Übersichtsfeld für den Baum + Teilbäume können im Modell aus bereits anderen erzeugten Modellen importiert werden + Simulation des Modells direkt im Baum mit Szenarienbildung	– Kooperativer Modus nicht sofort sichtbar – Zoomen nur eingeschränkt möglich – Eigene Symbolik für temporale Relationen, die gelernt werden muss

Tabelle 3.2 - Bewertung: CTTE für interaktives Display

3.1.3 K-MADe

Das Kernel of Model for Activity Description environment wird am nationalen französischen Forschungsinstitut INRIA (*Institut national de recherche en informatique et en automatique)* entwickelt. Dieses Werkzeug versucht dazu beizutragen, durch Aufgaben- und Aktivitätenanalyse die Ergonomie in den Designprozess interaktiver Systeme einzugliedern. Um dies zu erreichen, wird als Basis ein Modell zu Grunde gelegt, dessen Ausdruckskraft auf einer formalen Semantik beruht. Diese Eigenschaft soll die Aufgabenbeschreibung und -analyse erleichtern, aber auch die Ausführung des Modells sowie die Migration zwischen Modellen oder Abschnitten im Softwarelebenszyklus (K-MADe 2012). Die Evaluation erfolgt mit der momentan aktuellsten Version 1.0.0 vom November 2010.

Gleich beim ersten Programmstart steht man vor einer Hürde, da die Standardsprache auf Französisch eingestellt ist und diese nur versteckt über das Einstellungsmenü auf Englisch geändert werden kann. Danach muss die Software neu gestartet werden, um die Änderungen wirksam zu machen. Wer der französischen Sprache nicht mächtig ist, kann im schlimmsten Fall nicht mit der Software arbeiten. Weiterhin sind nicht alle Teile der Software übersetzt, sodass zwischen den englischen Begriffen teilweise französische auftauchen. Im Konzept sollten solche Fehler berücksichtigt und unbedingt vermieden werden.

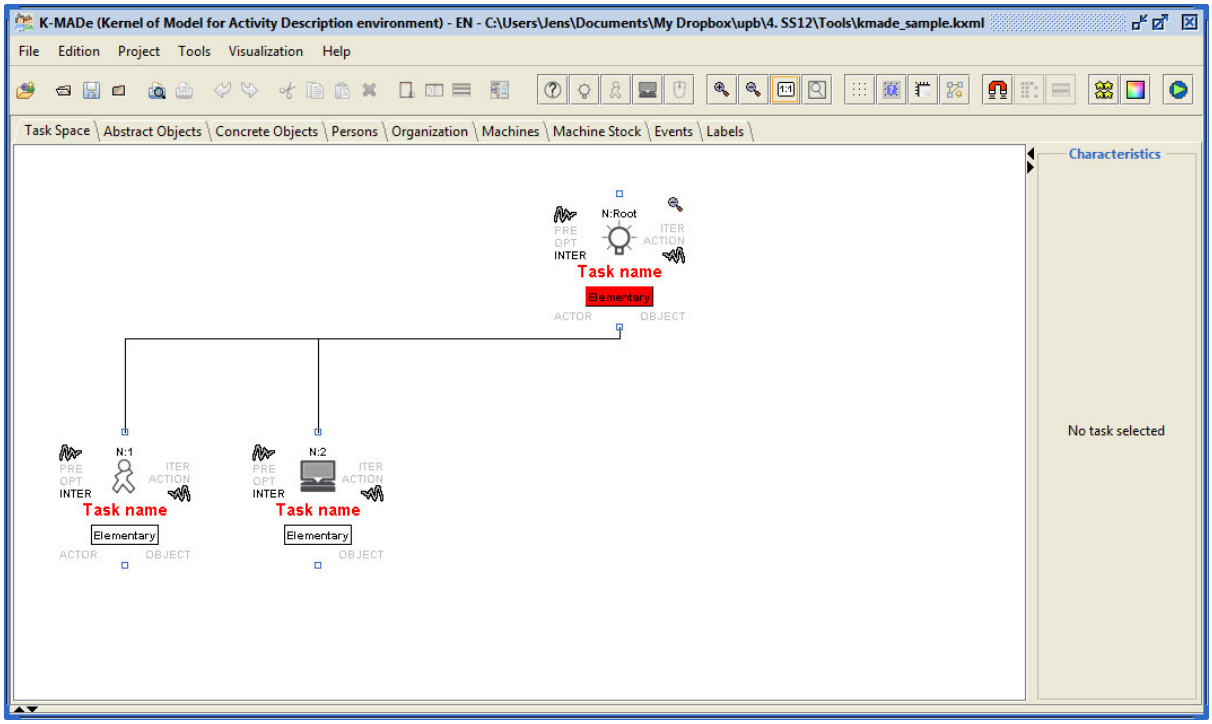


Abbildung 3.9 - K-MADe Beispiel

Schon in den Release Notes wird angekündigt, dass sich die Software noch im Entwicklungsstadium befindet und deshalb elementare Funktionen wie Undo/Redo oder verschiedene Task Characteristics, also Aufgabencharakteristiken, nicht funktionieren. Nichts desto trotz bietet K-MADe dennoch würdige Elemente und Ideen, die eine nähere Betrachtung rechtfertigen.

Legt man ein neues Projekt an, wird man in einem ersten Schritt wie bei einem Interview (*Interviews Management*) nach Auftraggeber und Zweck des zu erstellenden Falles befragt. Man kann dort Informationen zum Umfeld hinterlegen und weiterhin angeben, welche Personen zu der Thematik befragt wurden, um das Modell zielgruppengerecht aufbauen zu können. Dies hilft bei der späteren Einordnung in den Kontext, kann aber auch bei Bedarf einfach übersprungen werden.

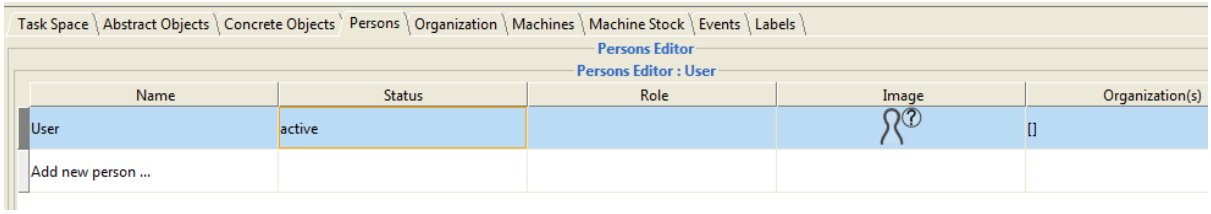


Abbildung 3.10 - K-MADe Elemente

Gleich zu Beginn fällt auf, dass die Entwickler an vielerlei nützliche Dinge bei der Entwicklung der Software gedacht haben. Es werden zu jedem neuen Projekt unterschiedliche Reiter mit angelegt, innerhalb derer Dinge mit direktem Bezug zum Modell definiert werden können bzw. müssen (s. Abbildung 3.10). Zur Definition vorgesehen sind abstrakte bzw. konkrete Objekte (wobei letzteres noch nicht funktioniert); Personen und Organisationen, in die die Personen eingeordnet werden können; Maschinen und Maschinenbestände, die die Maschinen zu Gruppen zusammenfassen; sowie Ereignisse (Events) und zu verwendende, farblich markierbare Label. Auf alle hier definierten Elemente kann später an dafür vorgesehenen Stellen im Modell zurückgegriffen werden, bzw. manche Funktionen im Modell werden erst durch die Definitionen auf diesen Reitern anwendbar. Analog zu CTTE bietet diese Methodik den gleichen Vorteil, die unterschiedlichen Reiter auf einem MTT parallel darzustellen und durch mehrere Teilnehmer individuell bearbeiten zu lassen.

Ähnlich wie CTTE verwendet K-MADe eine ausgeprägte Symbolik für die einzelnen Knoten, was eine schnelle Identifizierung der Knotenart möglich macht. Eine Glühbirne symbolisiert (wie die Wolke bei CTTE) eine abstrakte Aufgabe, die näher zu spezifizieren ist. Unterhalb können dann eine Figur, ein Bildschirm, eine Computermaus sowie weitere Glühbirnen angeordnet werden, um Benutzer-, System-, Interaktionsaufgaben oder weitere abstrakte Teilaufgaben zu definieren. Neu ist hierbei die Einführung einer sogenannten „Unknown Task“, bei der der Typ noch nicht fest steht. Dieser Knotentyp kann innerhalb der Modellierung beliebig als Platzhalter eingesetzt und ihm erst im späteren Verlauf eine Typisierung zugeordnet werden.

Die Darstellung eines Knotens im Diagramm ist sehr detaillierter Natur. Von allen bis jetzt untersuchten Werkzeugen präsentiert K-MADe die meisten Informationen direkt sichtbar am Knoten (s. Abbildung 3.11). In den Subknoten wird hinter der Anzahl N stets angegeben, um den wievielten Teilschritt es sich handelt, respektive dessen Zugehörigkeit. Beginnend von der Wurzel (*N:Root)* lässt sich so in gewisser Weise die Komplexität des Modells am Knoten nachvollziehen (z. B. N:1.2 oder N:5.6.3)

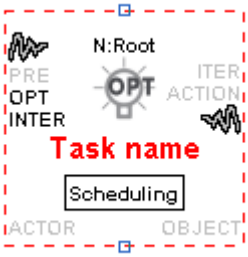


Abbildung 3.11 - K-MADe Knotendarstellung

Es ist unmittelbar erkennbar, ob ein Knoten Vorbedingungen (**PRE**conditions) besitzt, **ITER**ativ oder **OPT**ional ist, ob durch die Aufgabe weitere Aktionen hervorgerufen werden (**ACTION**s) oder der Vorgang unterbrochen werden kann (**INTER**ruptible). Des Weiteren können der Teilaufgabe zuvor festgelegte Akteure (**ACTOR**) oder Objekte (**OBJECT**) zugeordnet werden. Diese Zuordnung kann nur dann erfolgen, wenn in den vorgesehenen Reitern *Persons* und *Abstract/Concrete Objects* Elemente vorhanden sind. Für die Vorbedingungen, Iterationen oder Aktionen wird ein eigener Editor geöffnet, der sehr detaillierte Optionen liefert. Da diese jedoch nicht genau beschrieben werden, fällt es zunächst relativ schwer, dort Einstellungen vorzunehmen (eine Hilfe ist nicht implementiert). Vielleicht sollten entweder Hilfen direkt im Editor mit angegeben werden, oder die Vielfalt der verwendbaren Optionen eingeschränkt werden, was natürlich einen Funktionsverlust zur Folge hätte. CTTE bietet hier den intuitiveren Ansatz, und ist somit für das Konzept eher geeignet.

Für die Arbeit an einem Knoten liegt eine besondere Art des Kontextmenüs vor. Durch Doppel- oder Rechtsklick auf einen Knoten können verschiedene Menüs aufgerufen werden, die kreisförmig um den Knoten herum angeordnet sind (sog. Pie-Menüs oder Tortenmenüs, s. Abbildung 3.12). Per Doppelklick lässt sich der Knotentyp sehr leicht und direkt ändern; ein Rechtsklick öffnet das bekannte Bearbeitungsmenü mit Copy&Paste etc. Funktionen. Etwas unglücklich ist bei der zweiten Variante das Überlappen der Texte von Knoten und Menü; eine reine Symbolik wie beim Typenmenü plus nähere Erläuterungen in einem äußeren Ring beim Überfahren mit der Maus würde dieser Problematik entgegenwirken.

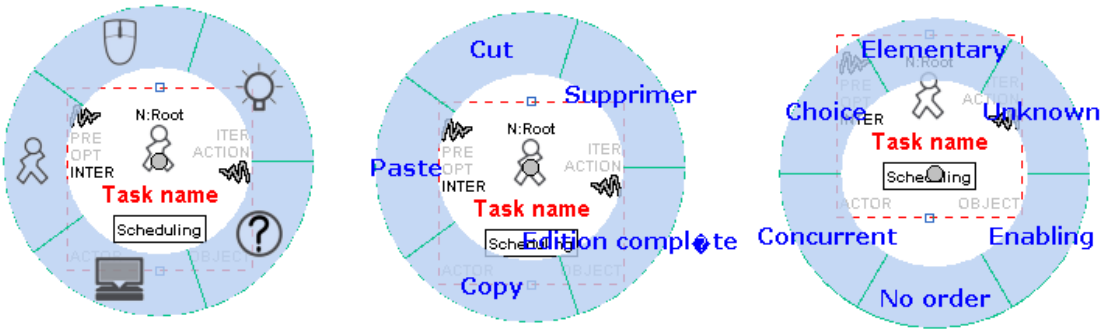


Abbildung 3.12 - K-MADe Knoten Pie-Menü

Ein zentraler grauer Punkt zeigt dabei stets an, welches Element gerade verändert wird. Dieses Menü eignet sich hervorragend für die Verwendung auf einem Touchscreen, da die Wege zu allen Optionen die gleichen sind und das Menü schnell und unkompliziert verwendet werden kann.

Zusätzlich kann man über einen Doppelklick auf den eingerahmten Text, der standardmäßig mit *Scheduling* belegt ist, die Art der temporalen Relation entsprechend abändern. Bemerkenswert ist, dass bei Fehlern in der Zusammenstellung der Knoten und Relationen dieser Kasten rot gefärbt ist und so den Benutzer auf gemachte Fehler hinweist; solange einer der Kästen im Modell rot ist, kann die Simulation nicht ausgeführt werden. Eine sinnvolle Anwendung der Signalfarbe, die sich auch im Konzept realisieren lässt.

Zuletzt lässt sich noch sagen, dass auch K-MADe einen Model-Checker implementiert hat, der spätestens dann zu Wort kommt, wenn man versucht die Simulation zu starten. Sollten noch einzelne Kästen des Modells rot sein, wird nun darauf hingewiesen (bei komplexen Modellen sind evtl. nicht alle Kästen direkt für den Benutzer einsehbar). Leider fehlt eine Funktion wie bspw. bei CTTE, die den Benutzer direkt zum Fehler hinführt. Es wird lediglich ausgegeben, um welche Art von Fehler es sich handelt, nicht wie er behoben werden kann.

Abschließend die Bewertung von K-MADe in komprimierter Form.

Pro	Contra
+ Detaillierteste Darstellung am Knoten + Gutes Kontextmenü einzelner Knoten (Pie Menu) + Erweitertes Rollenmodell + Ausgeprägter „Aufgabeninspektor“ + Einführung von Events (externe Effekte)	– Standardsprache Französisch, umständlich änderbar – Fenster vorgeschriebener Größe können bei zu kleinem Bildschirm nicht benutzt werden – Software noch nicht vollständig implementiert, wichtige Funktionen fehlen

Tabelle 3.3 - Bewertung: K-MADe für interaktives Display

3.1.4 IdealXML

Der Name *Interface Development Environment for AppLications specified in usiXML* lässt zunächst nicht direkt auf die Entwicklung von Aufgabenmodellen schließen. Die Auszeichnungssprache usiXML ist eine XML-konformer Ansatz zur Beschreibung von Benutzungsschnittstellen, der dazu entwickelt wurde, User Interfaces durch Nicht-Programmierer wie Analysten, Designer, Projektleiter oder Humanfaktor-Experten beschreiben zu lassen. IdealXML ist ein Werkzeug aus einer Reihe von Editoren, mit deren Hilfe man usiXML spezifische Modelle auf Basis von UI Pattern grafisch erstellen kann (Montero & López-Jaquero 2007) (Vanderdonckt & Simarro 2010).

Die Sprache sowie die Editoren wurden an der katholischen Universität Louvain in Belgien entwickelt und werden durch das CAMELEON Forschungsprojekt der Europäischen Kommission sowie das SIMILAR network of excellence unterstützt. Zur Evaluation wurde die Version *evolution2* verwendet, ein genaues Releasedatum ließ sich nicht ermitteln.

Beim ersten Programmstart von IdealXML bekommt der Benutzer zunächst ein leeres Fenster präsentiert. Über das Menü *Model* lassen sich die Module *Domain model*, *Task model*, *AUI model* sowie *Mapping model* laden. Jedes einzelne dieser Module öffnet sich in einem weiteren Fenster innerhalb des Hauptfensters. Diese lassen sich beliebig anordnen, oder können durch das *Window* Menü automatisch ausgerichtet werden. Im Domain Model lässt sich die Domäne des Modells spezifizieren. Zu diesem Zweck stehen UML-typische Notationsmöglichkeiten zur Verfügung, mit denen Klassen angelegt und miteinander durch Abhängigkeiten, Generalisierung, Assoziationen sowie Aggregation oder Komposition miteinander verknüpft werden können.

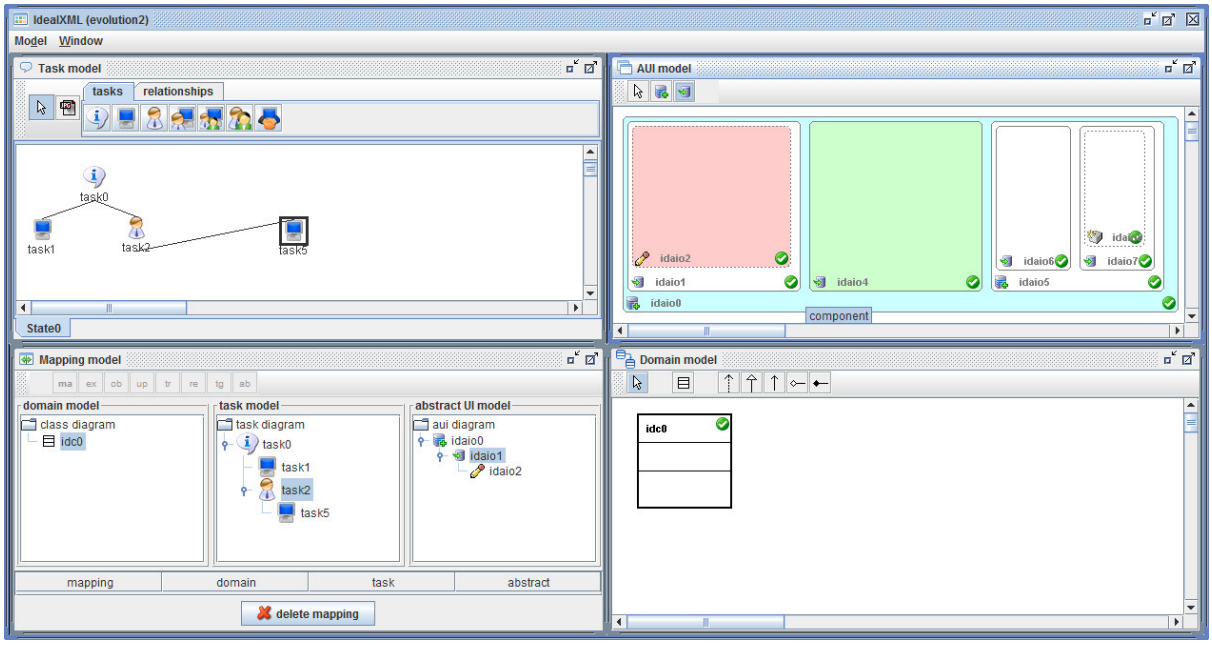


Abbildung 3.13 - IdealXML Beispiel

Das Modul *Task model* ist für die Modellierung der Aufgabe zuständig. Hierfür stehen die bekannten Arten *abstract task, application task* (system), *user task* und *interaction task* zur Verfügung (s. Abbildung 3.14). Neu sind hingegen die Elemente *cooperation with interaction task, working team* und *collaboration with interaction task.* So interessant diese Elemente auch klingen, eine Beschreibung über ihre Verwendung sucht man vergebens. Eine Hilfe ist zwar vorgesehen, lässt sich aber nicht öffnen.

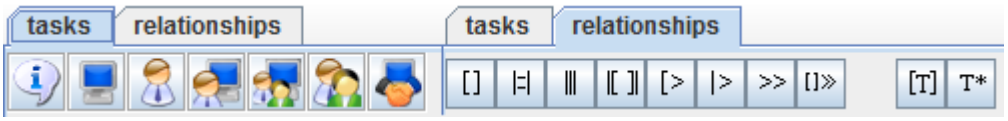


Abbildung 3.14 - IdealXML Aufgabensymbolik

Bei den temporalen Beziehungen (Reiter *relationships*) begegnet man „alten Bekannten“. Hier wurde auf die Symbolik und Funktionsweise von Paternò zurückgegriffen, wie er sie bei CTTE verwendet (nachträglich lassen sich selbst die Task Icons auf die von CTTE verwendeten einstellen). Auch die Art des Einfügens der Relationen erfolgt analog zu CTTE. Man wählt ein Element aus, klickt die gewünschte Relation an und die Verbindung zum nächsten Knoten gleicher Ebene erhält die gewünschte Eigenschaft. Dies stellt keinen Nachteil bezüglich der Funktion dar, bietet aber auch keine Neuerungen. Die Eigenschaften einzelner Aufgaben lassen sich per Doppelklick auf das entsprechende Icon aufrufen. Es erscheint ein Fenster, in dem relativ strukturlos Einstellungen vorgenommen werden können (s. Abbildung 3.15).

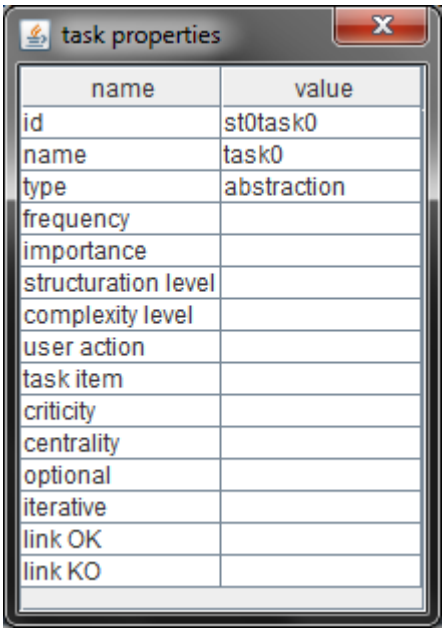


Abbildung 3.15 - IdealXML Task Properties

Eine Beschreibung der einzelnen Felder liegt nicht vor; unbekannte Felder respektive solche dessen Effekte sich nicht aus dem Namen erschließen lassen müssen entsprechend leer gelassen oder zufällig belegt werden. Für manche Felder sind Wertebereiche oder Belegungen vorgesehen; diesen Zustand bzw. seine möglichen Ausprägungen erfährt man jedoch erst wenn man darauf klickt. Auffällig sind an dieser Stelle noch die beiden Einträge *link OK* und *link KO*, mit denen anscheinend Verbindungen zu bestimmten Zuständen des Modells hergestellt werden können, da ein fester Wert *State0* ausgewählt werden kann. Aufgrund von fehlender Information zu dieser Funktion kann dies aber nicht hinreichend überprüft werden.

Manipulieren des Aufgabenbaums erweist sich als sehr umständlich, da hier eine kaum instinktiv nutzbare Technik verwendet wird. Will man Elemente verschieben, wählt man zunächst den *Selector* aus, ein Pfeilsymbol das links neben den Aufgabenicons platziert ist. Nun klickt man das zu verschiebende Objekt mit linker Taste an, und es wird an den Cursor angeheftet. Man bewegt es nun zur gewünschten Zielposition und führt einen Rechtsklick aus, um es abzulegen. Diese Art von Drag&Drop ist sehr gewöhnungsbedürftig und zu alledem weder bekannt noch ersichtlich. Weiterhin lassen sich neue Knoten im Baum nur umständlich einfügen. Zunächst wählt man die Knotenart aus und selektiert

den Knoten, unterhalb dessen man ein weiteres Element hinzufügen will. Im Anschluss muss dann irgendwo auf die weiße Arbeitsfläche geklickt werden, wo dann der Knoten mit einer Verbindung zum Superknoten angelegt wird – selbst wenn sich die Position geographisch oberhalb befindet. Will man diesen dann verschieben und trifft mit der Maus das neue Icon nicht richtig, wird direkt ein weiteres Element erstellt. Die Behebung solcher Fehler gestaltet sich als sehr zeitraubend und wirkt schlecht gelöst.

Ein wirklich neuer Aspekt ist das *Abstract UI model* (kurz *AUI model,* s. Abbildung 3.16). Mit seiner Hilfe kann bereits ein abstrakter Prototyp einer Benutzerschnittstelle für das erstellte Aufgabenmodell entwickelt werden. Dazu kann man innerhalb des Fensters Container anlegen, die verschiedene Komponenten aufnehmen können. Diesen Komponenten können wiederum Facetten (*facet*) zugeordnet werden, um ihre Art zu spezifizieren. Zur Verfügung stehen *input*, *output*, *navigation* und *control*. Auch hier fehlt wieder die Erklärung der Elemente und der jeweiligen Effekte, was die Benutzung als schwer nachvollziehbar erscheinen lässt. Die starke Abstraktheit mindert diese Erkenntnis noch weiter, zu sehen auf Abbildung 3.16.

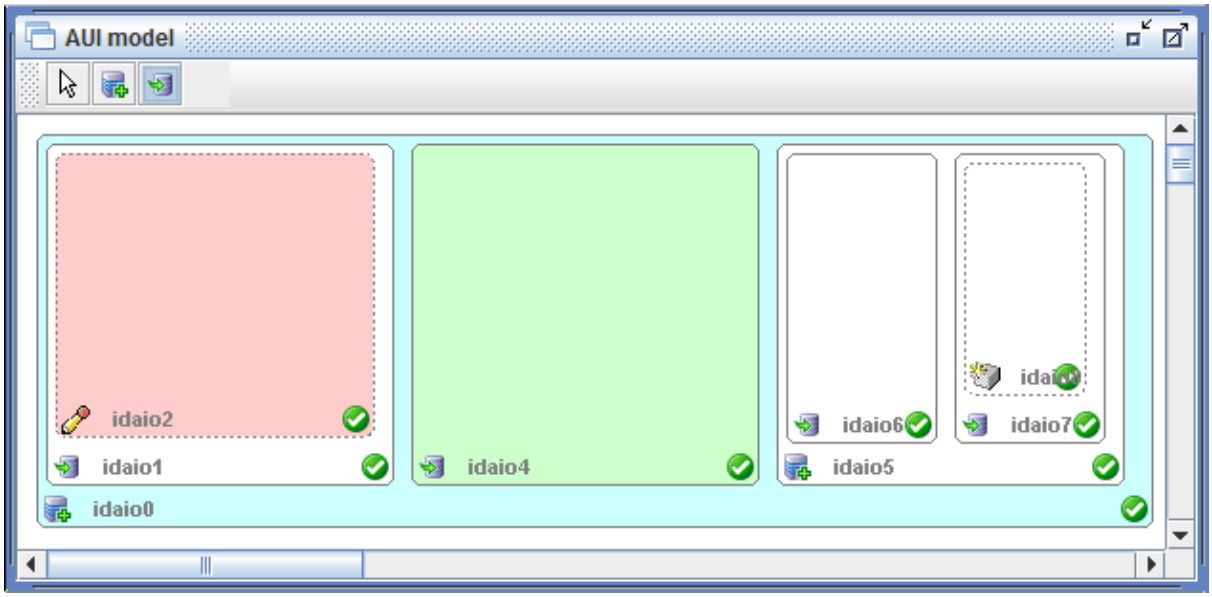


Abbildung 3.16 - IdealXML Abstract UI model

Es können beliebig viele Container erstellt werden, die ihrerseits wieder Container aufnehmen können, oder eben Components. Bei großen Modellen verliert man sehr schnell den Überblick, zumal auch das Zoomen in keinem der Modulfenster möglich ist. Zwar ist man in der Lage, alle Elemente farblich voneinander abzusetzen; jedoch ist die Anzahl der Farbnuancen ebenso wenig unendlich wie der ausnutzbare Raum für die Erstellung des abstrakten UIs. Positiv ist dennoch zu erwähnen, dass man das AUI Modell auch automatisch aus dem produzierten Aufgabenmodell erzeugen lassen kann.

Das *Mapping Model* (s. Abbildung 3.17) schließlich erweckt den Anschein, alle bisher vorgestellten Modelle miteinander zu verbinden. Nach einem Auffrischen des Mapping Modells über das Kontextmenü erscheinen alle angelegten Elemente der anderen Modelle in einem hierarchischen Baum, und können mit diversen Methoden in Beziehung gesetzt werden. Ausgewählt werden können **ma**nipulates, is**Ex**ecutedin, **ob**serves, **up**dates, is**T**ranslatedInto, is**Re**ifiedby, **t**ri**g**gers und is**Ab**stractedInto; jedoch wiederholt ohne Verweis auf die Funktionsweise etc.

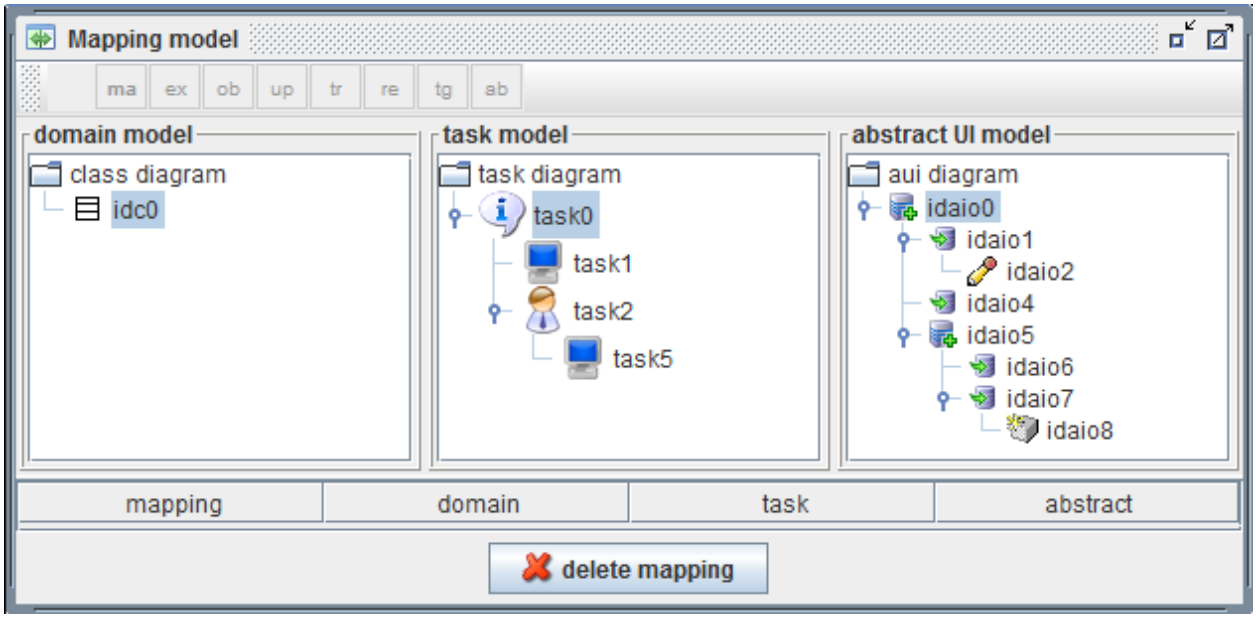


Abbildung 3.17 - IdealXML Mapping model

Dieses Fenster liefert im Wesentlichen einen kompakten Überblick über die in den Modulen geschaffenen Objekte und deren Struktur; wie man damit arbeitet wird aber keinesfalls deutlich.

Zuletzt bleibt noch zu erwähnen, dass Standardfunktionen heutiger Software zur Modellierung und Diagrammbearbeitung einfach fehlen. Eine Zoomfunktion sucht man innerhalb der Fenster vergebens, es kann lediglich immer eines der Fenster (durch das Hauptfenster beschränkt) maximiert werden. Das Verschieben der Arbeitsfläche ist nur mit den Scroll-Balken durchführbar. Will man das Modell sichern, stellt man fest, dass jedes einzelne der Module getrennt abgelegt werden muss – ein zusammenhängendes Speichern ist nicht vorgesehen. Das Mapping Modell kann inkonsequenter Weise nicht gespeichert werden, sodass dort gemachte Verknüpfungen jedes Mal neu angelegt werden müssen, nachdem die anderen drei Module wieder einzeln geladen wurden.

Obwohl der Editor sehr schöne Ansätze liefert, die auf dem MTT gut durch mehrere Akteure ausführbar wären, mangelt es jedoch an Verständnis für die Funktionsweise der einzelnen Module. Für eine direkte Adaption reicht dies nicht aus, da gravierende Mängel bzgl. der Bedienung der einzelnen Elemente/Module bestehen. Weiterhin stören immanente Fehlfunktionen die Benutzbarkeit, ein Vermeiden dieser Fehler wäre für eine Adaption unvermeidlich. Vor allen Dingen das getrennte Speichern trägt wenig dazu bei, die Software produktiv einzusetzen. Die vorgestellten Erweiterungen werden dennoch Beachtung bei der Entwicklung des Konzeptes finden.

Zum Abschluss der Kurzüberblick über die Evaluation von IdealXML:

Pro	Contra
+ Aufteilung der verschiedenen Modelltypen auf verschiedene Module	– Unintuitive Art der Diagrammmanipulation
+ Domänen und Abstraktes UI Modell	– Kein Zoomen, Reduzieren o.Ä. möglich
+ Mapping Modell zur Herstellung von Verknüpfungen zwischen den Modelltypen	– Hilfe vorgesehen, aber nicht implementiert
+ AUI Modell lässt sich automatisiert erzeugen	– Wenigste Usability aller getesteten Werkzeuge
	– Nur getrenntes Speichern der Modelle vorgesehen

Tabelle 3.4 - Bewertung: IdealXML für interaktives Display

3.2 Evaluation von Techniken zur Interaktion auf Displays

Während im vorherigen Abschnitt die erste für die Konzeptionierung wichtige Thematik erörtert wurde, soll dieser Abschnitt der Untersuchung und Bewertung existierender Techniken zur Interaktion auf einem Display dienen. Dazu werden die unterschiedlichsten Ansätze analysiert und im Hinblick auf Eignung zur Erstellung von Diagrammen und Modellen sowie zum gemeinsamen Arbeiten evaluiert.

Dabei soll es grundsätzlich um die Frage gehen, wie die Eingabe durch den Anwender durchgeführt wird. Kommen zum Beispiel nur die Finger zum Einsatz oder wird auf weitere Hilfsmittel wie einen Stift o. Ä. Zeichengerät zurückgegriffen? Oder können im Sinne der Tangibles auch andere Objekte sinnvoll zur Interaktion eingesetzt werden? Macht es Sinn, mobile Endgeräte wie Smartphones in den Prozess mit einzubeziehen? Und auf welche Weise unterstützten diese Techniken das gemeinsame Arbeiten verschiedener Personen an ein und demselben Aufgabenmodell?

Um all diese Fragen zu klären, gilt es in diesem Kapitel einen Überblick über aktuelle Ergebnisse der Forschung zu schaffen, die die Problematik dieser Arbeit mit State-of-the-Art Methoden bereichern. Beispielsweise der in 2011 etablierte Lehrstuhl für User Interface & Software Engineering der Universität Magdeburg beschäftigt sich intensiv mit der Gestaltung von Benutzungsschnittstellen. Professor Dr.-Ing. Raimund Dachselt ist Initiator vieler verschiedener Projekte im Bereich User Interfaces und wird durch das Bundesland Sachsen-Anhalt bei dem Projekttitel *„Diagramm-Interaktion“* unterstützt (Forschungsportal Sachsen-Anhalt 2012).

3.2.1 Node-Link Diagramme auf interaktiven Displays

Aufgabenmodelle werden typischerweise als sogenannte *Node-Link Diagramme* dargestellt, also Diagramme, bei denen im Graphen Knoten (Node) über Kanten miteinander verknüpft (Link) sind. Das besondere bei Aufgabenmodellen ist die Repräsentation der hierarchischen Struktur des Modells als Baum. Klassischerweise wurden Node-Link Diagramme in der Vergangenheit zunächst als direkte Zeichnung (sog. *Sketch*) mit der Hand und einem Stift auf Papier angefertigt, oder später dann mit Hilfe der Maus und Tastatur mit entsprechender Software auf einem Desktop-PC.

Große Touchscreens liefern in der Gegenwart idealere Voraussetzungen für die Erstellung solcher Diagramme, da sie nicht nur eine größere Arbeitsfläche bieten als herkömmliche Bildschirme, sondern eben auch instinktiv für das Zeichnen eingesetzt werden können, da sie metaphorisch „wie ein Blatt Papier“ funktionieren können. Jedoch kann es auch durchaus passieren, dass die Modelle unter Umständen einfach zu groß werden, um sie real abzubilden. Aus diesem Grund ist es erforderlich, das heutige Systeme den Komfort bieten, intuitiv etwas zu skizzieren, und es gleichzeitig schaffen, die Beschränkungen früher verwendeter Techniken zu reduzieren.

3.2.1.1 Sketch-basierte Interaktionstechnik mit digitalen Stiften und MT-Gesten

Heutige Forschung auf dem Gebiet der Diagrammerstellung beschäftigt sich häufig mit Methoden und Algorithmen, die Diagramme automatisiert produzieren und aufbereiten. Dies beinhaltet gewöhnlicher Weise eher die Generierung und Transformation der Diagramme oder automatisches Diagramm-Layout. Obwohl das Erstellen von Diagrammen in digitalen Editoren heute gängige Praxis ist, sind diese auf traditionelle „Point und Click“ Operationen beschränkt; beispielsweise das Einfügen von Objekten per Drag&Drop aus einer Toolbar oder das Umschalten von Zuständen per Schaltflächen oder Menüs (Frisch et al. 2009).

Um die genannte Beschränktheit zu umgehen, haben FRISCH ET AL. die Verwendung von Diagrammbearbeitungstechniken auf interaktiven Displays untersucht, die Multi-Touch und Stift-Gesten verwenden. Sie argumentieren, dass das Zeichnen mit Stiften eine natürliche menschliche Eigenschaft ist und deshalb das Freihandzeichnen (freehand *sketching*) oftmals bevorzugt wird – nicht zuletzt aus diesem Grund werden in vielen Situationen in der Praxis Diagramme auf Whiteboards oder Flip Charts angefertigt. Dieser informelle Weg hat jedoch den Nachteil, dass die ad-hoc erstellten Graphen in der Konsequenz meistens digital neu modelliert werden müssen (Frisch et al. 2009).

Abbildung 3.18 - Gesten und digitales Sketching[14]

Werkzeuge, die den Vorgang des *Digital Sketching* zur Einhaltung einer gewissen Formalität unterstützen, sind aber üblicherweise weiterhin rein auf Stifteingabe oder Single-Touch Interaktionen reduziert (Abbildung 3.18 a)-b)). Zwar bieten sie Basisfunktionen wie das (Re)Arrangieren, Gruppieren oder Skalieren von Elementen an, aber bei Funktionen wie z. B. Zoomen oder dem Ändern des Knotentyps müssen die Anwender sich weiterhin anstrengend durch Menüs navigieren, was den Editierungsprozess erheblich verlangsamt. Aus diesem Grund schlagen die genannten Autoren die Kombination aus Stifteingabe und Multi-Touch Unterstützung vor, mit der bestimmte Aufgaben wie das Löschen oder Kopieren von Knoten mit zwei Händen einfach durchzuführen sind, ohne den Editierungsprozess signifikant zu unterbrechen. Ein Beispiel für das Kopieren zeigt Abbildung 3.18 c)-e), wo direkt durch die Verwendung von Multi-Touch in den Kopiermodus gewechselt wird.

Zur Erreichung ihrer genannten Ziele wurde eine Studie durchgeführt, bei der die Anwender eine definierte Anzahl an Aufgaben auf einem Tabletop mit Stiftunterstützung in gegebener Reihenfolge ausführen mussten. Die Vorgehensweise wurde dabei mit einer Videokamera festgehalten, um später die durch die Probanden vollzogenen Handlungen zu analysieren. Um sicherzustellen, dass die Ergebnisse für eine Vielzahl an Diagrammtypen anwendbar sind, wurde eine elementare Form von Node-Link Diagrammen verwendet, bei der einfache Rechtecke die Knoten verkörperten, welche durch gerichtete oder ungerichtete Kanten miteinander verbunden wurden.

Beobachtet wurde, dass die gestellten Aufgaben durch eine Vielzahl verschiedener Gesten (658!) durch die Probanden gelöst wurden; im Umkehrschluss bedeutet das, dass keine Aufgabe durch ein und dieselbe, eindeutige Geste bearbeitet wurde. Hohe Variationen (33) waren bei der *Copy Node* Aufgabe beobachtbar, die wenigsten (13) bei der *Select Node* Aufgabe. Darüber hinaus wurden zwei grundlegende Klassen mentaler Modelle identifiziert, die sich die Anwender zu Nutze machten: *Sketching* sowie *Structural Editing*, also strukturelles Editieren. Während sich letztere stark an das Verhalten heutiger Editoren aus dem Desktopbereich anlehnt, die auf Manipulation der Diagrammstruktur ausgelegt sind, geht es bei ersterem um das physische Imitieren der realen Welt durch echtes Zeichnen. Zusätzlich fallen alle Gesten mit metaphorischem Charakter in diese Klassifizierung (z. B. das Wiping von Knoten, s. Abbildung 3.19, Aufgabe 5.), obwohl diese nicht zwingend mit einem Stift durchgeführt werden müssen.

[14] Quelle: (Frisch et al. 2009), S.2

Ergebnis der Untersuchung ist ein Set aus Gesten, die für die Diagrammbearbeitung vorgesehen sind. Einen Auszug aus diesem Set, das sich mit allen Ausprägungen im Anhang befindet (s. Abbildung 7.3), zeigt Abbildung 3.19. Zu den Designzielen zählte, Unterstützung sowohl für das *Sketching* als auch für das *Structural Editing* zu gewährleisten; jedoch unter der Prämisse, diese Unterstützung nicht durch Neueinführung spezieller Gesten zu erschweren und den Bearbeitungsprozess nicht durch zeitaufwendige Menünavigation zu lähmen. Um Konflikte zu vermeiden, wurde durch die Autoren an jedem Knoten eine interaktive Randregion hinzugefügt, die die Bearbeitung erleichtert.

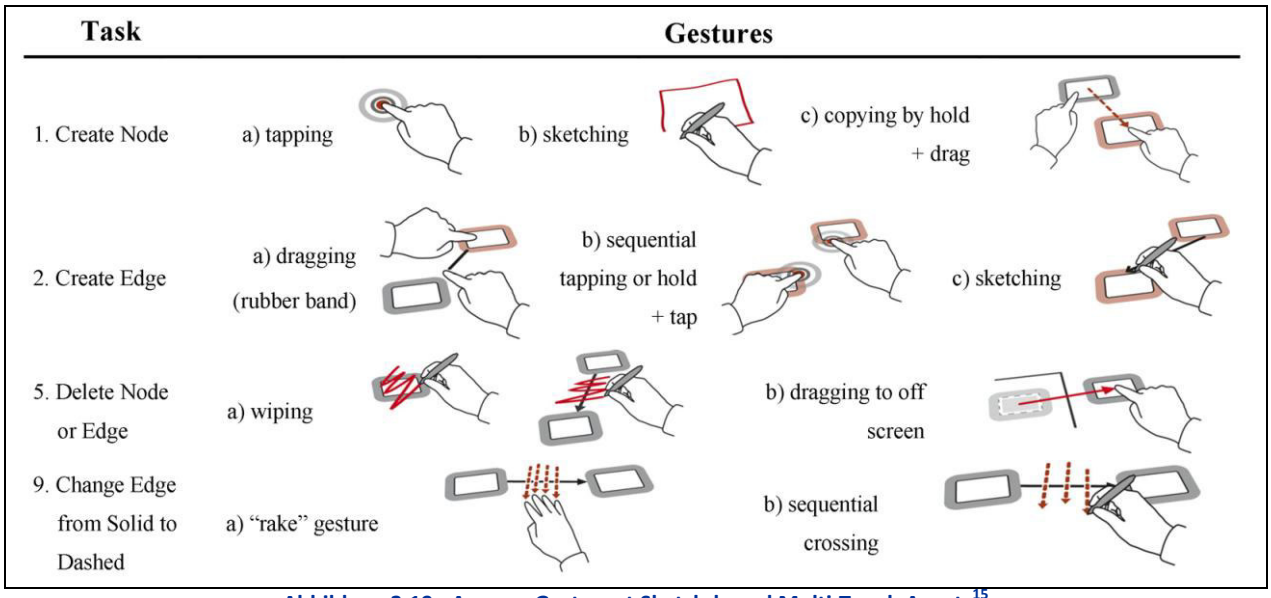


Abbildung 3.19 - Auszug: Gestenset Sketch-based Multi-Touch Ansatz[15]

Abbildung 3.19 zeigt sehr eindrucksvoll die Trennung von Sketching und Structural Editing. Will man beispielsweise einen neuen Knoten erstellen (Task 1), kann man dies durch 1.a Tapping oder 1.b Sketching bewerkstelligen. Sind schon Knoten vorhanden, können neue auch einfach durch die Multi-Touch Geste 1.c kopiert werden, bei der ein Knoten mit einer Hand festgehalten wird und durch Dragging dieses Knotens mit der anderen Hand ein neuer erzeugt wird. Zur Erzeugung einer Kante zwischen den Knoten (Task 2) gibt es wieder verschiedene Möglichkeiten, nämlich durch 2.a Dragging, 2.b entweder sequentielles Tapping oder Halten des Anfangs- und Tippen des Endknotens, oder 2.c erneut durch Zeichnen der Kante.

Hierbei kommt erstmals die erwähnte Randregion des Knotens zum Einsatz, da diese beim Dragging in 2.a zwingend berührt werden muss, um die Geste von 1.c abzugrenzen; es wäre sonst nicht eindeutig, ob ein neuer Knoten oder eine Kante erzeugt werden soll. Ebenso muss bei 2.b in die Randregion getippt werden, da ein Tippen des Knotens selbst seine Auswahl bzw. dann Mehrfachauswahl zur Folge hätte. Zum Löschen eines Knotens oder einer Kante (Task 5) benutzen die zitierten Autoren eine sehr natürliche Geste – das Durchstreichen (vgl. 5.a *Wiping*), das mit dem Stift analog zur realen Welt durchzuführen ist. Alternativ können Objekte auch einfach von der Arbeitsfläche gezogen werden, um sie zu entfernen (5.b off screen dragging).

Um die Art der Kante von „durchgezogen" auf „gestrichelt" zu verändern (Task 9), wurde bemerkenswert einfach die *Rake* Geste etabliert, bei der mit mehreren Fingern einer Hand über die Linie „geharkt" wird. Alternativ kann der gleiche Effekt durch sequentielles kreuzen der Linie mit dem Stift erreicht werden. Im nächsten Abschnitt geht es nun um einen Ansatz, der gänzlich auf die Verwendung

[15] Quelle: (Frisch et al. 2009), S. 8; modifiziert

von MTTs zu verzichten versucht, bzw. abschließend dessen Verwendung durch individuelle Paletten optimieren will.

3.2.1.2 Sketch-basierte Interaktionstechnik mit Sketchbooks und digitalen Schablonen

In einer anderen Veröffentlichung haben DACHSELT ET AL. den Versuch unternommen, die Flexibilität von Papier mit digitalen Features aufzuwerten, um so das spontane Skizzieren zu optimieren. Sie schlagen ein weiteres Mal die Benutzung von digitalen Stiften vor, jedoch in einem leicht veränderten Kontext. Ihnen geht es darum, ein eventuelles Fehlen eines Touchscreens bzw. Whiteboards oder das zeitraubende Hochfahren dieser Geräte zu kompensieren, um die spontane Entwicklung von Ideen nicht unnötig zu verzögern. Obwohl es nach ihrer Untersuchung schon einige Werkzeuge gibt, mit denen Diagramme effizient digital erzeugt werden können, mangelt es weiterhin an den Vorzügen, die das Zeichnen auf Papier mit sich bringen. (Dachselt et al. 2008)

Mit Hilfe der eingesetzten Technologie soll es Anwendern einfacher gemacht werden, UML Diagramme „from scratch“ zu entwickeln, aber gleichzeitig eine nahtlose Schnittstelle der entworfenen Diagramme zwischen Papier und interaktiven Oberflächen herzustellen – und das in Echtzeit oder bei anschließender Übertragung. Sie ist im Single-User Umfeld ebenso einsetzbar wie für Multi-User Anwendungen. Dafür wurde die für dieses Anwendungsgebiet typische *Anoto Technologie* verwendet, die mit einem Bildsensor und einer Infrarotkamera ausgestattete, digitale Stifte benutzt. Diese Sensorik wertet ein Punktmuster (*Anoto dot pattern*) aus, das nahezu unsichtbar auf das Papier aufgetragen ist und auf diese Weise mit jedem Bild genügend Informationen liefert, um die Position des Stiftes auf dem Papier zu bestimmen; oder besser gesagt das, was der Anwender geschrieben oder gezeichnet hat.

Die Anoto Technologie bietet drei verschiedene Wege, Daten vom Stift an die Applikation zu übertragen. Entweder, man setzt den Stift in eine mit der Applikation verbundene Docking-Station ein, was einer stationären Lösung entspräche; oder man wählt mit dem Stift auf das Papier aufgedruckte spezielle Boxen aus, die einen Bluetooth-Transfer initiieren. Alternativ kann permanentes Bluetooth Streaming verwendet werden. DACHSELT ET AL. verwenden diese Technologie in ihrem Ansatz aber nicht nur mit Papier, sondern auch im Zusammenspiel mit einem Multi-Pen befähigtem Tabletop mit Rück-projektionsbildschirm. Das Anoto Pattern ist in diesem Setup unter der Oberfläche des Bildschirms angebracht und der mit dem Stift erfasste Inhalt wird direkt an den Tabletop übermittelt.

Ein Vorschlag der Kombination von UML Skizzierungstools mit digitalen Stiften und Papier zeigt Abbildung 3.20. Die genannten Autoren tendieren zu der Verwendung von ihnen vorgeschlagener *UML-Sketchbooks*, welche auf jeder Seite eine größere Fläche zum Skizzieren des Diagramms vorsehen, ergänzt um einen definierten Bereich mit Auswahlboxen verschiedener Diagrammtypen. Per Tippen mit dem Stift auf den gewünschten Typ und abschließendem *Send* wird der Übertragungsprozess gestartet. Auf diese Weise kann der Skizze eine definierte Bedeutung zugesprochen werden, um eine verbesserte Erkennung zu unterstützen.

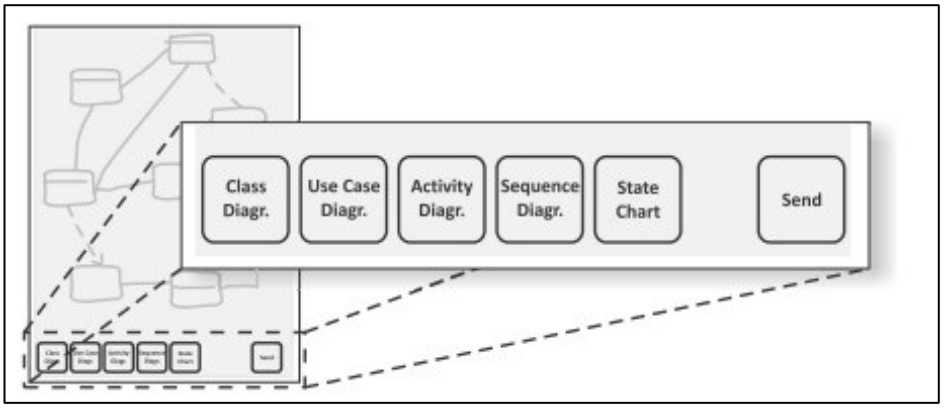


Abbildung 3.20 - Sketch-basierter Enstehungsprozess in UML-Sketchbooks[16]

Die Applikation versucht bspw. nach Auswahl von „Class Diagr." nun, so viele Teile der übermittelten Skizze wie möglich in Klassendiagramme zu konvertieren, auch wenn diese nicht konform gezeichnet wurden. Denkbar ist auch die Methode, dass nach dem Antippen der Box mit dem Stift ein Einkreisen der gewünschten Elemente der Skizze folgt. Software Designer können dadurch verschiedene Diagrammtypen dem Inhalt einer Seite zuordnen. Weiterhin haben DACHSELT ET AL. auch über die Verwendung mehrerer Seiten Papier im Kollektiv nachgedacht, da der verfügbare Platz den größten Nachteil der Verwendung von Papier darstellt. Dadurch wird es Benutzern ermöglicht, einfach weitere Seiten Papier an das derzeit verwendete anzuhängen, oder aufgeteilte Arbeit gezielt zusammen zu bringen. Der letztgenannte Grund ist besonders interessant für das kollaborative Arbeitsklima.

Bei Verwendung des zuvor genannten Tabletops legen die Autoren die Benutzung individualisierter UML-Paletten nahe. Diese beinhalten UML-spezifische Elemente, beispielsweise angeordnet in einem Gitterlayout (s. Abbildung 3.21). Sie sind auf Anoto-aktiviertem Papier gedruckt und können durch den Software-Designer wie eine Karte in der Hand gehalten werden. Die Verknüpfung zwischen Stift und Tabletop-Applikation ermöglicht es, Elemente per Stift von der Palette durch Tapping „aufzunehmen" und direkt auf dem Tisch zu platzieren. Die ausgewählten Elemente erscheinen nach Berührung des Stiftes mit dem Tisch unterhalb seiner Spitze und können auf diese Weise direkt platziert werden.

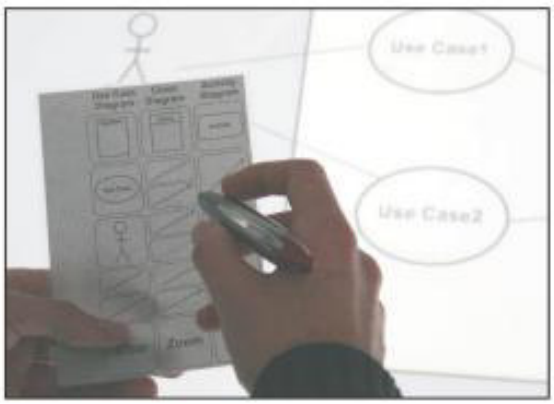


Abbildung 3.21 - Einsatz von digitalen Schablonen (UML-Paletten)[17]

Die platzierten Elemente können dann durch Beziehungen miteinander verbunden werden, die ebenfalls von der Palette stammen; auf diese Weise lassen sich ad hoc formale UML-Diagramme realisieren. Besonders geeignet ist dieser Ansatz für das gemeinsame Arbeiten, da jeder Anwender seine eigene,

[16] Quelle: (Dachselt et al. 2008), S. 1
[17] Ebd.

individuell angepasste Palette mitbringen kann – was gleichzeitig als Gedächtnisunterstützung dient, sollte man sich nicht die gesamte UML Syntax merken können. Darüber hinaus können Elemente gut erreicht werden, weil sie nicht auf dem großen Bildschirm verteilt sind. Auch kann dieser Ansatz mit Sketching zusammen verwendet werden, jedoch ist es hierbei erforderlich, falsch oder nicht erkannte Objekte nachträglich mit Hilfe der Palette zu ersetzen.

Zusammenfassend lässt sich sagen, dass sich die vorgestellten Techniken der vorangegangenen beiden Abschnitte 1:1 für Erstellung von Aufgabenmodellen verwenden lassen. Das Gestenset funktioniert uneingeschränkt auf alle Arten von Diagrammen, und nach Anpassungen der vorgestellten Sketchbooks oder Paletten wären diese ebenso für den Einsatz in der Aufgabenmodellierung geeignet. Insgesamt wurden bisher praxistaugliche Techniken zur Erstellung von Node-Link Diagrammen analysiert, die für eine Verwendung im Konzept geeignet sind.

3.2.1.3 Off-Screen Visualisierung für Node-Link Diagramme

Ein großes Problem bei der Visualisierung von Diagrammen ist deren Komplexität (und damit der verbrauchte Raum) versus die für die Darstellung zur Verfügung stehende Fläche. Geht das Diagramm über die Grenzen der Arbeitsfläche (bspw. des Bildschirms) hinaus, wird ein Schwenken dieser Fläche erforderlich, um alle erzeugten Objekte weiterhin erreichen zu können. Während in Fenstern die Ansicht klassischerweise nur per Scroll-Balken verschoben werden konnte (*Panning*), haben sich im Laufe der Zeit für diesen Vorgang Gesten entwickelt, mit deren Hilfe man mit der Maus – oder auf Touchscreens mit den Fingern – die Oberfläche per sogenanntem *Dragging* verschieben kann.

Eine weitere Möglichkeit, die zweidimensionale Darstellung effektiver ausnutzen zu können, stellt das *Zooming* dar. Durch Herauszoomen werden alle Diagrammelemente gleichzeitig verkleinert, sodass man einen komprimierten Überblick über das gesamte Diagramm erhält. Aufgrund der zwangsweise verringerten Lesbarkeit ist ein effizientes Arbeiten ohne ein anschließendes Hereinzoomen aber in diesem Zustand meist nicht mehr gewährleistet. Dies ist einer der Gründe, warum einige der in Kapitel 3.1 untersuchten Editoren eine Miniaturkarte zur Orientierung anbieten, um auf dem gewünschten Abschnitt des Modells wieder zu vergrößern. Beide Varianten werden vorwiegend in der Kartennavigation eingesetzt, und die meisten der heute verfügbaren Editoren adaptieren diese Steuerung an Stelle von einer Navigation basierend auf Diagrammtopologie oder -semantik.

Um die genannten Probleme zu kompensieren, haben sich FRISCH ET AL. mit einer Methodik beschäftigt, die Informationen zu Diagrammobjekten außerhalb des sichtbaren Bereiches für den Anwender zugänglich macht. Diese nennt sich *Off-Screen Visualisierung* für Node-Link Diagramme, da sie Elemente außerhalb des sichtbaren Bereiches des Diagramms – also „off-screen“ – zur erleichterten Navigation im Randbereich bereitstellt (Frisch & Dachselt 2010). Sie haben eine Applikation entwickelt, die Off-Screen Visualisierungstechniken für Node-Link Diagramme im Allgemeinen und UML-Diagramme im Speziellen realisiert. Grundgedanke bei dieser Methode ist das Integrieren einer kontextuellen Sicht auf alle vom aktuellen Bildausschnitt (*Viewport*) abgeschnittenen Knoten. Außerdem werden semantische Informationen integriert, indem Eigenschaften der Kanten wie Label und Multiplizitäten oder die Arten der verbundenen, abgeschnittenen Knoten angezeigt werden. Für ein UML-Klassendiagramm zeigt Abbildung 3.22 beispielhaft den Viewport der Oberfläche:

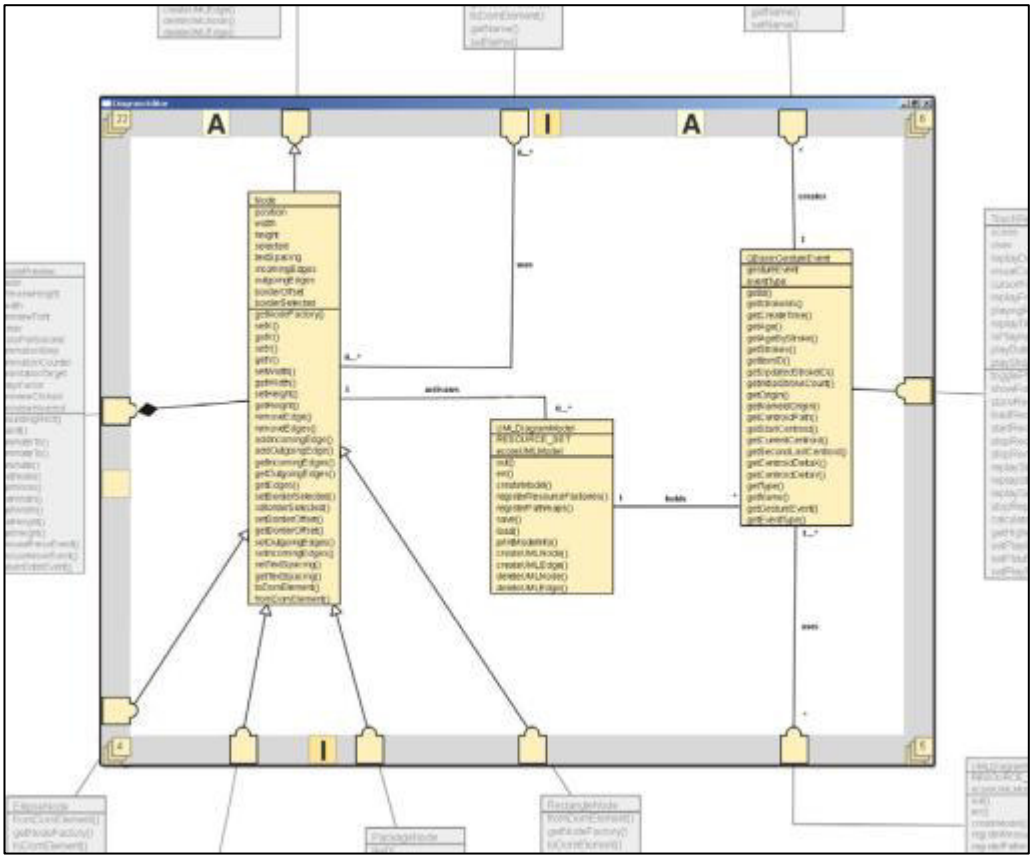


Abbildung 3.22 - Viewport eines UML Klassendiagramms mit Off-Screen Visualisierung[18]

Dieses Diagramm besteht aus 51 Klassen, von denen drei per Hineinzoomen fokussiert wurden. Wie man der Grafik entnehmen kann, ist der interaktive Randbereich grau dargestellt. Er nimmt so genannte *Proxy*-Elemente auf, die analog zu den aus der Netzwerktechnik bekannten Kommunikations-schnittstellen Verbindungen zu außerhalb gelegenen Knoten bündeln, bzw. deren eigentliche Position im Raum an den Rand projizieren. Die Art der Verbindung (in einem UML-Klassendiagramm typischerweise *Assoziation*, *Generalisierung* oder *Aggregation;* repräsentiert durch Symbolik am Anfang/Ende der Kanten) sowie eventuell vorhandene *Multiplizitäten* werden zur verbesserten Orientierung direkt an den Proxys widergespiegelt, und nicht erst bei Kontakt mit dem eigentlichen Knoten. Auf diese Weise können sie auch direkt vom Anwender bearbeitet werden, da sich dieser normalerweise in dem von ihm oder ihr erstellten Diagramm orientieren kann und ggf. weiß, welchen Knoten der Proxy repräsentiert.

Zusätzlich wird die Art der verknüpften Klasse ebenfalls zugänglich gemacht; in diesem Beispiel ein „I“ für Interface bzw. ein „A“ für Abstraktion in Verbindung mit farblicher Absetzung von den Proxyknoten. Die Interaktivität der Knoten wird dadurch hergestellt, dass die Proxyelemente nicht nur auf den assoziierten Off-Screen-Knoten verweisen, sondern direkt anklickbar sind und zum entsprechenden Knoten navigieren sowie durch Panning/Zooming die Ansicht automatisch an den Zielknoten anpassen. Weiterhin können beim Hovern mit der Maus Basisinformationen in einer Vorschau angezeigt werden, ohne das zu dem durch den Proxy ausgewählten Knoten gesprungen werden muss. Bemerkenswert ist, dass sich die Proxy Elemente beim Verschieben oder Zoomen der Arbeitsfläche dynamisch anpassen. Da die Autoren die Technik auf hohe Skalierbarkeit ausgelegt haben, funktioniert dies auch dann, wenn sich im Diagramm mehrere hundert Knoten befinden. Obwohl andere Off-Screen Techniken bei diesen

[18] Quelle: (Frisch & Dachselt 2010), S. 163

Größenordnungen oft unter Grafikfehlern bei der Proxydarstellung leiden, versucht dieser Ansatz der Problematik mit automatischer Clusterbildung und interaktivem Filtern zu begegnen.

Abbildung 3.23 beschreibt die Herangehensweise bei der Bildung der Proxy-Elemente. Betrachtet man die Klassen C1 und C2 fällt auf, dass beide mit der Klasse C3 als Oberklasse verbunden sind (Generalisierung, angedeutet durch die UML-spezifischen, nicht ausgefüllten Pfeile). Ohne Benutzung eines Proxys würde diese Information verloren gehen, da die grauen Pfeile verwendet würden. Der Proxy 3‘ kanalisiert die Verbindung zu C3 und macht das Verhältnis der Klasse zu C1 und C2 zugänglich, ohne dass die Klasse selbst durch den Anwender wahrgenommen wird.

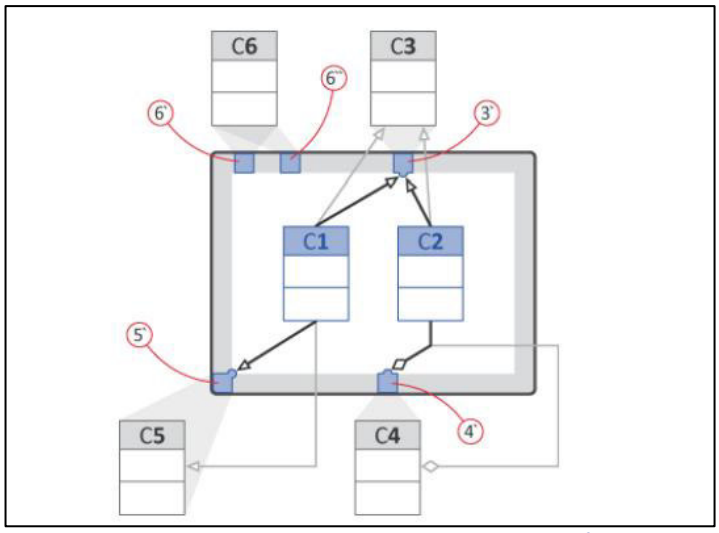


Abbildung 3.23 - Bildung der Proxy-Elemente[19]

Dieser Proxy wird automatisch freigegeben, sobald durch Verschieben oder Zoomen der weißen Fläche die Klasse C3 erreicht wird. Das bedeutet, das Element verschwindet vom Rand und die Kanten zeigen wie ursprünglich auf die Klasse C3. Die geometrische Projektion der Knoten an den Rand des Viewports lässt sich auf zwei Arten realisieren: *orthogonal* oder *radial.* Bei orthogonaler Projektion werden die Knoten im rechten Winkel an den Rand projiziert, für radiale Projektion ist der Ausgangspunkt die Mitte des Viewports. Beide Varianten werden für die Klasse C6 dargestellt, wobei der Proxy 6‘ die orthogonale Projektion und 6‘‘ die radiale Variante verwendet. Alle anderen Klassen verwenden orthogonale Projektion.

Unabhängig davon, welche Variante Verwendung findet: Ein Problem der Projektion ist immer das Ändern der Route der repräsentierten Kante, da sich die projizierte Kante am Proxy-Element orientiert. Dies kann zu signifikanten Änderungen der ursprünglichen Darstellung führen, wie Abbildung 3.23 für die Klassen C5 (Generalisierung) und C4 (Aggregation) zeigt. Besonders problematisch stellt sich dies bei Mehr-Segment-Kanten dar. Die Kanten werden gebogen und neue Beugungspunkte anhand des letzten On-Screen Beugungspunktes vergeben. Verletzungen der Grundregeln ästhetischen Diagrammdesigns können die Folge sein, genauso wie erschwertes Verstehen des Nutzers für das permanente Ändern der Kanten beim Panning/Zooming. Um diesem Umstand entgegenzuwirken, haben FRISCH ET AL. zwei Methoden vorgeschlagen, um dem Benutzer das Verstehen der erzwungenen Änderung der Route so einfach wie möglich zu machen. Weiterhin soll durch die in Abbildung 3.24 und Abbildung 3.25 vorgestellten Vorgehensweisen wenigstens der sichtbare Teil der ursprünglichen Route gewahrt werden.

[19] Quelle: (Frisch & Dachselt 2010), S. 166

Um das durch den Proxy erzeugte Kantensegment zu kennzeichnen, wurde in Erwägung gezogen, dieses in einer anderen Farbe als die ursprüngliche Kante darzustellen (s. Abbildung 3.24). Weiterhin wurde zur besseren Orientierung festgelegt, dass die Originalkante bis zur Randregion wie ursprünglich verläuft. Ab dem Berührungspunkt mit dem Rand werden dann *am Rand entlang* die neuen Kantensegmente in Richtung Proxy aufgebaut. Die Abbildung zeigt links, wie sich die neue Route analog zur originalen verhält. Rechts wird die Technik anhand einer Bezierkurve impliziert.

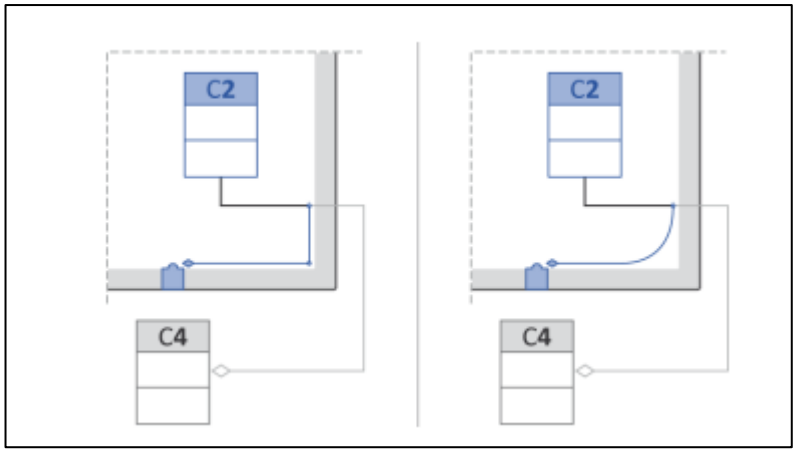


Abbildung 3.24 - Routen der Kanten bei Proxys „along the border“[20]

Ein anderes Verfahren stellen die Entwickler mit Abbildung 3.25 vor. Beim Projizieren *entlang der Kanten* werden die Proxyelemente beim Berührungspunkt der Originalkante mit dem Rand erzeugt, und nicht orthogonal oder radial. Am deutlichsten sieht man dies bei den Proxys 4‘ und 5‘; auf diese Weise bleibt die Orientierung des Diagramms erhalten. Da nun aber bei einem Klick auf den Proxy sich das Diagramm nicht in die erwartete Richtung nach rechts bewegt (sondern in diesem Fall nach unten), könnte dies erneut zu erschwertem Verständnis des Anwenders führen. Um das zu kompensieren, wurde die Überlegung angestellt, einen temporären Proxy 4‘‘ in einer anderen Farbe einzuführen, der beim Berühren des Proxys 4‘ mit dem Mauscursor eingeblendet wird. Dieser suggeriert dem Benutzer die Richtung, in die sich das Diagramm bei einem Klick bewegen wird.

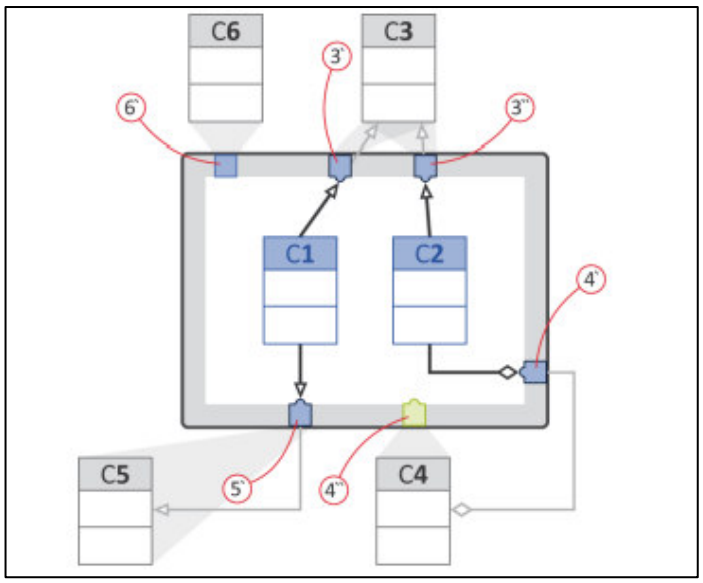


Abbildung 3.25 - Routen der Kanten bei Proxys „along the edges“[21]

[20] Quelle: (Frisch & Dachselt 2010), S.167
[21] Ebd.

Folge der gemachten Änderung ist jedoch, dass sich der Proxy 3‘ nun in die zwei Proxys 3‘ und 3‘‘ aufteilen müsste, und nicht mehr durch einen einzigen repräsentiert werden kann. Dadurch geht evtl. die Information verloren, dass sich beide Proxys auf ein und dieselbe Klasse beziehen, dies ist jedoch ein vertretbarer Verlust verglichen mit den gewonnenen Vorteilen.

Gewinnt das Diagramm an Komplexität, kann es am Rand zu Fehlern kommen (sog. *Clutter*). Zur Vermeidung des Clutter schlagen die Autoren die Clusterbildung vor und erörtern erneut zwei Methoden, einmal das *geometrische Clustern* und das *semantische Clustern.* Fall 1 der Abbildung 3.26 zeigt das geometrische Cluster. Es kommt zum Einsatz, wenn mehr als ein Knoten auf ein und dieselbe Position projiziert würde. Da sich die Klassen C3 und C4 orthogonal hintereinander befinden (aus der Sicht von C1), wird ein Stapelproxy eingefügt, der die Anzahl der sich dahinter verbergenden Elemente anzeigt. Der Zähler wird ebenso dynamisch angepasst, sollte der Viewport geändert werden und Klassenknoten sichtbar werden, die zu dem Stapel gehören. Weiterhin werden bei orthogonaler Projektion Proxyelemente für Off-Screen Knoten in Richtung der Ecken des Randes erzeugt (s. C6 und C7).

Mit dieser Technik werden Stapel auch dann erzeugt, wenn noch genügend Platz vorhanden ist. Aus diesem Grund wurde durch die Entwickler ein Algorithmus *avoid cluster* entwickelt, der die Nachbarschaft der Proxys überwacht und bei genügend Freiraum interveniert, sodass nicht zwingend Cluster aufgebaut werden. Dieser ist aber nicht für jeden Diagrammtyp geeignet, da z. B. ein vertikales Anzeigen von Elementen, die in Wirklichkeit horizontal angeordnet sind, eher verwirrt als nützt.

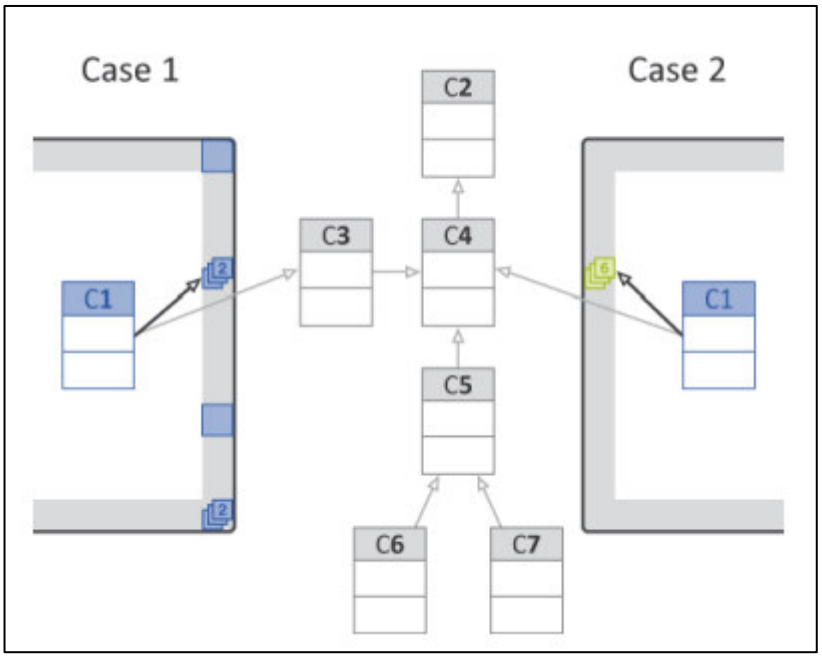


Abbildung 3.26 - Clusterbildung[22]

Fall zwei auf der rechten Seite zeigt das semantische Clustern. Es basiert auf semantischen Regeln bezogen auf die Diagrammnotation. Für UML wurde in diesem Beispiel die Vererbung herangezogen, möglich wären auch Klassenzusammengehörigkeit in einem Paket oder die Verbindung zwecks Aggregations- oder Kompositionsbeziehungen. Die Klasse C1 gehört semantisch zu einem Off-Screen Vererbungscluster, dem weitere sechs Klassen angehören. Alle Klassen, die direkt oder indirekt von C2

[22] Quelle: (Frisch & Dachselt 2010), S. 168

erben, werden zu einem Cluster zusammengefasst. Da C1 mit C4 verbunden ist, erfolgt die Projektion des Stapelproxys auf orthogonaler Höhe mit diesem Knoten.

Die für die Off-Screen Visualisierung verwendete Symbolik, die sich im interaktiven Randbereich wiederfindet, ist noch einmal detailliert in Abbildung 3.27 dargestellt. Die verwendeten Buchstaben können sehr leicht an die Anforderungen der Aufgabenmodellierung angepasst werden, indem beispielsweise ein „U“ für *User Task*, ein „S“ für *System Task* und ein „I“ für *Interaction Task* etc. eingeführt würde. Auf diese Weise könnten die nächstgelegenen Knoten einfach identifiziert werden. Für die Proxydarstellung ergäben sich verschiedene Möglichkeiten. So könnten nach dem Ansatz von VTMB die temporalen Relationen direkt am Proxy-Element wiedergegeben werden, oder bei Verwendung der CTTE-Notation durchaus auch innerhalb der Kanten in Richtung des Proxys.

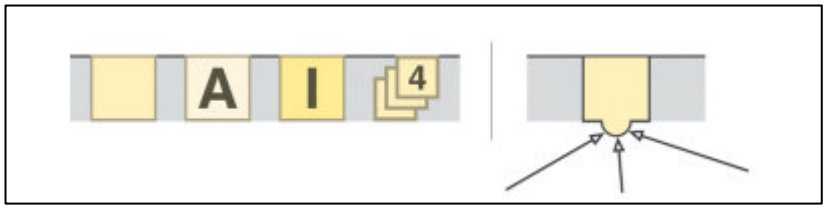


Abbildung 3.27 - Off-Screen Visualisierung: Symbolik[23]

Abschließend lässt sich sagen, dass sich die vorgestellte Technik effektiv für die Aufgabenmodellierung nutzen ließe; und obwohl sie grundsätzlich als Desktopanwendung konzipiert wurde, das gleiche Potential auf einem MTT entfalten kann. Die Probleme, die bei komplexen Modellen entstehen, werden durch die Methodik erfolgreich bekämpft und entwickeln eine bisher nie da gewesene Dynamik bzgl. der Bearbeitung großer Diagramme.

3.2.1.4 Explorationstechniken für Node-Link Diagramme

Mit zunehmender Komplexität von Modellen nimmt zwangsläufig die Lesbarkeit bei der grafischen Repräsentation immer weiter ab. Verantwortlich dafür sind häufig die Kanten, die Knoten miteinander verbinden. Wenn ihre Anzahl erheblich anwächst und sie sich überlagern oder kreuzen, wird es zunehmend schwieriger zu erkennen, welche Kanten welche Knoten miteinander verbinden. Um die Analyse eines erzeugten High-Level Graphen zu erleichtern, haben sich SCHMIDT ET AL. mit Explorationstechniken für Node-Link Diagramme beschäftigt, die aus dem Desktopbereich bekannte Techniken auf einen Multi-Touch Bildschirm portieren (Schmidt et al. 2010).

Entstanden ist ein „Interaction Technique Set“ mit den Methoden *TouchPlucking*, *TouchPinning*, *TouchStrumming*, *TouchBundeling* sowie *PushLens*, die allesamt durch temporäre Reorganisation beliebiger Kanten die Untersuchung eines vorhandenen Diagramms unterstützen sollen. Dabei betonen die zitierten Autoren, dass es nicht um das Erstellen von Graphen oder das Editieren selbiger geht, sondern rein um deren Analyse. Das ist auch der Grund, wieso von temporärer Reorganisation gesprochen wird, da die Änderungen nicht dauerhaft auf das Diagrammlayout übertragen werden.

Das TouchPlucking ist eine einfache, aber dennoch mächtige Technik, um die Trajektorie der Kanten zu manipulieren. Die Basisfunktion zeigt Abbildung 3.28a, bei der durch Berühren einer Linie und gleichzeitigem Ziehen ihre Kurve in Richtung des Berührpunktes verändert wird, aber die Verbindung zu den Knoten weiterhin erhalten bleibt. Die Linien verhalten sich wie flexible Schnüre, die an beiden En-

[23] Quelle: (Frisch & Dachselt 2010), S. 168

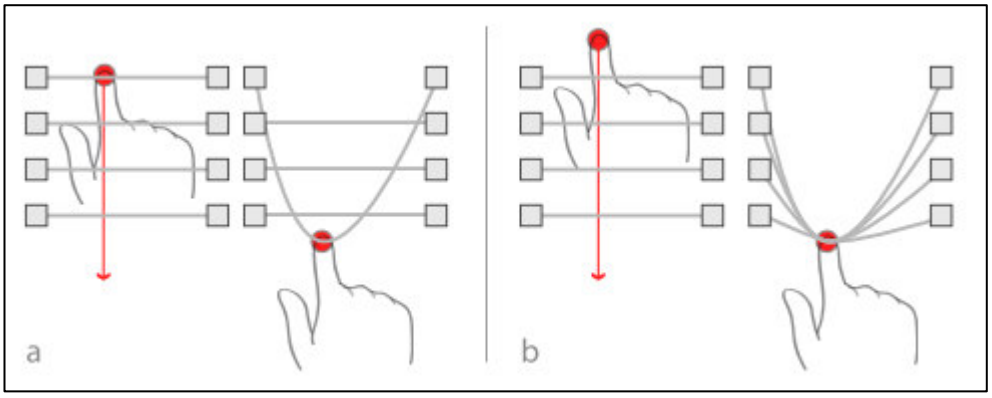


Abbildung 3.28 - TouchPlucking[24]

den fixiert sind. Auf diese Weise lässt sich sehr schnell herausfinden, welche zwei Knoten durch diese Linie miteinander verbunden sind. Alle weiteren Kanten, deren Strecke die gemachte Bewegung tangiert, bleiben davon unberührt; es sei denn der initiale Berührungspunkt umfasst mehrere Kanten, dann werden alle gemeinsam aufgenommen. Will man gleich mehrere Kanten „zupfen“, startet die Geste im Freiraum und alle durch die Bewegung berührten Linien werden zusammen zu flexiblen Bändern (Abbildung 3.28b). Sobald man den Finger von der Oberfläche nimmt, nehmen die Kanten ihre Ursprungsposition wieder ein.

Manchmal kann es schwierig sein, eine spezifische Kante in verdichteten Bereichen mit dem Finger aufzunehmen. Für diesen Fall haben SCHMIDT ET AL. eine Methode des TouchPlucking ausgehend von einem Knoten vorgesehen. Wird der Knoten, zu dem die gewünschte Kante gehört, für mindestens 1,5 Sekunden getippt und gehalten, werden alle seine Kanten verstärkt und in der Farbe geändert, wie auf Abbildung 3.29 zu sehen.

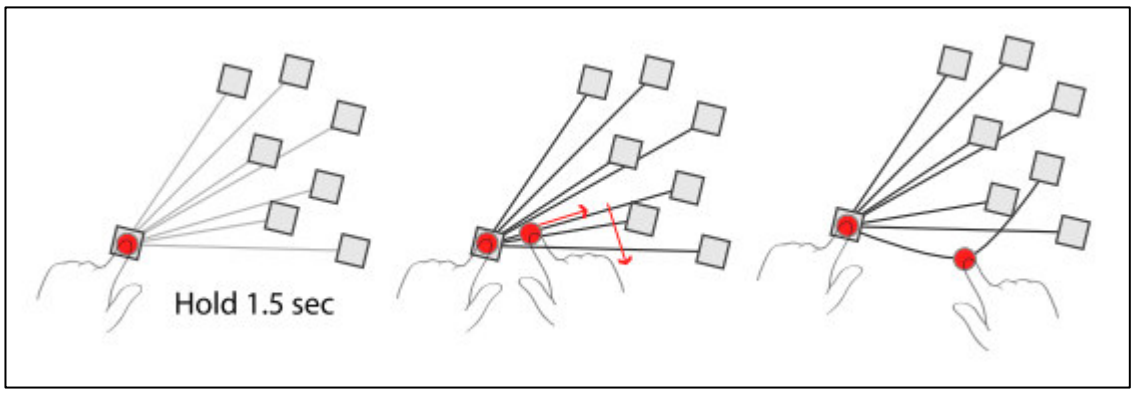


Abbildung 3.29 - TouchPlucking ausgehend vom Knoten[25]

Da dies bis zu diesem Zeitpunkt noch keinen Vorteil gegenüber der Ausgangssituation erbringt (außer vielleicht dass man die Menge der Kanten besser sieht), muss nun in Richtung der designierten Kante mit der anderen Hand eine Drag-Geste ausgeführt werden. Dadurch wird die Kante, die mit der gezogenen Spur am ehesten übereinstimmt, ausgewählt und kann wie vorher erläutert gehandhabt werden. Die erste Hand kann ab diesem Zeitpunkt vom Knoten genommen werden, da die gewünschte Kante jetzt ausgewählt ist.

[24] Quelle: (Schmidt et al. 2010), S. 2
[25] Ebd.

Es kann passieren, dass man nachträglich mehrere Kanten gebündelt ziehen will, obwohl man dies zuvor aufgrund mangelnder Sichtbarkeit nicht erfassen konnte. Mit Multi-Touch ist es möglich, zwei oder mehr TouchPlucking Operationen durch zusammenziehen zu vereinigen. Kommt man mit dem zweiten Finger in die Nähe des ersten, werden beide verknüpft und der freie Finger kann für weitere Operationen genutzt werden.

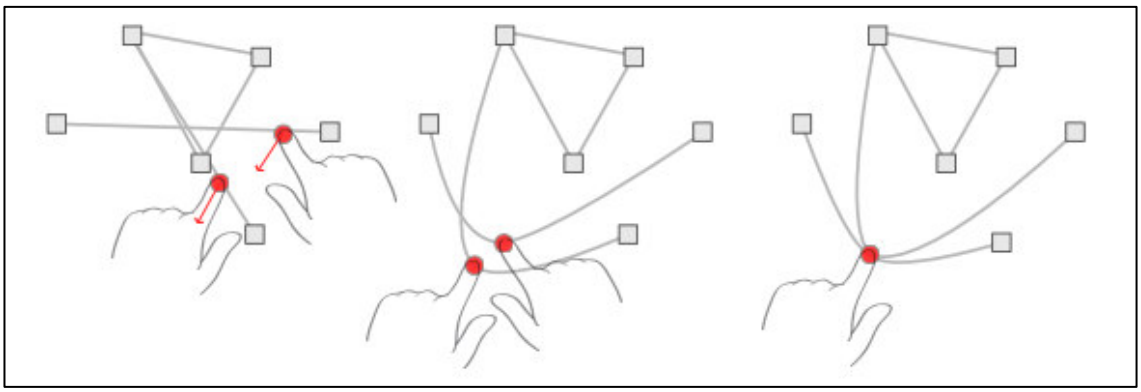


Abbildung 3.30 - Kombination von TouchPlucking Operationen[26]

Wie bereits erwähnt, sind alle Operationen nicht persistent und die zuvor genannten Aktionen werden bereits beim Loslassen des Fingers von der Oberfläche rückgängig gemacht. Jedoch kann es durchaus gewollt sein, das Diagramm in Ruhe zu betrachten und den erzeugten Zustand länger beizubehalten. Für diesen Vorgang führen SCHMIDT ET AL. das TouchPinning ein.

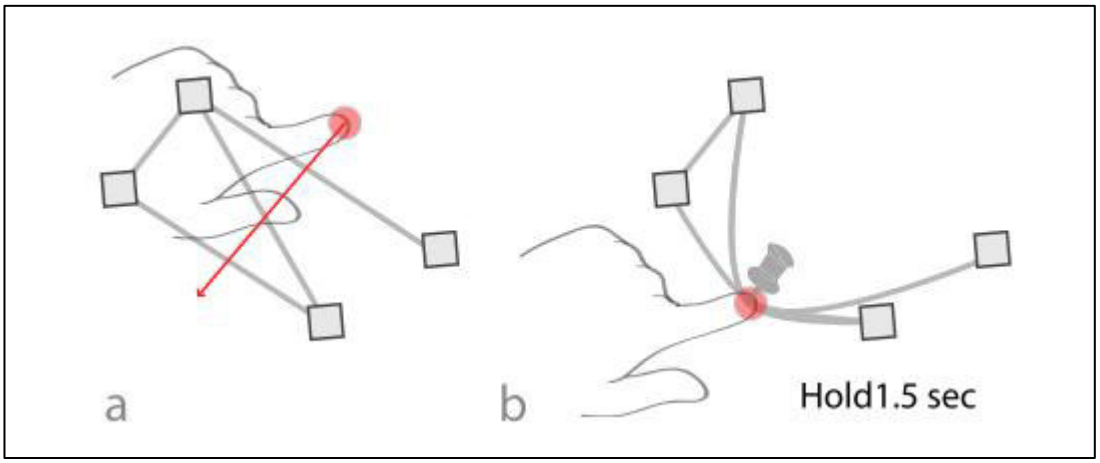


Abbildung 3.31 - TouchPinning[27]

Hat man eine oder mehrere Kanten per TouchPlucking ausgewählt (s. Abbildung 3.31a), können diese verzogenen Linien mit Hilfe einer virtuellen Stecknadel festgepinnt werden. Dazu verweilt man nach der TouchPlucking-Geste für 1,5 Sekunden auf einem Punkt, und es erscheint metaphorisch ein Pin, der die Linien auf Position hält (s. Abbildung 3.31b), auch wenn man den Finger vom Bildschirm nimmt. Diesem Pin können per TouchPlucking von außerhalb weitere Kanten hinzugefügt werden, wenn der ausführende Finger in die Nähe des Pins kommt. Der Pin selbst kann per Touch-Drag wieder zu einer TouchPlucking Operation umgewandelt werden, und durch Tapping wird der Pin entfernt und alle Kanten gehen auf Ausgangsposition zurück. Der Einsatz mehrerer Pins kann die Übersichtlichkeit stark verbessern, da eine Vielzahl verschiedener Kanten an unterschiedlichen Positionen fixiert werden kann.

[26] Quelle: (Schmidt et al. 2010), S. 2
[27] Quelle: (Schmidt et al. 2010), S. 3

TouchStrumming verkörpert eine weitere Möglichkeit, die Verbindung eines oder mehrerer Knoten zu identifizieren. TouchStrumming ist dem TouchPlucking sehr ähnlich, wird aber durch eine schnelle Streifgeste (sog. *Flick*) initiiert. Die Kante verhält sich dabei wie eine Gitarrensaite, die gezupft wird. Durch kurzes Berühren der Kante, schnellem Ziehen und wieder loslassen beginnt die Kante zu schwingen, und zeigt somit ihre Enden respektive die verbundenen Knoten an (vgl. Abbildung 3.32). Die dabei entstehende Amplitude der Schwingung ist proportional zur Länge der durchgeführten Flick-Geste.

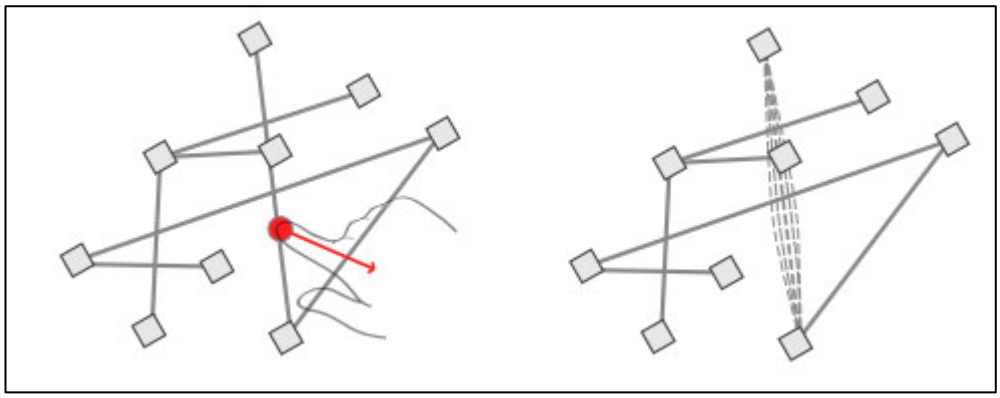


Abbildung 3.32 - TouchStrumming[28]

Dieselbe TouchStrumming-Operation lässt sich auch auf vorhandenen Knoten ausführen, wie Abbildung 3.33 zeigt. Durch die Flick-Geste am Knoten beginnen alle seine Kanten zu schwingen. Durch Benutzung dieser Technik ist es sehr einfach möglich, alle verbundenen Nachbarn des ausgewählten Knotens ausfindig zu machen. Darüber hinaus kann diese Methode parametrisiert werden, sodass sich zum Beispiel die Vibration auf anschließende Kanten mit bspw. geringerer Amplitude ausweiten ließe. Auf diese Weise würde sich die Operation wie ein Beben durch den gesamten verbundenen Graphen schwingen.

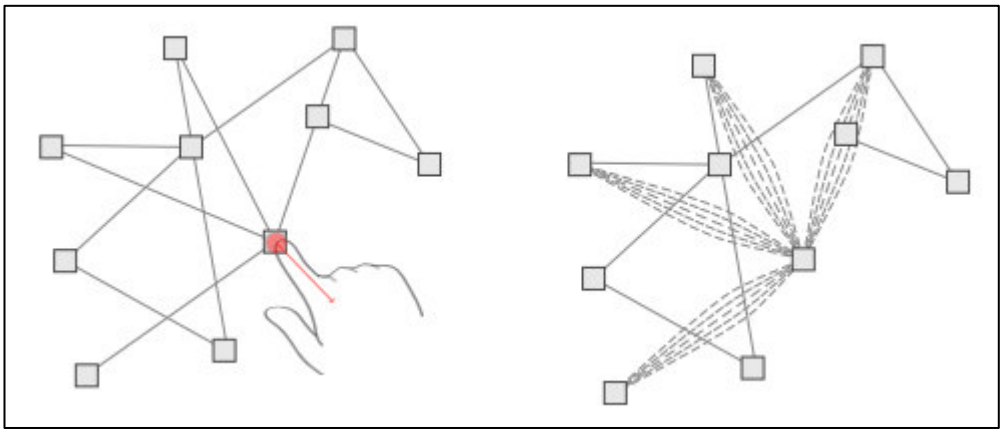


Abbildung 3.33 - TouchStrumming am Knoten[29]

Mit Hilfe von TouchBundeling soll es den zitierten Autoren nach ermöglicht werden, durch Bündelung von Pfaden mehrerer Kanten das Cluttering zu vermeiden, oder die Zusammengehörigkeit von Knotengruppen hervorzuheben. Um ein Bündel zu erzeugen, bewegt man zwei Finger parallel wie einen Schacht, um die gewünschten Kanten einzugrenzen (Abbildung 3.34). Um weitere Kanten aufzunehmen, bewegt man die beiden Berührungspunkte immer weiter. Befindet sich ein Knoten innerhalb der Geste,

[28] Quelle: (Schmidt et al. 2010), S. 3
[29] Ebd.

werden alle seine Kanten zum Bündel hinzugefügt. Die Länge des Bündelsegments ist definiert durch die Strecke der ausgeführten Geste (s. Abbildung 3.34, rechts).

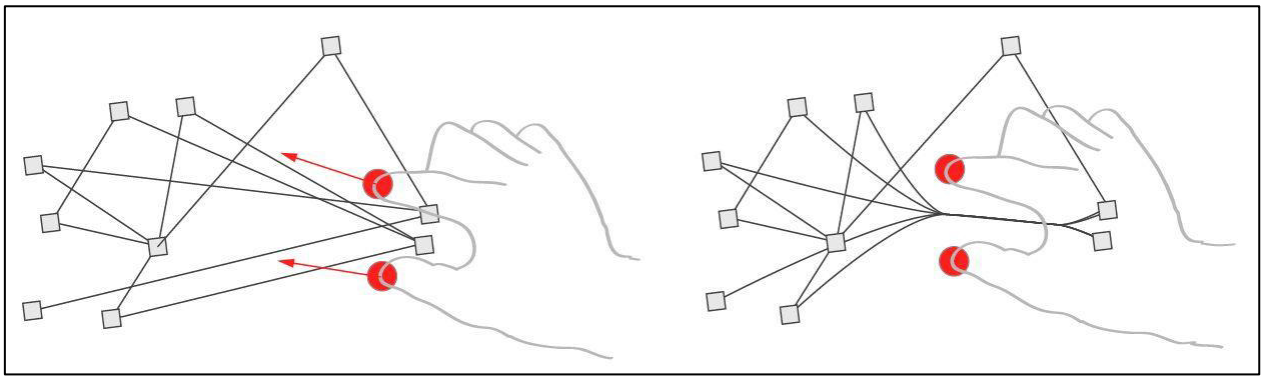


Abbildung 3.34 - TouchBundeling[30]

Bündel bleiben bestehen und können später in Aussehen und Position durch Ziehen des Bündelendes verändert werden. Das Hinzufügen oder Entfernen von Kanten zum oder vom Bündel erfolgt analog zum TouchPlucking. Da die gebündelte Kante als eine eigenständige, aggregierte Kante angesehen werden kann, kann auch sie via TouchPlucking manipuliert werden. Um ein Bündel wieder zu entfernen, genügt ein Antippen des Bündels irgendwo innerhalb des aggregierten Segments.

Die sogenannte PushLens ist ein Objekt mit runder Form innerhalb der Schnittstelle. Kanten von Knoten innerhalb dieser Linse werden nicht verändert. Alle Kanten, die nicht mit dem innen liegenden Knoten verbunden sind, aber mit ihren Pfaden die Linse kreuzen, werden um die Linse herumgelenkt (s. Abbildung 3.35, rechts). Auf diese Weise wird es möglich, Bereiche zu definieren, in denen es kein „Durcheinander“ von Kanten gibt. Enthält eine Linse einen einzigen Knoten, werden seine Kanten und deren Richtungen sofort sichtbar. Umschließt die Linse weitere Knoten, werden die Verbindungen zwischen den eingekreisten Knoten hervorgehoben und gleichzeitig alle anderen Kanten, die keine Verbindung mit diesen besitzen, ausgeblendet. Innerhalb der Linse können alle zuvor vorgestellten Operationen verwendet werden, solange sie nicht ihren Ursprung am Rand der Linse haben.

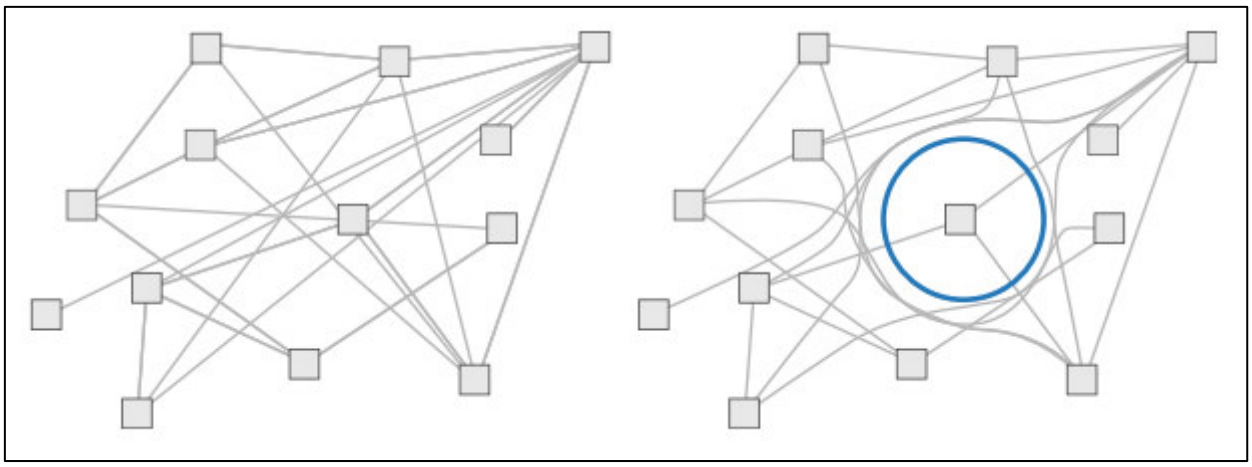


Abbildung 3.35 - PushLens[31]

Die PushLens wird durch eine Gruppe verschiedener Interaktionstechniken manipuliert. Erzeugt wird sie durch eine Drei-Finger-Geste, bei der alle Berührungspunkte räumlich und zeitlich nah beieinander liegen. Sobald sie erzeugt ist, wird die Linse durch einen Kreis repräsentiert, der durch Single-Touch

[30] Quelle: (Schmidt et al. 2010), S. 3
[31] Ebd.

Berührung seines Randes verschoben werden kann. Vergrößerung oder Verkleinerung erfolgt durch eine Pinch-Geste auf zwei Punkten auf dem Rand. Gelöscht werden kann die Linse durch einfaches Antippen der Kontur.

Insgesamt gesehen bilden die hier vorgestellten Techniken zur Diagrammexploration allesamt sehr intuitive und nützliche Funktionen, um ein bereits erstelltes Modell besser verstehen bzw. untersuchen zu können. Als Frage stellt sich am Ende heraus, ob für die Verwendung bei Aufgabenmodellen alle vorgestellten Operationen benötigt werden, wenn Aufgabenmodelle standardmäßig in der definierten Struktur eines Baumes dargestellt sind. Die genaue Verwendung wird in Kapitel 3.3 Thema der weiteren Diskussion sein.

3.2.2 Tangible Interaktion für Objekte auf Tabletops

Während sich im vorangegangenen Kapitel weitestgehend mit Node-Link Diagrammen und dafür entwickelte Multi-Touch Gesten, Visualisierungs- und Explorationstechniken beschäftigt wurde, soll es im folgenden Abschnitt um eine Verlagerung vom Natural UI zu einem Tangible UI gehen. Node-Link Diagramme eigenen sich durchaus dazu, mit realen Objekten erstellt oder bearbeitet zu werden, und es haben sich bereits einige Forschende aus diesem Bereich mit der Thematik beschäftigt. Besonders Kinder sind beliebte Testpersonen, weil sie in jungem Alter bisher wenig mit Technik in Berührung gekommen sind. Meist nehmen sie die Technik dennoch problemlos an, weil sie mit Dingen, die sie in die Hand nehmen können, sicht- bzw. auch hörbare Ergebnisse auf einem interaktiven Display erzielen können – eine Feststellung, der für die intuitive Benutzbarkeit von Tangibles spricht. Aus diesem Grund befassen sich einige der auf diesem Forschungsgebiet gemachten Veröffentlichungen vorwiegend mit Kindern. Die für Aufgabenmodelle tendenziell am geeignetsten erscheinenden Techniken werden nun im Detail vorgestellt. Dabei geht es primär um die Verwendung der Tangibles als eine Art Stempel, um Objekte schnell und zielsicher auf der Arbeitsfläche zu erzeugen und zu platzieren, oder um Funktionen mit ihnen auszuführen.

3.2.2.1 Jabberstamp, ToyVision

Zu Beginn sollte aus dem Gebiet der Tangibles einmal *Jabberstamp* genannt werden. RAFFLE ET AL. vom MIT Media Lab verfolgten mit Jabberstamp (von engl. *to jabber* = plappern und *stamp* = Stempel) die Idee, Kindergartenkindern mit Hilfe eines „Soundstempels“ die Möglichkeit zu geben, ihre eigene Stimme auf ihren gemalten Bildern zu platzieren. Auf diese Weise sollen durch die Kinder ganze Geschichten zu einem gemalten Bild erzählt werden können bzw. Kommentare zu bestimmten Aspekten oder mit dem Bild assoziierte Geräusche auditiv abrufbar werden – und das noch bevor die Kinder überhaupt zu Schreiben gelernt haben (Raffle et al. 2007).

Abbildung 3.36 - Jabberstamp[32]

[32] Quelle: (Raffle et al. 2007), S. 1

Der Vorgang gestaltet sich relativ einfach: Das Kind platziert ein normales Blatt Papier auf einem speziellen Tablett (des Herstellers WACOM) und initialisiert die Seite durch Drücken der Ziffern 1, 2, oder 3, die oben links auf dem Papier aufgedruckt sind. Nun kann es ganz normal ein Bild mit Stiften etc. malen und dann einen Gummistempel (den *Jabberstamp*) dazu verwenden, seine Stimme an einer bestimmten Stelle auf dem Papier aufzunehmen und diese Stelle entsprechend zu markieren (stempeln). Dazu ist in diesem Stempel ein Wacom Stift eingelassen sowie ein Mikrofon, dass die gesprochenen Worte oder Laute aufnimmt. Während der Stempel aufgedrückt wird, leuchtet eine rote LED zur Indikation der laufenden Aufnahme. Nach dem Wegnehmen des Stempels ist die Aufnahme abgeschlossen.

Die so gewonnen Daten werden einer auf dem angeschlossenen Computer laufenden Java-Applikation übergeben, die diese speichert und ihnen Positionen auf der Papierseite zuordnet. Es können viele verschiedene Aufnahmen pro Blatt gestempelt werden, so dass die Intention der Kinder in das Bild eingebettet übermittelt werden kann. Abgerufen werden können die so gemachten Sounddateien, indem eine Trompete, in der ein Wacom Löschstift integriert ist, das Abspielen der mit der Position verknüpften Datei über einen Lautsprecher initiiert. Diese Technik ist der erste Schritt in die Richtung, Skizzen durch Tangibles mit digitalen Informationen aufzuwerten; sie eignet sich aber aufgrund ihrer Beschränktheit rein auf Audio nicht direkt für die Entwicklung von Aufgabenmodellen.

Den besseren Ansatz dafür liefern JAVIER MARCO ET AL. mit ihrem Projekt *TOYVision.* Sie haben ein Toolkit zur einfachen Entwicklung von Videospiel-Prototypen auf Tabletops realisiert, dass frei zur Verfügung steht. Auf Basis dieses Toolkits stellen sie ein interaktives Farmspiel für Kindergartenkinder vor, dass rein über Spielzeugfiguren in Form von Tieren gesteuert werden kann, die auf einem mit Sensorik ausgestattetem Tisch von den Kindern bewegt werden (Marco et al. 2009). Dabei war die Prämisse, die Kosten gering zu halten, weshalb die Anzeige nicht in den Tisch integriert wurde, sondern auf einen externen Bildschirm ausgelagert wird (vgl. Abbildung 3.37a).

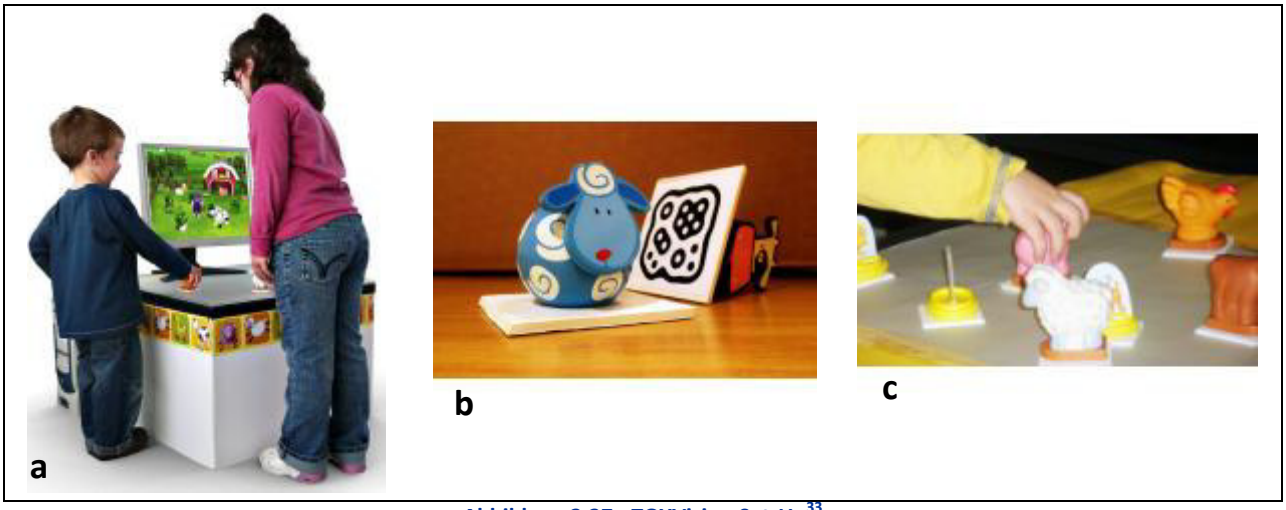


Abbildung 3.37 - TOYVision Set-Up[33]

Die Spielzeuge wurden auf einer Platte angebracht, auf deren Unterseite Fiducials zur Erkennung durch den Tisch aufgebracht wurden (b). Die Erkennung erfolgt über eine simple USB Kamera, die sich im inneren des Tisches befindet. Mit Hilfe verschiedener Fiducials kann so eine breite Anzahl an Spielfiguren in den Prozess integriert werden, mit denen später im Spiel unterschiedliche Aufgaben erfüllt werden können (c).

[33] Quelle: (Marco et al. 2009), S. 106 {a,b}; S. 108 {c}

Für den Spielbetrieb sind verschiedene Bewegungsmöglichkeiten etabliert worden, mit denen die Kinder die Tiere auf dem Bauernhof manipulieren können (s. Abbildung 3.38). Diese Schlüsselbewegungen wurden durch die genannten Autoren beim Beobachten der Kinder während des Spielens mit herkömmlichen Spielzeugen wahrgenommen und als Gesten implementiert. So lassen sich bspw. Objekte auf der Tischplatte verschieben (a) oder durch drehen ihre Orientierung verändern (b), was sich direkt auf die Tiere im Spiel übertragen lässt. Bei diesen Bewegungen muss allerdings sichergestellt bleiben, dass die Objekte mit ihren Fiducials während der Bewegung den Kontakt zur Oberfläche nicht verlieren.

Abbildung 3.38 - TOYVision: Gestik für die Benutzung[34]

Viele der Kinder vollführten eine Art Hüpfen mit den Spielzeugen, was die Anwendung „unter“ dem Tisch zunächst nicht erkennen konnte, weil das Objekt kurzzeitig weggenommen wurde und somit eine Erfassung über die Kamera nicht mehr gegeben war. Nach zurückstellen wurde das Spielzeug wieder erkannt, sodass MARCO ET AL. dies ausnutzten, um den sogenannten „Toy Click“ (analog zum Mausklick) zu etablieren (c). Die Software erkennt, wenn das Spielzeug für einen kurzen Moment entfernt und wieder platziert wird, und interpretiert dies somit als traditionelles Klicken.

Solange noch freier Platz auf dem Tisch ist, ist der Anzahl an Spielzeugen mit Fiducials grundsätzlich keine Grenze gesetzt. Auf diese Weise können durch ein Kind zwei Spielfiguren gleichzeitig bedient werden, oder was noch viel wichtiger ist, dass gemeinsames Spielen ermöglicht und somit auch das Sozialverhalten der Kinder gestärkt wird. Weiterhin lassen sich auch gleichzeitig verschiedene Operationen auf der Oberfläche ausführen (s. Abbildung 3.39).

Abbildung 3.39 - TOYVision: Multi-Tangible[35]

Den Kindern wurde als Tutor ein digitaler Farmer zur Seite gestellt. Dieser ist das einzige virtuelle Objekt, das keinen Pendant aus der realen Welt besitzt; alle anderen Objekte innerhalb der 3D Welt haben Repräsentanten unter den Spielzeugen (vgl. Abbildung 3.40a und b). Der Farmer gibt den Kindern Anleitungen, was für Aufgaben auf der Farm durchzuführen sind. So müssen die Tiere beispielsweise zu ihrer Futterstelle manövriert werden (b) um zu fressen. Dazu werden Hotspots auf der Fläche angezeigt, zu denen die Tiere auf dem Tisch geschoben werden müssen. Spezielle Szenarios lassen beispielsweise die Kinder durch vollführen des Toy Click auf einem Hotspot-Bereich mit dem Huhn virtuell ein Ei in das

[34] Quelle: (Marco et al. 2009), S. 107 {a,b,c}
[35] Ebd.

Nest auf dem Bildschirm legen (c). Nachdem die vom Farmer erfragte Anzahl an Eiern gelegt wurde, ist die Aufgabe beendet. Analog dazu kann z. B. die Kuh auf einem Eimer Milch geben.

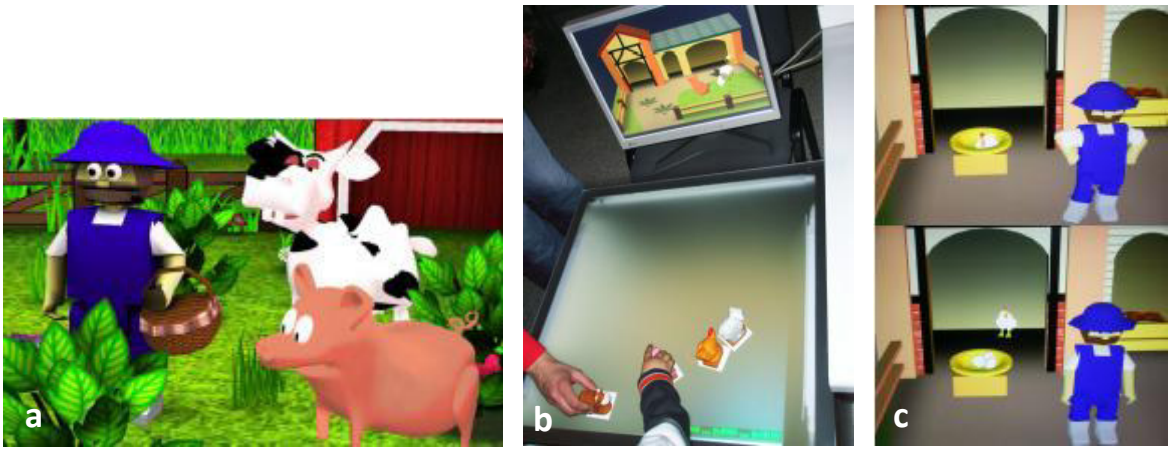


Abbildung 3.40 - Interactive Farm Game[36]

In einer aktualisierten Veröffentlichung wurde das Toolkit noch einmal verbessert. Mit den gemachten Erweiterungen ist man nun in der Lage, zwischen *Simple Tokens*, *Named Tokens*, *Constraint Tokens* und *Deformable Tokens* zu unterscheiden (Marco et al. 2012). Simple Tokens sind dabei mit einfacheren Markern ausgestattet als die typischen Fiducials (bspw. einfache Kreise) und Named Tokens dienen zur Benennung und Identifizierung von Objekten. Als Constraint Token wird von MARCO ET AL. ein physisch manipulierbares Objekt bezeichnet, das aus Simple Tokens und einem Named Token besteht. Durch Hinzufügen oder Entfernen der Simple Tokens zum/aus dem Kontext kann die Bedeutung des Constraint Token verändert werden und darauf innerhalb der Software reagiert werden (Abbildung 3.41 oben). Zu guter Letzt stellen sie Deformable Tokens vor, deren veränderbare Form erkannt werden kann (Abbildung 3.41 unten).

Abbildung 3.41 - TOYVision: Constraint Tokens (oben) und Deformable Tokens (unten)[37]

Obwohl das vorgestellte Toolkit vorwiegend für Spiele gedacht und die Zielgruppe eher auf Kinder ausgerichtet ist, bieten sich durch die Vorschläge von MARCO ET AL. interessante Aspekte an, die für die Erstellung und Bearbeitung von Diagrammen geeignet sind. Die Verwendung von Figuren oder Stempeln ist eine vielversprechende Technik, mit der sich grundsätzlich sowohl Knoten als auch Kanten erzeugen

[36] Quelle: (Marco et al. 2009), S. 108 {a}; S. 109 {b,c}
[37] Quelle: (Marco et al. 2012), S. 75

lassen. Im nächsten Abschnitt folgt nun eine erweiterte Methode, die einen dynamischen Stempel realisiert.

3.2.2.2 Mobiles Endgerät als dynamischer Stempel und Kontrolleinheit

Die zuvor in diesem Abschnitt beschriebenen Konzepte aus dem Bereich der Tangibles beschäftigen sich primär mit dem Ziel, Kindern moderne Technik spielerisch näher zu bringen und durch anfassbare Objekte erste Funktionen und Aufgaben auf einfache Weise innerhalb einer Software zu bewältigen. Dass dies auch brauchbare Ansätze für Aufgabenmodelle liefert steht außer Frage und wird in Kapitel 3.3 noch weiter vertieft; jedoch lässt sich der Bezug bei anderen Veröffentlichungen wesentlich deutlicher erkennen.

Nachfolgend soll nun eine Technik vorgestellt werden, die die Verbindung von mobilen Endgeräten und einem Tabletop herstellt und die Erzeugung virtueller Objekte auf eine Art ermöglicht, die optimal für Aufgabenmodelle geschaffen zu sein scheint. MARKUS OTT ET AL. beschäftigten sich mit einer Methode, privaten Besitz auf einer öffentlich zugänglichen Arbeitsfläche zu etablieren und mit Hilfe eines mobilen Endgerätes neue Objekte auf dieser Fläche anzulegen oder bestehende zu manipulieren (Ott et al. 2012). Für diesen Zweck haben Sie ein Raumplanungswerkzeug entwickelt, mit der verschiedene Personen gemeinsam einen Raum einrichten und dabei Tablets zu Hilfe nehmen können. Jeder Anwender arbeitet in seinem eigenen sog. *Kontext,* welche durch farblich umrandete Boxen dargestellt werden, die den Raumplan beinhalten. Schritt für Schritt zeigt Abbildung 3.42 beispielhaft den Vorgang des Erstellens neuer Objekte in jenem Raum.

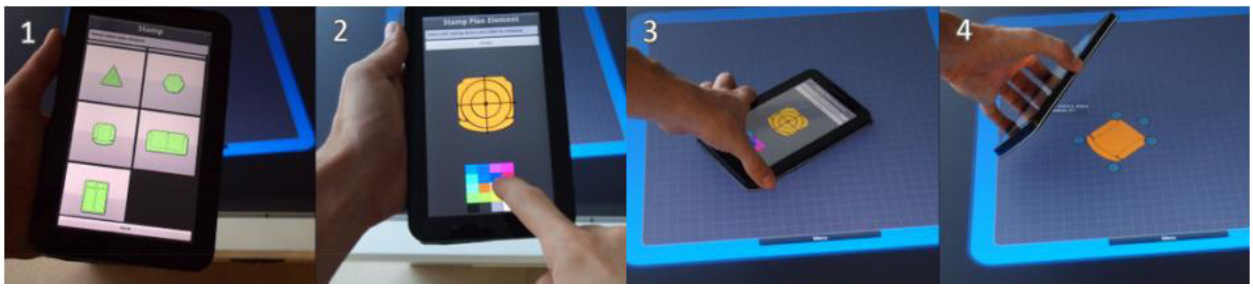


Abbildung 3.42 - Erzeugen neuer Objekte durch Definieren auf einem Tablet und anschließendem Stempeln[38]

Zunächst wird die Grundform des Objektes durch den Anwender auf seinem Gerät ausgewählt (1). Als folgender Schritt kann die Farbe des einzufügenden Objektes bestimmt werden (2). Nachdem das mobile Endgerät in gewünschter Ausrichtung auf dem Tisch platziert wurde (3), erscheint das erzeugte Objekt nach dem Hochheben auf der Arbeitsfläche (4). Dazu wurde im Vorfeld vom Benutzer ein eigener Kontextbereich auf der Arbeitsfläche etabliert, auf dem neue UI Elemente platziert werden können und der auch mit anderen Anwendern geteilt werden kann. Dieser Prozess bietet bereits jetzt großes Potential zur Erstellung neuer Knoten und Kanten in Aufgabenmodellen und greift die in Kapitel 3.2.1.2 diskutierte Objektpalette (digitale Schablone) wieder auf.

Aber die Urheber der Technik haben sich weitere Gedanken zu dieser Thematik gemacht und mehr Funktionen in ihr Konzept integriert. So wird das mobile Endgerät zum Editierungswerkzeug für erzeugte Objekte, indem es auf das gewünschte Objekt gelegt wird (s. Abbildung 3.43). Nachdem das Tablet platziert wurde, erfolgt eine Fokussierung auf das Objekt (2), welches der Benutzer bestätigen muss (3). Im Anschluss werden verfügbare Operationen für das gewählte Objekt auf dem Bildschirm des Tablets angezeigt und können aufgerufen werden (4).

[38] Quelle: (Ott et al. 2012), S. 2

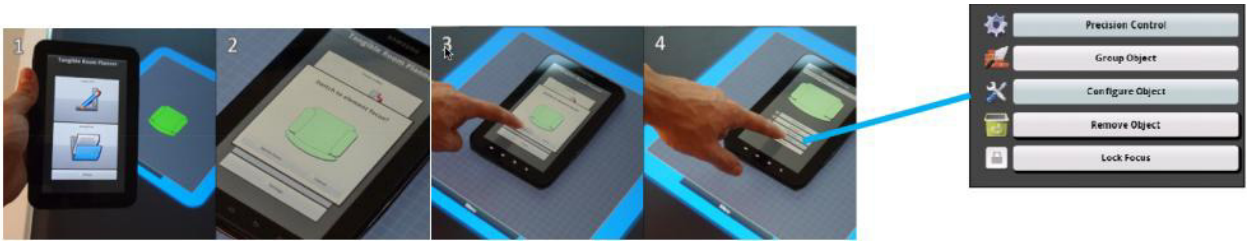


Abbildung 3.43 - Bearbeiten von Objekten mit Hilfe des Tablets[39]

Interessant ist hierbei zu erwähnen, dass bereits hier an Besitz gedacht wurde. Weil der Tisch öffentlich zugänglich ist, könnten theoretisch alle Objekte auf dem Bildschirm durch die Anwesenden bearbeitet werden. Die Bearbeitung des Objekts durch andere ist aber insofern eingeschränkt, dass sie mindestens auch ein angeschlossenes Tablet besitzen müssen. Dieses muss darüber hinaus aber auch noch mit dem Kontext gekoppelt sein, um alle Funktionen benutzen zu können. Das Auslagern der privaten Funktionen auf das mobile Endgerät kann somit beispielsweise unbefugtes Verändern außerhalb der Intention des Erstellers restringieren. Genauso können an das Objekt angehängte Daten public oder private sein, je nach Anforderung des Erstellenden.

Weiterhin lassen sich durch den Anwender neue Kontexte mit dem Tablet erstellen, die in diesem Fall weitere Räume zur Planung darstellen (Abbildung 3.44). Dazu wählt man zunächst die entsprechende Funktion aus (1). Nachdem Einstellungen zu dem neuen Kontext getroffen wurden (2), erhält der Anwender den Hinweis, dass Gerät auf den Tisch zu legen (3). An platzierter Stelle wird unmittelbar ein neuer Kontext erzeugt (4), erkennbar durch die blaue Umrahmung. Innerhalb dieses Kontextes entsteht nun die Planung eines weiteren Raumes. Kontexte lassen sich mit anderen teilen (*share*) oder ihre Inhalte miteinander verschmelzen (*merge*).

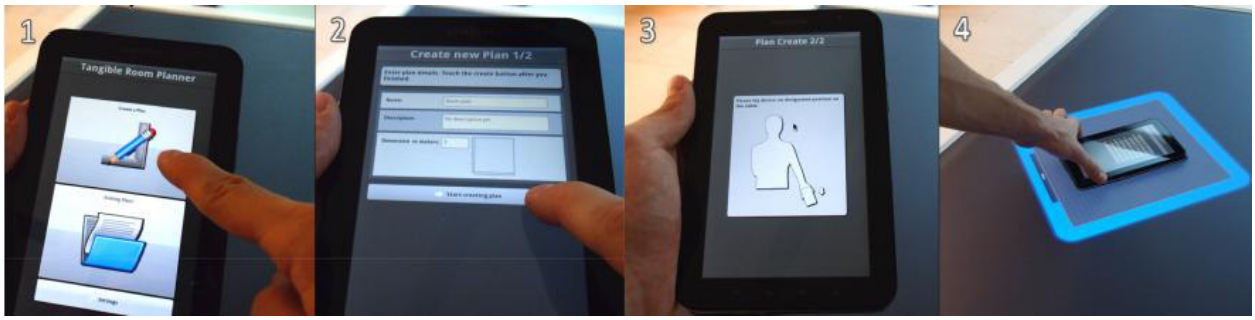


Abbildung 3.44 - Erzeugung neuer Kontexte[40]

Reicht für die Bearbeitung eines UI Elementes die Auflösung des MTTs nicht aus, weil Eingaben nicht präzise genug möglich sind, hilft auch an dieser Stelle das Mobilgerät weiter. Nachdem ein Objekt mit dem Tablet fokussiert wurde, lassen sich Präzisionseinstellungen auf dessen besser aufgelösten Touch Screen vornehmen. Dieser Modus hat zwei Zustände, die durch Drücken des Mittelpunkts des Kreises umgeschaltet werden können (1): Einmal die Genauigkeitseinstellung (Accuracy) und einmal die Objektmanipulation.

[39] Quelle: (Ott et al. 2012), S. 5
[40] Quelle: (Ott et al. 2012), S. 6

Abbildung 3.45 - Präzisionsbearbeitung von Objekteigenschaften[41]

Im Accuracy Modus kann durch ausführen einer Pinchgeste auf dem grünen Kreis die Präzision der objektmanipulierenden Funktionen eingestellt werden (control-display gain, CDG). Zur Orientierung des CDG dient der innere Kreis, der sich entsprechend der Weite der gespreizten Finger anpasst (2). Nachdem die Genauigkeit eingestellt wurde, können die üblichen Gesten zum Verschieben (3), Skalieren (4) (5) und Rotieren (6) benutzt werden, um das Objekt exakt passend zu verändern.

Alle in diesem Abschnitt beschriebenen Aspekte sind sehr interessant für Zwecke der Diagrammbearbeitung und besonders die gemachten Überlegungen zu Besitz in öffentlich zugänglichem Raum sind hilfreich für die kollaborative Arbeit. Mögliche Anwendungsszenarien für das Konzept werden in Kapitel 3.3 vertiefend diskutiert. Im weiteren Verlauf werden nun existierende Kollaborations-techniken vorgestellt und bewertet.

3.2.3 Kollaborationstechniken für interaktive Displays

Ein weiterer wichtiger Aspekt bei großen interaktiven Displays ist wie schon genannt das gemeinsame Erstellen und Bearbeiten von Projekten auf einer für alle Mitwirkenden erreichbaren Oberfläche. Obwohl sich bereits viele Forschungsarbeiten um eine distanzübergreifende Kollaboration mit verteilten Tabletops bemühen (Tuddenham & Robinson 2009) (Ardaiz et al. 2010) (Yamashita et al. 2011), soll es in diesem Abschnitt um Techniken gehen, die sich auf lediglich einen Tabletop mit mehreren Benutzern beschränken, weil das in Kapitel 4 folgende Konzept sich auch nur auf ein einzelnes interaktives Display bezieht.

Dabei soll es zunächst noch einmal um die von MARKUS OTT ET AL. eingeführten Kontexte und ihre Verwendung im kollaborativen Umfeld gehen, bevor daran anschließend weitere Überlegungen von STACEY SCOTT und TSE ET AL. zu Territoriums-basierten Ansätzen für die Interaktion bei der Zusammenarbeit auf Tabletops diskutiert werden.

3.2.3.1 Kollaboration durch geteilte Kontexte

Die von OTT ET AL. eingeführten *Tangible Contexts* wurden bereits im vorherigen Kapitel näher vorgestellt (Ott et al. 2012). Sie ermöglichen private Bereiche und Besitz auf einer öffentlich zugänglichen Oberfläche mit Hilfe mobiler Endgeräte. Ein Tangible Context besteht aus UI Elementen auf dem Tabletop und einem UI auf den mobilen Geräten. Im Sinne der Kollaboration ist interessant, dass jeder Kontext auf dem Tisch auch mehreren Benutzern zugeordnet sein kann, anstatt lediglich einem. Jedes UI Element auf dem Tisch ist mindestens einem Kontext zugewiesen und verkörpert das öffentlich zugängliche, mit dem alle Benutzer interagieren können. Abbildung 3.46 zeigt so ein mögliches Szenario.

[41] Quelle: (Ott et al. 2012), S. 6

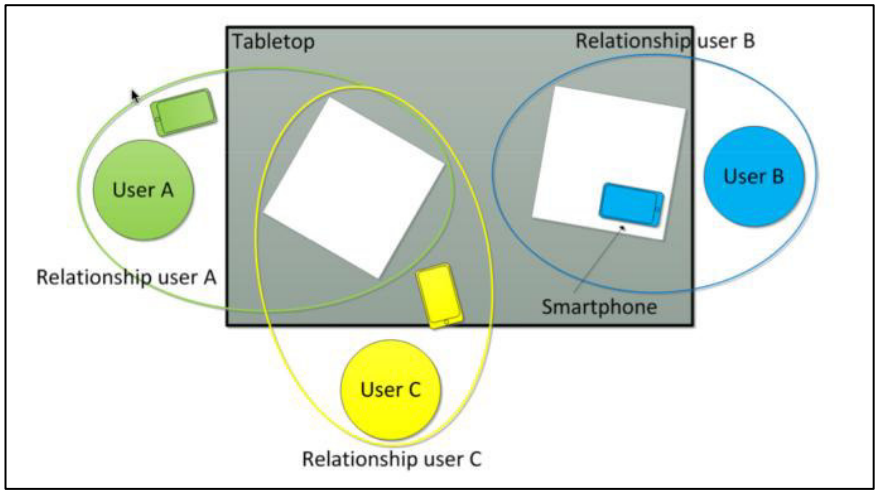


Abbildung 3.46 - Shared Contexts auf dem Tabletop[42]

User A und User C teilen sich in ihrem Public Context ein UI Element (repräsentiert durch das weiße Rechteck) auf dem Tabletop, wobei sie weiterhin gleichzeitig noch ihre privaten Kontexte auf dem Mobilgerät besitzen, mit dem sie interagieren. User B arbeitet in diesem Fall allein an einem UI Element. Durch auflegen des Smartphones oder Tablets auf UI Elemente können Teilnehmer ihre privaten User Interfaces mit dem Interface des Tisches verknüpfen, um so simultan (aber unabhängig voneinander) in ein und demselben Kontext zu agieren – und die Ergebnisse anschließend zu vergleichen und ggf. zu vereinigen (s. übernächster Absatz).

In ihrer Veröffentlichung haben die zitierten Autoren versucht, ihr Konzept durch sogenannte *Context Hulls*, also Hüllen zu illustrieren, die einem Kontext zugewiesen sind und entsprechend UI Elemente aufnehmen. Bezüglich der Raumplanungsanwendung nehmen diese Hüllen als UI Elemente Räume auf, in denen sich Möbel befinden. Alle Elemente, die dem öffentlichen Kontext angehören, sind visuell in farblich gekennzeichneten, konvexen Hüllen arrangiert (s. Abbildung 3.47). Die Hintergrundfarbe der Hülle ändert sich je nach Kontext, zu dem sie gehört. User A bearbeitet in seiner Hülle zwei Räume, während sich User B in seiner Hülle mit einem anderen Raum beschäftigt. In beiden Hüllen können weitere Benutzer als Teilhaber integriert werden.

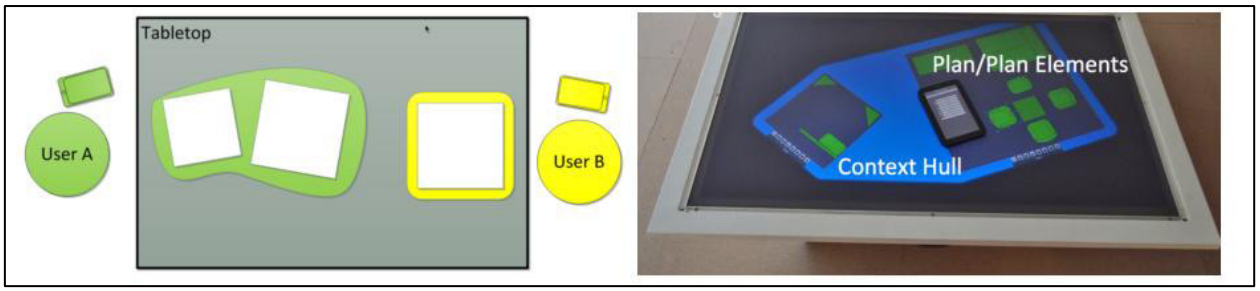


Abbildung 3.47 - Tangible Contexts und Context Hulls[43]

Als weitere Funktion geben OTT ET AL. mit *Merge Control* die Möglichkeit vor, verschieden Zustände eines UI Elements zu vergleichen und zu vereinen. Auf diese Weise können verschiedene Benutzer im privaten Kontext ihres Tablets an demselben Raumplan arbeiten, während sich die Ergebnisse nach abgeschlossener Arbeit aller auf dem öffentlichen Kontext des MTTs zusammenführen lassen. Abbildung 3.48 zeigt beispielhaft diesen Vorgang.

[42] Quelle: (Ott et al. 2012), S. 4
[43] Quelle: (Ott et al. 2012), S. 5

Abbildung 3.48 - Zusammenlegen von Kontexten[44]

Vorerst ist nur der aktuelle Raumplan auf dem Tabletop sichtbar (1). Sobald der Anwender auf dem Tablet in der Merge Control die Anzeige existierender Pläne auswählt (2), werden diese in einer Liste angezeigt (3). Nach dem Selektieren der Räume auf dem Tablet (4) zeigt der Tabletop die beiden Varianten vergleichend an (durch Overlay und Farbliche Trennung) (5), bevor der Benutzer am Ende den Vereinigungsvorgang abschließt (6).

Die Bearbeitung solcher Overlays gestaltet sich auf interaktiven Oberflächen meist kompliziert, was in ihrer zweidimensionalen Natur begründet liegt. Aus diesem Grund haben OTT ET AL. das Tablet dazu benutzt, auch diese Funktion aufzunehmen (vgl. Abbildung 3.49). Nachdem der Anwender den Raumplan mit dem Tablet fokussiert hat (1), kann durch Drehen des Gerätes über dessen Lagesensor automatisch die *Layer Control* aufgerufen werden (2).

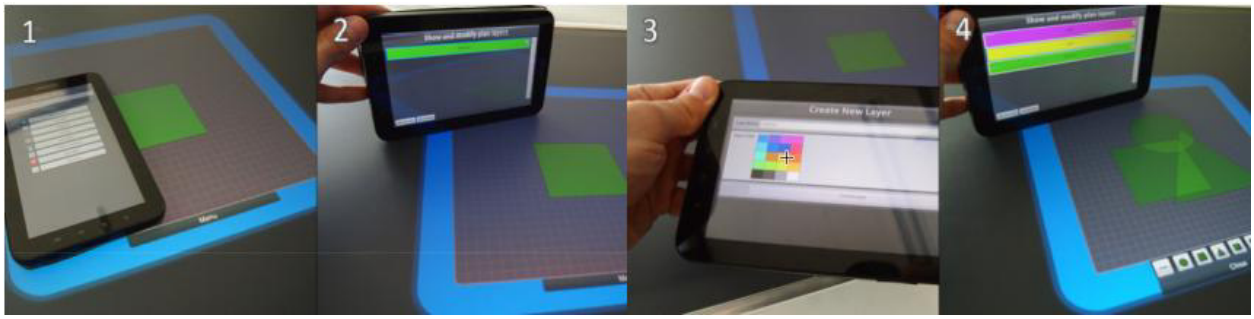


Abbildung 3.49 - Bearbeitung von Overlays durch Kippen des Tablets[45]

Innerhalb dieser Kontrolle kann man nun weitere Schichten auf dem Raumplan hinzufügen und farblich voneinander abgrenzen (3). Diese können als halbtransparente Objekte angesehen werden, deren Rangfolge auf dem Tablet angepasst werden kann (4).

3.2.3.2 Kollaboration durch Aufteilung des Arbeitsbereichs (Territorien)

Im vorangegangen Abschnitt wurde viel Wert auf Privatsphäre und die Verwendung von mobilen Endgeräten als Tangibles gelegt. Durch Verwendung von Smartphones oder Tablets wird aber auch der Erstellungsprozess eher vom Tisch ausgelagert und findet zunehmend isolierter statt. STACEY SCOTT hat sich bereits Jahre zuvor mit der Einrichtung von Territorien auf der Oberfläche des Tisches beschäftigt, um Bereiche für spezielle Zwecke für die Benutzer auszuweisen (Scott 2003). Die Grundidee der Aufteilung der zur Verfügung stehenden Arbeitsfläche harmoniert tendenziell besser mit der Erstellung von Aufgabenmodellen.

In den gemachten Beobachtungen wurde festgestellt, dass die an der Studie teilnehmenden Personen nahezu automatisch ihren Arbeitsbereich einteilen. Diese Partitionierung findet in drei eindeutige Typen von Territorien statt und wurde von Scott durch die Bezeichnungen *personal, group* und *storage* geprägt (basierend auf der jeweiligen Rolle, die sie während der Kollaboration zu spielen schienen). Die Probanden benutzten persönliche Bereiche als Vorbereitung von individueller Arbeit, die oftmals später

[44] Quelle: (Ott et al. 2012), S. 7
[45] Ebd.

in die Gruppenarbeit integriert wurde. Dazu wurde dann das Gruppenterritorium benutzt, um am Produkt der Gruppe zu arbeiten, während Lagerbereiche für das temporäre Ablegen momentan nicht benötigter Objekte gebraucht wurden. Das Setting eines solchen Szenarios zeigt Abbildung 3.50.

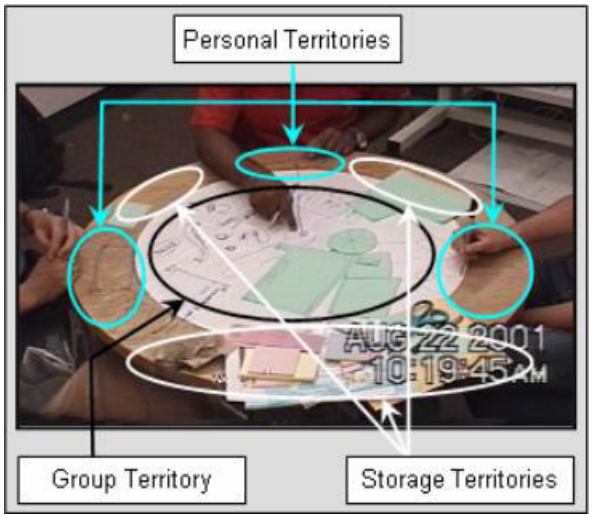


Abbildung 3.50 - Territorien bei einer realen Tabletop-Kollaboration[46]

Die Positionen der Territorien ergaben sich dabei bereits aus den Standpunkten der Teilnehmer am Tisch. Persönliche Bereiche wurden direkt vor den Betreffenden eingerichtet, während der Gruppenbereich in der Mitte eingerichtet wurde, um von allen gut erreichbar zu sein. Die übriggebliebenen Zwischen- und Randbereiche dienten als Ablage. Diese Erkenntnisse halfen, die Territorien von der analogen Welt auf den digitalen Anwendungsfall zu übertragen.

Durch die intensive Observation der Probanden wurden die folgenden Systemanforderungen identifiziert, die essentiell für die Unterstützung von natürlichem, menschlichem Verhalten in Bezug auf Territorien sind.

1. Einfache Anpassung der Größe von Territorien
2. Einfache Anpassung der Orientierung verschiedener Territorien
3. Einfache Anpassung der Elementausrichtung, unabhängig von der Position und
4. Einfaches Übersteuern von system-unterstützten Aktionen.

Als Ergebnis der Überlegungen wurde die Entwicklung einer Prüfstand-Umgebung (test-bed environment) initiiert, die nach dem Schema in Abbildung 3.51 arbeitet. Hierbei wird die Entstehung der Partitionen deutlich.

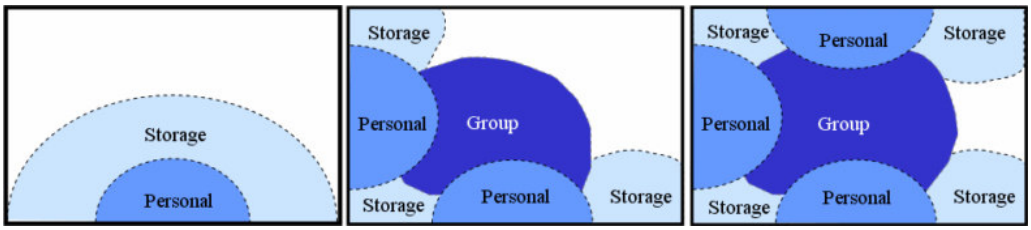


Abbildung 3.51 - Anwendung der gemachten Beobachtungen im digitalen Ansatz[47]

Zunächst gibt es im Single-User Fall nur einen persönlichen Bereich mit einem Ablagebereich, der außen um den persönlichen Bereich herum angeordnet ist (Abbildung 3.51, links). Das Gruppenterritorium

[46] Quelle: (Scott 2003), S. 2
[47] Quelle: (Scott 2003), S. 3

wird in diesem Fall nicht benötigt. Kommt ein Benutzer hinzu, wird in der Mitte der Arbeitsfläche für alle gut erreichbar der Gruppenbereich etabliert (Abbildung 3.51, Mitte). Persönliche Bereiche bleiben erhalten bzw. werden für die neuen Benutzer angelegt, die Ablagebereiche aufgeteilt und in den verbleibenden Raum verlagert. Dabei ist es unvermeidbar, dass sich Anwender die Ablagen teilen müssen. Je mehr Benutzer hinzukommen, desto deutlicher wird die Belegung der Territorien (Abbildung 3.51, rechts). Außerdem wird deutlich, dass die Belegung mit mehr als vier Benutzern sich höchstwahrscheinlich als sehr schwierig gestaltet.

Auch haben TSE ET AL. für kooperatives Arbeiten den multimodalen Einsatz einer aufgesplitteten Ansicht (*Split View*) über existierende Anwendungen näher untersucht (Tse et al. 2007). Dazu wurden drei mögliche Szenarien vorgestellt, die bei einem *Split Screen* in Frage kommen (vgl. Abbildung 3.52). Zunächst einmal ist die Benutzung voneinander unabhängiger Applikationen naheliegend (links), während im Sinne eines geteilten Bildschirms (Mitte) auch gemeinsames Betrachten und Bearbeiten derselben Anwendung denkbar wäre. Für Aufgabenmodelle ist sicherlich der Einsatz von echter Groupware (rechts) tendenziell am besten geeignet, weil damit zusammenhängende Applikationen realisiert werden können, die untereinander verknüpft sind und sich so gegenseitig beeinflussen.

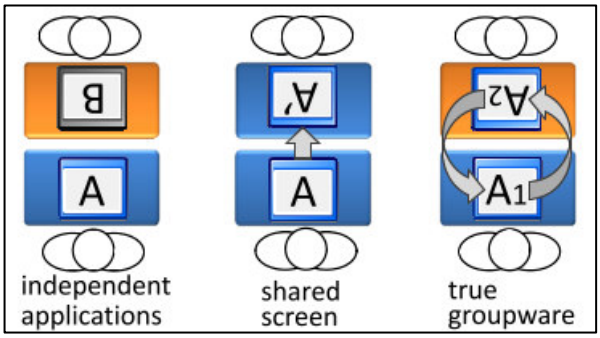


Abbildung 3.52 - Softwarekonfigurationen für zwei gegenübersitzende Personen auf einem Split View Tabletop[48]

Weiterhin wurden mögliche Sitzarrangements für die Zusammenarbeit zweier Personen identifiziert. Die möglichen Anordnungen für diesen Personenkreis wurden mit *Face-to-Face*, *Side-by-Side* und *Catty Corner* identifiziert (s. Abbildung 3.53). Eine Kombination aus Face-to-Face und Side-By-Side bietet sich für die Anordnung von vier Personen an, während Catty Corner ideal für drei Personen eingesetzt werden kann.

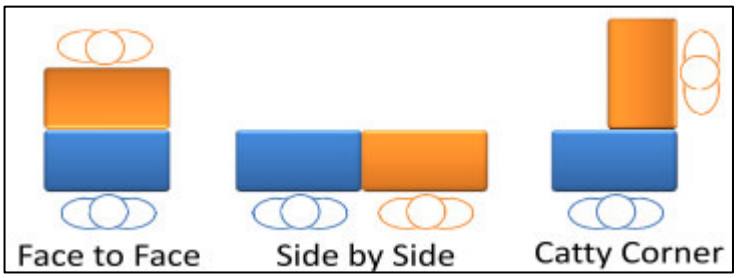


Abbildung 3.53 - Sitzarrangements bei zwei Personen[49]

Schlussendlich soll das Kapitel 3 nun mit einem Zwischenfazit abgeschlossen werden, welches die gesamten Gedanken noch einmal reflektiert und die analysierten Techniken kritisch für die Verwendung im Konzept bewertet.

[48] Quelle: (Tse et al. 2007), S. 129
[49] Quelle: (Tse et al. 2007), S. 131

3.3 Zwischenfazit

Nachdem verschiedene Werkzeuge zur Aufgabenmodellierung und unterschiedliche Techniken der Interaktionen auf MTTs untersucht wurden, soll nun eine Bilanz zu den gewonnenen Erkenntnissen aufgestellt werden.

Aus dem Gebiet der Aufgabenmodellierung scheint das Tool CTTE am geeignetsten, um als Adaption auf dem interaktiven Display realisiert zu werden. Der reiche Funktionsumfang sowie die geeignete Darstellung erweisen sich als die beste der untersuchten Anwendungen – nicht umsonst hat sich dieses Tool als einziges bis in die Industrie etabliert. CTTE wirkt insgesamt vollständig, kann aber an der einen oder anderen Stelle durchaus sinnvoll durch einzelne Elemente der anderen Umgebungen VTMB, K-MADe und IdealXML ergänzt werden. Insbesondere der mehrfach erwähnte kooperative Modus ist höchst interessant für die gemeinsame Entwicklung an einem Bildschirm.

Die gefundenen Nachteile bei CTTE, die vorwiegend diagrammtechnischer Natur sind, lassen sich bei der richtigen Konzeption auf einem interaktiven Display sicherlich vermeiden. Die Probleme sind vielleicht alle auf die Programmiersprache Java zurückführen, in der die Anwendung geschrieben ist und die sich manchmal nur bedingt für graphische Programmierung eignet. Wahrscheinlich wurde sie aus Gründen der Plattformunabhängigkeit und Kompatibilität von den Autoren ausgewählt.

Im Abschnitt Node-Link Diagramme wurden durchgehend verwendbare Ideen vorgestellt, die einerseits auf die Entwicklung solcher Diagramme abzielen, andererseits auf deren Darstellung oder Exploration. Besonders das vorgestellte Gesten-Set eignet sich hervorragend für das Konzept, weil es für die Erstellung verschiedenster Diagrammtypen universell einsetzbar ist. Diese Universalität macht das Set sehr attraktiv, und der geringe Umfang des Sets lässt noch Raum für die Entwicklung etwaiger spezieller Gesten rein für die Aufgabenmodellierung zu.

Die vorgestellten Techniken, die sich rein auf die Entwicklung auf speziellem Papier beziehen, bietet für eine Integration in die folgenden Ideen nur bedingt Eignung. Da es in dieser Arbeit vorwiegend um ein interaktives Display gehen soll, ist die Verwendung der vorgestellten Sketchbooks für diese Arbeit tendenziell eher obsolet. Jedoch bieten die zitierten Autoren auch ihren Ansatz für einen MTT an, und positiv fällt die Verwendung der gezeigten Paletten-Kärtchen auf, die wiederum erneut rein papierbasiert präsentiert wurden. Argumentiert wurde mit einer individuell angepassten Palette, die jeder Anwender einfach mitführen und selbst gestalten kann.

Die Verwendung eines Smartphones als Träger einer solchen Palette erscheint in der Gegenwart jedoch zunehmender als sinnvoll, da ein solches in den meisten Fällen ebenfalls bereits durch den Anwender mitgeführt wird und mit einer App um die Funktionalität der Palette erweitert werden kann. Dies würde sogar den Vorteil mit sich bringen, dass die Palette auch nachträglich noch geändert werden könnte, ohne sie erneut auf dem speziellen Papier drucken zu müssen. Jedoch ginge dabei natürlich der gewollte Aspekt des spontanen Zeichnens verloren, und es müsste eine Verbindung der auf den Smartphones laufenden Apps mit dem MTT implementiert werden, was zusätzlichen Programmier- und Rechenaufwand zur Folge hat – ein Umstand, der zugunsten der Vorteile sicherlich vertretbar wäre.

Sehr interessant gestaltet sich auch die Technik zur Off-Screen Visualisierung von Node-Link Diagrammen. Werden Aufgabenmodelle sehr komplex – eine Eigenschaft, die auch schon bei weniger umfangreichen Softwareprogrammen sehr schnell erreicht werden kann – ist dies eine der besten Varianten, nicht im Arbeitsbereich befindliche Elemente für den Anwender sichtbar zu machen. Zwar wurde die Methode ursprünglich rein für UML Diagramme konzipiert, sie lässt sich aber mit kleinen

Änderungen sehr leicht im Sinne der Aufgabenmodellierung anpassen. Diese Anpassung wird im anschließenden Kapitel vorgenommen und als Baustein des Konzeptes vorgestellt.

Von den Explorationstechniken sind nicht alle der vorgestellten Varianten direkt brauchbar, was aber wie bereits erwähnt der speziellen Struktur der Aufgabenmodelle geschuldet ist. Dadurch, dass Bäume verwendet werden, liegt bereits eine geordnete Struktur der Kanten vor. Dieser Umstand relativiert die Verwendung von TouchPlucking, -Pinning und -Strumming sowie der PushLens für Aufgabenmodelle etwas. Dennoch kann das temporäre Verändern von Kanten für Übersicht in komplexen Bäumen sorgen, wenn Teilbäume bspw. isoliert betrachtet werden sollen. Per Pinning kann dieser Zustand vorübergehend fixiert werden (oder es wird eine entsprechend große Linse verwendet). Strumming kann mit der Erweiterung auf nachgelagerte Ebenen weiteren Aufschluss über die Tragweite einer Teilaufgabe liefern, wenn ihr Knoten gezupft wird. TouchBundeling lässt sich eventuell verwenden, um Äste mit vielen Kindern übersichtlicher darzustellen. Auf diese Techniken wird im Konzept nicht mehr explizit eingegangen, da sie sich problemlos auf Aufgabenbäumen verwenden lassen.

Betrachtet man weiterhin die vorgestellten Tangible UIs, wird die effektive Verwendung von Stempeltechniken zur Erstellung von Objekten auf einem interaktiven Display deutlich. Obwohl die Technologien zunächst vorwiegend auf Kinder abgezielt haben, kann das direkte Stempeln von Elementen auf der Arbeitsfläche den Erstellungsprozess bei Diagrammen stark beschleunigen. TOYVision bietet in dieser Hinsicht den richtigen Ansatz für statische Stempel; einzig die Trennung von Anzeige und Arbeitsfläche wären für die Bearbeitung von Aufgabenmodellen tendenziell eher ineffizient, da eine integrierte Lösung benötigt wird. Der Ansatz des dynamischen Stempels bietet weitere interessante Vorteile, bspw. dass lediglich ein Gerät (anstatt vieler) für das Stempeln aller Elemente benötigt wird und eben durch die Verwendung elektronischer Geräte weitere Steuerungs- und Bearbeitungsmöglichkeiten realisierbar erscheinen.

Die Untersuchungen zur Kollaboration auf einem Tabletop haben gezeigt, dass eine Aufteilung der Arbeitsfläche für das gemeinsame Arbeiten nahezu unausweichlich erscheint. Obwohl das Verwenden von gemeinsam nutzbaren Kontexten eine schöne Metapher darstellt und die Auslagerung privater Funktionen auf das mobile Endgerät durchaus sinnvolle und diskutable Ansätze bietet, ist die Erstellung komplexer Modelle auf einem Tablet eventuell noch denkbar; auf einem Smartphone aufgrund der geringen Displaygröße jedoch bereits mindestens grenzwertig. Zudem schmälert diese Arbeitsweise eher das Kollektiv, als das es der Zusammenarbeit dient. Darüber hinaus würde der Vorteil des MTTs, nämlich seine beachtliche Größe, dadurch wieder ansatzweise relativiert. Die Aufteilung eben dieser Größe erscheint daher als die sinnvollere Variante, aber muss auch hier auf die Art der Aufteilung Rücksicht genommen werden.

Die Einrichtung von Benutzerterritorien ist grundsätzlich der richtige Weg, für Diagramme ist die Einrichtung eines Gruppenbereichs in der Mitte der Fläche jedoch ungeeignet. Speziell die Diagrammform des Aufgabenmodells erfordert getrennt isolierte Bearbeitungsmöglichkeiten, da die Baumstruktur eine Orientierung des Diagramms voraussetzt, die durch einen Gruppenbereich in der Mitte der Arbeitsfläche zumindest für alle anderen Teilnehmer nicht gegeben wäre. Die herausgestellten Anforderungen an die Territorien haben aber weiterhin ihre Gültigkeit und werden deshalb Beachtung finden. Die Aufteilung gemäß dem multimodalen Ansatz spielt aber eine übergeordnete Rolle.

Alle bisherigen Feststellungen sind nun ausreichend, um mit der Entwicklung des Konzeptes zu beginnen.

4 Konzeption und Entwicklung

In diesem Kapitel wird nun sukzessive ein Entwurf entstehen, der die Aufgabenmodellierung auf interaktive Displays zu bringen versucht. Dabei werden alle bisher gewonnen Erkenntnisse harmonisiert und schlüssig in den Kontext eingebracht. Es werden Aspekte der Darstellung, Interaktion und Kollaboration diskutiert und versucht, deren ideale Kombination herauszuarbeiten. Als Quintessenz der aufgestellten Thesen soll sich ein zusammenhängendes und überzeugendes Konzept abzeichnen, das als unmittelbares Grundgerüst für die Entwicklung einer Aufgabenmodellierungsapplikation auf einem Multi-Touch Tabletop dienen kann. Anschließend wird in Kapitel 5 das Ergebnis in der Zusammenfassung reflektiert und ein Ausblick auf zukünftige Entwicklungen auf diesem Gebiet gegeben.

Da in Kapitel 3 CTTE als bestes verfügbares Werkzeug zur Aufgabenmodellierung identifiziert wurde, lehnen sich viele Aspekte in den folgenden Betrachtungen in Symbolik und Aufbau an CTTE an. Als Referenzmodell wird *ATM.ctt* verwendet (vgl. Abbildung 4.1), das als Beispiel bei CTTE mitgeliefert wird. Besonders die später vorgestellten Gesten beziehen sich bei ihrer Ausführung auf dieses Modell. Geht es um ein kooperatives Modell, wird *GuitarSalesOrder.cctt* verwendet, welches ebenfalls mitgeliefert wird.

4.1 Darstellung

Dieser Abschnitt befasst sich detailliert mit der visuellen Repräsentation von Aufgabenmodellen. Dabei gilt es, verwendbare Strukturen der Aufgabenmodelle zu definieren und als geeignet zu verifizieren. Weiterhin spielt die zu benutzende Symbolik innerhalb des Diagramms eine tragende Rolle, darum sollte sie intuitiv leicht zu verstehen sein, aber auch Wiedererkennungswerte bieten. Symbolik und Struktur sollten aufeinander abgestimmt sein, was im Folgenden zu realisieren versucht wird.

Auch die Gestaltung des Diagramms z. B. am Knoten oder an den Kanten spielt dabei eine Rolle. Welche Informationen wie detailliert direkt an Knoten/Kanten angezeigt werden, wird im Folgenden festgelegt. Darüber hinaus können Aufgabenmodelle mit anwachsender Komplexität sehr detailreich werden, was das vorübergehende Ausblenden von Diagrammelementen unausweichlich macht. Die Visualisierung solcher Off-Screen Elemente ist ein weiterer Teil dieser Betrachtung.

4.1.1 Struktur

In der hierarchischen Natur von Aufgabenmodellen liegt die Begründung, warum diese bisher ausschließlich als Bäume dargestellt werden. Bäume spiegeln am besten die hierarchische Struktur der Modelle wieder, da sie ausgehend von einer Wurzel ein Problem sukzessive in die Tiefe aufbauen und dabei Reihen- und Rangfolgen bereits innerhalb ihrer Struktur ableiten. Demzufolge sollte man bei der strukturellen Gestaltung der Aufgabenmodelle weiterhin an Bäumen festhalten, da es keine optimalere Form resp. Datenstruktur als diese für hierarchisch aufgebaute Modelle gibt. Hier sollen lediglich Denkanstöße für andere Sichten auf die vorhandenen Bäume gegeben werden, um vielleicht neue Perspektiven auf verschiedene Ebenen oder Pfade zu ermöglichen, an die bisher vielleicht nicht gedacht wurde.

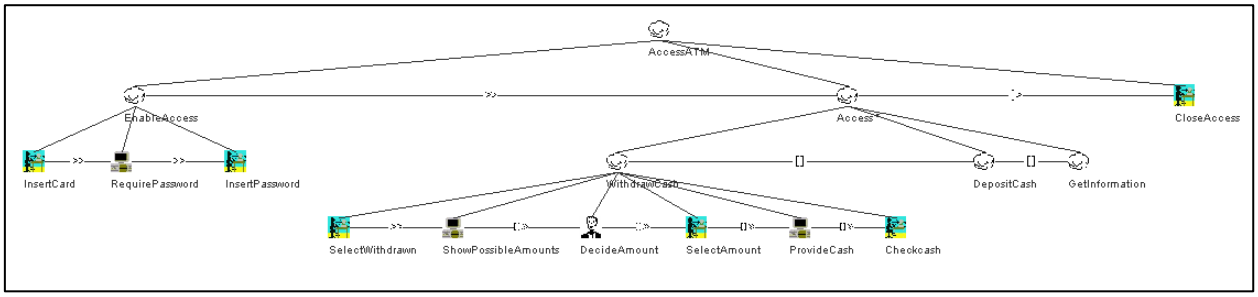


Abbildung 4.1 - Aufgabenmodell Beispielbaum ATM.ctt

Es mag trivial erscheinen, aber bereits das vertikale Kippen des Baumes kann die Wahrnehmung positiv beeinflussen. Obwohl in der Informatik traditionell Bäume von oben nach unten aufgebaut werden, könnte es für den Anwender von Vorteil sein, die Herangehensweise umzudrehen. In der Realität beginnt ein Baum auch im Boden mit der Wurzel, und bildet nach oben gerichtet seine Äste und Blätter aus. Diese Analogie kann für manche Menschen die Arbeit mit Aufgabenmodellen zugänglicher machen, da sie aus der Natur jedem bekannt ist. Daher wird eine Flip-Funktion vorgeschlagen, die den Baum einfach in die entgegengesetzte Richtung der Wurzel spiegelt (vgl. Abbildung 4.2).

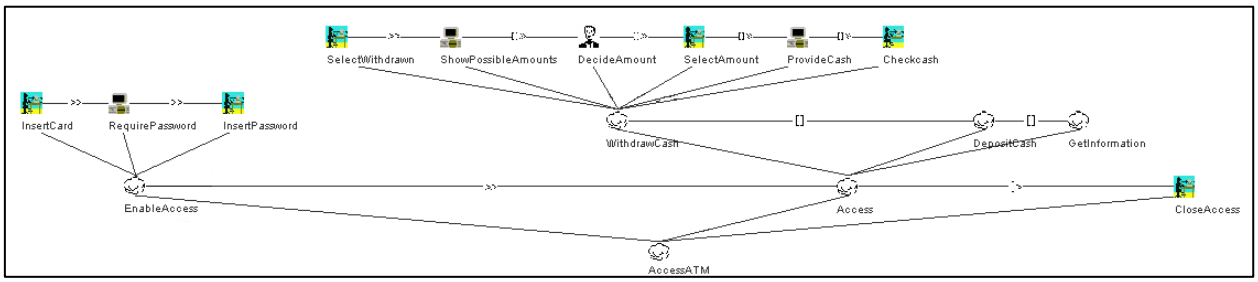


Abbildung 4.2 - Beispielbaum nach Flip

Mit Hilfe dieser Funktion ließen sich die Bäume genauso bearbeiten wie zuvor, jedoch mit veränderter Orientierung, die für manche Benutzer intuitiver sein kann – gerade dann, wenn sie sich zum ersten Mal mit Aufgabenmodellen beschäftigen. Die Reihenfolge und Leserichtung wird dabei nicht verändert, sodass keine Unstimmigkeiten zwischen den beiden Varianten entstehen. Soll später jemand anderes das Modell weiter bearbeiten, der die „top-down“ Ansicht präferiert, kann mit einem simplen Umkehren der Funktion die ursprüngliche Ansicht ganz einfach wieder herstellen.

Es wurden auch Überlegungen dazu angestellt, den Baum in Schichten respektive Ebenen einzuteilen, durch die navigiert werden kann, um Elemente einer Stufe des Baums zusammenhängend betrachten zu können. Dieser Denkansatz wurde jedoch wieder verworfen, weil damit die Übersicht über das gesamte Modell genommen würde und somit eher Verwirrung als Orientierung die Folge gewesen wäre.

4.1.2 Symbolik

Die Symbolik spielt für den Benutzer eine ganz entscheidende Rolle. Sie bietet nach einer kurzen Eingewöhnungsphase Wiedererkennungswerte sowie Assoziationen mit bestimmten Funktionen. Da die Symbolik aus CTTE weit verbreitet ist und auch schon in anderen Editoren Verwendung fand (Beispiel IdealXML), sollte diese auch weiterhin innerhalb der Diagramme genutzt werden. Diese Annahme wird dadurch bekräftigt, dass Personen mit dem richtigen Hintergrund diesen Wiedererkennungsfaktor direkt ausnutzen und sofort mit der Modellierung beginnen können.

Wie bereits in Kapitel 3.1.2 beschrieben, bietet CTTE jedoch schon standardmäßig die Möglichkeit an, die verwendeten Icons auszutauschen. Diese Funktion sollte genauso bestehen bleiben, da die verwendete Standardsymbolik unter Umständen nicht jedermanns Geschmack entspricht. Die

verwendeten Grafiken sind für heutige Verhältnisse nicht mehr zeitgemäß und wurden zusätzlich in niedriger Auflösung produziert – was besonders beim Vergrößern ein wiederkehrendes Problem darstellt, weil sich durch das Skalieren die Darstellung immer weiter verschlechtert. Weiterer Kritikpunkt ist, dass sich das Icon für die Cooperative Task kaum von dem einer Interaction Task unterscheidet, weil es nahezu identische Form- und Farbgebung verwendet. Obwohl ein simples Umfärben dieses Problem schon hinreichend lösen würde (z. B. in die Farbe Rot, s. Abbildung 4.3), soll trotzdem ein Vorschlag für ein etwas moderneres Icon Set gemacht werden.

Abbildung 4.3 - CTTE verbesserte Abgrenzung

Den besseren Ansatz für eine modernere Symbolik liefert das Icon Set von IdealXML (s. Abbildung 4.4). Die Icons sind zeitgemäß mit Licht- und Glanzeffekten ausgestattet und wirken dadurch mehr state-of-the-art. Weiterhin sind sie eindeutiger unterscheidbar – insbesondere was die Interaction und Cooperative Task angeht. Die beiden Figuren tragen unterschiedliche Kleidung und das Setzen der hinzugekommenen Figur in den Vordergrund der kooperativen Aufgabe bietet bessere Abgrenzung zur User Task. Einzig das i innerhalb der Sprechblase, welche die Abstraction Task repräsentiert, verwirrt aufgrund der unterschiedlichen Anfangsbuchstaben der Begriffe und sollte deshalb entfernt werden. Ebenso könnte sogar das ganze Icon wieder gegen eine Wolke ausgetauscht werden, da sich mit dieser eher Abstraktion assoziieren lässt als mit einer Sprechblase.

Abbildung 4.4 - Icon Set aus IdealXML

Das Icon Set lag jedoch nicht außerhalb des Programms vor, sodass die Auflösung unzureichend ist. Die Icons müssen deshalb komplett remodelliert werden: im besten Fall auf Basis von Vektorgrafiken, um beim Zoomen die optimale Schärfe der Icons zu erhalten, oder einfach als ausreichend große, herkömmliche Grafiken mit erhöhter Auflösung und Farbtiefe (bspw. heute übliche 256x256x32bit). Unter diesen Voraussetzungen sollte aber auch das standardmäßige Icon Set von CTTE nachmodelliert werden, um auch hier einen ausreichenden Vergrößerungsfaktor sicherzustellen.

4.1.3 Diagrammgestaltung

Bei der Gestaltung des Diagramms kommt es darauf an, so viele Informationen wie möglich sichtbar unterzubringen, ohne dass das Diagramm überladen wirkt. Bei CTTE liegen am Knoten außer der Bezeichnung der (Teil-)Aufgabe keine weiteren Informationen vor – es lassen sich zwar Vor- und Nachbedingungen usw. einstellen, aber bei einem bloßen Blick auf das Diagramm werden diese gemachten Optionen vom Benutzer nicht wahrgenommen. In dieser Hinsicht bietet K-MADe den besseren Ansatz, weil dort alle für einen Knoten verfügbaren Optionen in Kurzform direkt am Knoten arrangiert wurden (vgl. Abbildung 3.11). Diese Darstellung nimmt jedoch für einen Knoten sehr viel Platz in Anspruch, sodass diese Idee aufgegriffen und für die CTTE Darstellung optimiert wird.

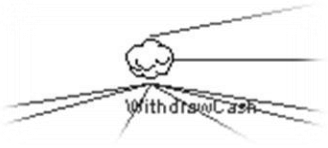


Abbildung 4.5 - CTTE: störende Linienführung

Besonders störend wirkt bei CTTE die Tatsache, dass die Kanten zu den Kinderknoten den Namen einer Teilaufgabe stets durchkreuzen (s. Abbildung 4.5). Auf diese Weise wird das Lesen stark beeinträchtigt und das Diagramm wirkt allgemein unruhig, besonders bei komplexerer Ausprägung. Darüber hinaus bietet das direkte Verbinden der Icons keinen Raum für spezielle Markierungen, die Eigenschaften des Knotens anzeigen. Aus diesem Grund ist es von Vorteil, für Kanten eine rechtwinklige Technik anstelle einer radialen zur Verbindung von Knoten zu verwenden (vgl. Abbildung 4.6). Das Diagramm wirkt bereits dadurch wesentlich organisierter, und auch grafisch bedingte Fehler wie eine Treppenbildung (sog. *Alias-Effekt*) entlang der Kanten wird gleichzeitig vermieden (obwohl dies bspw. durch Kantenglättung ebenso verbessert werden könnte).

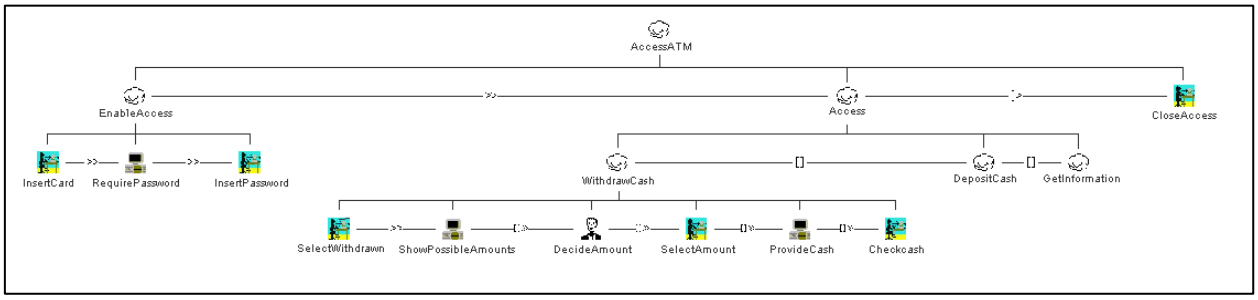


Abbildung 4.6 - Rechtwinklige Baumstruktur

Um alle zuvor genannten Nachteile zu vermeiden, wird (in Analogie zu K-MADe) zunächst ein definierter, rechteckiger Raum um das Icon des Knotens gelegt. Dieser ist in Abbildung 4.7 als roter, gestrichelter Kasten dargestellt und bietet Platz für die wichtigsten Eigenschaften. Er wird später nicht mehr zu sehen sein und dient in der Abbildung nur zur Veranschaulichung der Positionen. Innerhalb dieses Kastens können um das eigentliche Aufgabenicon herum kleinere Symbole sichtbar gemacht werden, die dessen Eigenschaften repräsentieren bzw. anzeigen, ob eine Eigenschaft aktiv ist oder nicht. Unterhalb des Icons ist Platz zur Beschriftung des Knotens sowie für Zeitangaben, welche beide in der Breite auch über den Kasten hinausgehen können. Dies birgt deshalb keine grafischen Konflikte, weil Kanten dank rechtwinkliger Verbindungen nur noch mittig am Rand des roten Kastens beginnen können und deshalb nie mit dem Text kollidieren, egal wie „breit“ dieser ist. Eine maximale Textlänge sollte aber doch gesetzt werden, damit Anwender die Namensgebung von Knoten kurz genug halten.

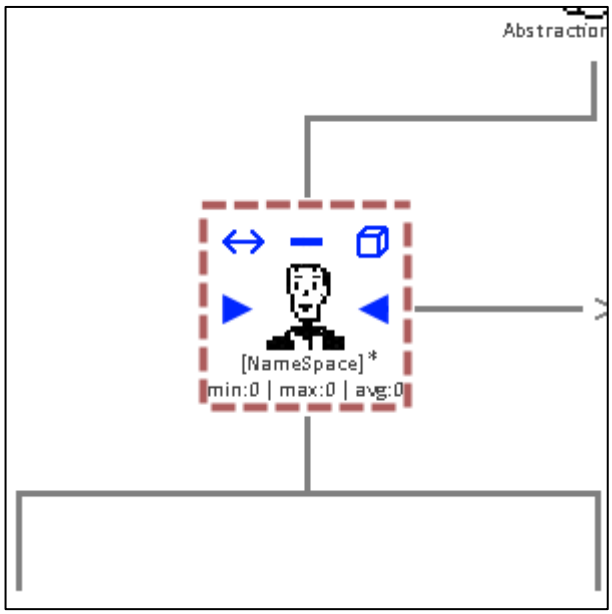


Abbildung 4.7 - Knotengestaltung

Die Darstellung von Optionalität sowie Iteration wurde von CTTE übernommen, mit dem Unterschied, dass nicht nur die Iteration selbst, sondern bei Bedarf auch die Anzahl der Iterationsschritte dargestellt werden kann. Dazu lässt sich das Sternsymbol durch eine entsprechende Ziffer ersetzen. Eckige Klammern um die Bezeichnung markieren den Knoten als optional. Die Zeiten unterhalb geben Aufschluss über die minimale, maximale und durchschnittliche Dauer (min|max|avg) in Sekunden, die für die Durchführung dieses Knotens veranschlagt wird. Außerdem wurden die folgenden Symbole eingeführt, um eine Visualisierung der Eigenschaften direkt am Knoten zu gewährleisten:

Symbol	Eigenschaft
	Knoten ist Teil einer kooperativen Aufgabe
	Knoten zu-/aufklappbar
	Knoten benötigt Objekte, um die Aufgabe durchführen zu können
	Knoten hat Vorbedingungen
	Knoten hat Nachbedingungen

Tabelle 4.1 - Eingeführte Symbolik am Knoten

Das erste der Symbole stammt aus den Standardicons von CTTE und bietet somit Wiedererkennungswert. Es markiert den Knoten als Teil einer kooperativen Aufgabe und zeigt die Verbindung zum übergeordneten Metamodell an. Die Symbole Plus und Minus signalisieren, ob der Knoten einklappbar ist (Minus) oder ob er bereits eingeklappt ist und ausgeklappt werden kann (Plus). Diese Symbolik ist bei der Verwendung von Bäumen üblich und weitestgehend aus Ordnerstrukturen in Dateimanagern bekannt (z. B. Windows Explorer). Das Würfelsymbol deutet an, ob für das Durchführen der Teilaufgabe Objekte benötigt werden. Zu guter Letzt bilden zwei zur Mitte gerichtete Pfeile links und rechts vom Aufgabenicon definierte Vor- und/oder Nachbedingungen des Knotens.

Das -/+ Symbol ist nur dann sichtbar, wenn der Knoten überhaupt Kinder besitzt und somit eingeklappt werden kann. Es ist dann persistent, kann jedoch nicht über das im nächsten Schritt vorgestellte Menü verändert werden, weil dafür später eine eigene Geste vorgesehen wird.

4.1.4 Menügestaltung

Im nächsten Schritt wird zum Bearbeiten von Knoten und Kanten im Diagramm ein Kontextmenü entwickelt. In erneuter Anlehnung an K-MADe entsteht ein Pie-Menü, das ringförmig um zu verändernde Knoten oder Kanten eingeblendet wird. Dabei bleibt das gesamte Blickfeld des Knoten mit all seinen Informationen erhalten (vgl. Abbildung 4.8a). Bei der Anordnung des Menüs wurde sehr darauf geachtet, dass sich die Schaltflächen in Ausrichtung zu den Symbolen aus 4.1.3 befinden. Es wurden acht Schaltflächen in den acht Himmelsrichtungen {N, NO, O, SO, S, SW, W, NW} positioniert. Diese dienen entweder zum Ein-oder Ausschalten von Funktionen (toggle), zum Hervorrufen weiterer Menüs oder um ein Fenster mit erweiterten Optionen zu erreichen. Des Weiteren wäre im Single-User Betrieb eine Abdunkelung der gesamten Restfläche von Vorteil, um den Fokus auf das zu bearbeitende Diagrammobjekt zu legen (angedeutet sowohl in Abbildung 4.8a und b). Diese Technik ist z. B. aus Webanwendungen oder Windows ab Produktversion „Vista“ für kritische Berechtigungsabfragen bekannt, stört jedoch das gemeinsame Arbeiten in einem nicht zu unterschätzenden Maße und wird deshalb wieder verworfen (Begründung in Kapitel 4.3).

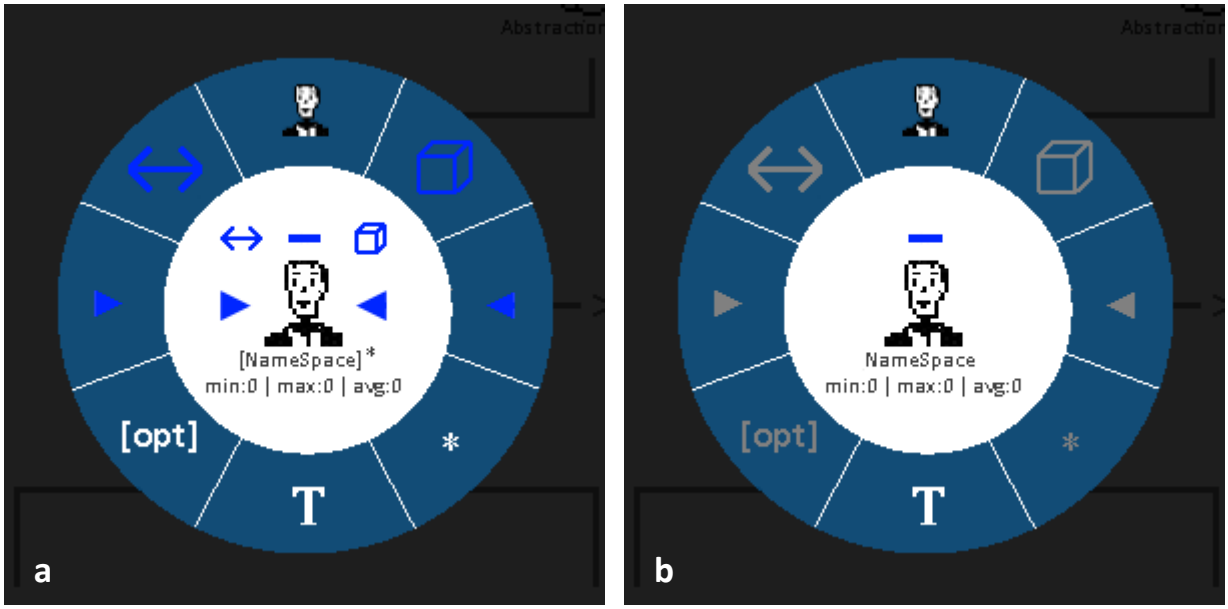


Abbildung 4.8 - Knoten PIE-Menü

Abbildung 4.8 zeigt das Menü einmal um einen voll definierten Knoten (a) und denselben Knoten ohne definierte Eigenschaften (b). Bei der Signalisierung aktiver und inaktiver Objekte wird auf Farbe gesetzt; momentan inaktive Elemente werden in einem Grauton gehalten (b), deren Farbe sich ändert, sobald sie aktiviert werden (a). Die oberen Felder wechseln dabei in die blaue Farbe, die auch am Knoten für die Symbolik verwendet wird, während die unteren Felder, die sich rein auf den Text unterhalb des Knotens beziehen, zu weiß wechseln. Die Schaltfläche für das Aufgabentypicon {N} und das Textsymbol {S} nehmen dabei eine besondere Rolle ein, da sich oberes dynamisch dem ausgewählten Knotentyp anpasst, während das untere immer gleich bleibt.

Der Doppelpfeil {NW} und das [opt] {SW} sind lediglich zum Umschalten der Aufgabe als kooperatives und optionales Element vorgesehen. Die beiden Pfeile {W,O}, der Würfel {NO} und das Textsymbol {S} erfordern für die weitere Bearbeitung ein neues Fenster, in dem beispielsweise das Erstellen von Bedingungen, verwendbaren Objekten oder einfache Texteingaben ermöglicht werden. Darauf wird im anschließenden Abschnitt 4.1.5 detailliert eingegangen. Die Verwendung der Typschaltfläche {N} oder die der Iteration {SO} ersetzen das ursprüngliche Pie-Menü durch neue Elemente, gezeigt in Abbildung 4.9 und Abbildung 4.10.

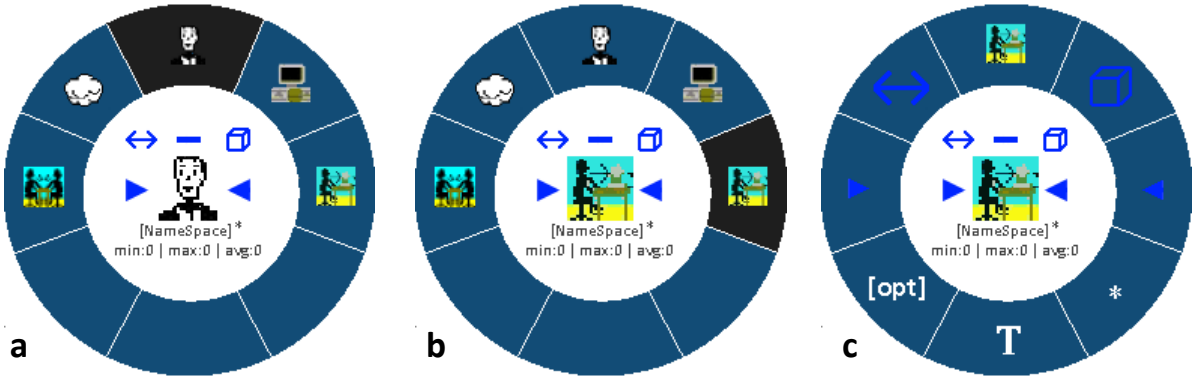


Abbildung 4.9 - Knotentypmenü

Nachdem vom Ur-Menü das Typsymbol ausgewählt wurde {N}, wandelt sich der ursprüngliche Ring zu einem neuen Ring, der nur aus den bekannten Typsymbolen besteht. Das aktuell ausgewählte Symbol ist dabei markiert (a) und kann auf jedes der angezeigten geändert werden. Wählt man einen neuen

Aufgabentyp aus (b), ändert sich sofort das Knotensymbol in der Mitte entsprechend ab und die Markierung wechselt auf das gewählte Symbol im Ring. In Konsequenz wird im Ur-Menü jetzt das aktuelle Knotensymbol verwendet (c), sodass direkt wiedererkannt wird wo man es erneut wechseln kann. Bei der Erstellung neuer Knoten sollte grundsätzlich immer die Variante (a) auftauchen, damit der Typ des neuen Knotens direkt bestimmt werden kann. Es ist dabei zu überlegen, ob sich am zuletzt verwendeten Typ orientiert wird oder immer standardmäßig z. B. die Wolke erscheint, sobald neue Knoten erzeugt werden.

Weiterhin sei an dieser Stelle angemerkt, dass aus den erweiterten Ringmenüs aktiv zum Ur-Menü zurück gewechselt werden muss, nachdem eine Auswahl getroffen wurde. Dies könnte z.B. durch ein Tippen in die Mitte respektive auf den gesamten weißen Kernbereich des Rings vollzogen werden. Zusätzlich muss das Symbol links (a) für den Typ der kooperativen Aufgabe {W} eine Abfrage mit einem Hinweis erzeugen, dass die Modellart auf das kooperative Modell erweitert wird, falls dies nicht direkt von Beginn an gewählt wurde. Für den Fall, dass bereits ein kooperatives Metamodell existiert und es sich in der aktuellen Sicht nicht um dieses handelt, könnte die Verwendung dieses Knotentyps sogar unterbunden werden, bspw. indem er erst gar nicht im Menü angeboten wird.

Analog zu den gerade gemachten Erläuterungen wurde auch das Iterationsmenü aufgebaut (s. Abbildung 4.10). Nachdem im Ur-Menü das Sternsymbol {SO} ausgewählt wurde, wechselt das Menü zu einem neuen Ring (a), der Anpassungen der Iteration ermöglicht. War als Iterationsparameter der Stern ($*$) ausgewählt, ist dieser in der nächsten Ansicht markiert {O}; war keine Iteration angegeben, ist das X markiert {W}.

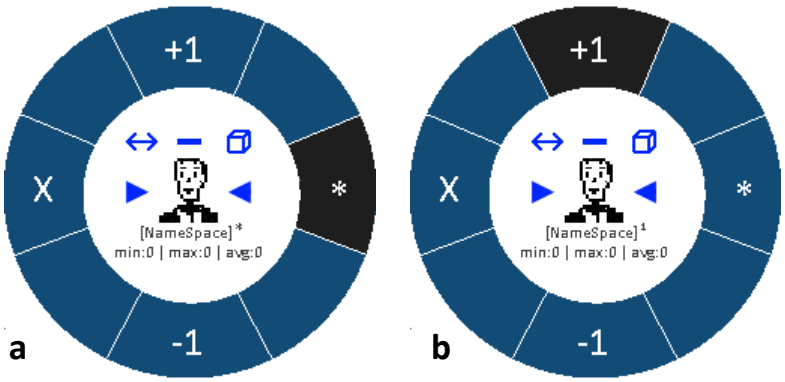


Abbildung 4.10 - Iterationsmenü

Sobald der Anwender beginnt, die Anzahl der Iterationen mit +1 {N} zu erhöhen, wird die Markierung des Sterns {O} bzw. X {W} aufgehoben. Synchron dazu ändert sich auch – entsprechend der Anzahl n der durchgeführten Schritte – der Zähler des hochgestellten Zeichens an der Bezeichnung des Knotens (im Beispiel *[NameSpace]*) inkrementell von $1..n$ (b), sodass der Benutzer ein direktes Feedback zum Ergebnis seiner Handlung erhält. Zur weiteren Rückmeldung an den Benutzer leuchtet nach Betätigung von +1 deren Fläche einmal kurz auf. Die Zahl kann auf die gleiche Weise mit -1 {S} dekrementiert werden. Ist die Zahl 0 erreicht, wird wieder das X {W} markiert. Die Anzahl von null Iterationen resp. das Abschalten der Iterationsfunktion kann aber mit Hilfe des X auch direkt erreicht werden. Auch kann per Betätigen des Sterns {O} von jeder beliebigen Anzahl wieder auf $*$ gewechselt werden; ein erneutes Hochzählen aus dem Zustand $*$ oder X beginnt jedoch immer wieder bei 1.

Abschließend ist noch zu erwähnen, dass dieselbe Darstellung und Funktionsweise auch für das Ändern von Kanten vorgesehen ist. Abbildung 4.11 zeigt beispielhaft, wie Kanten verändert werden können. Nachdem die gewünschte Kante ausgewählt wurde, erscheint um die temporale Relation herum das Pie-

Menü, in dem die momentan verwendete Relation wieder markiert ist (a). Verändert man die Kantenart durch Betätigen eines der übrigen Felder, ändert sich das Symbol in der Mitte und die ausgewählte Fläche wird hervorgehoben (b).

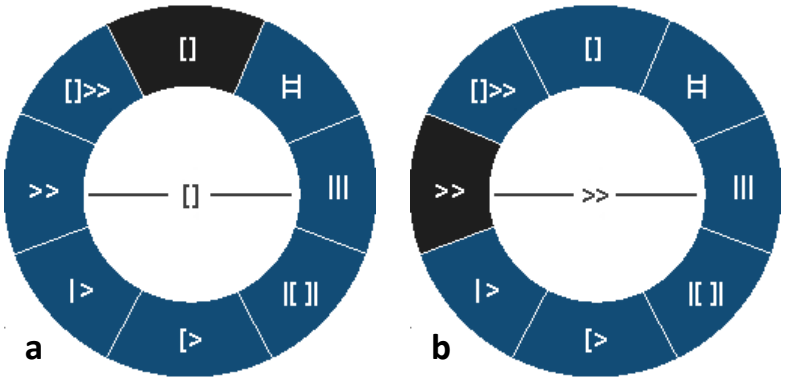


Abbildung 4.11 - Kanten PIE-Menü

Idealerweise sollte der eingeblendete Ring die beiden verbundenen Knoten nicht verdecken, damit die Orientierung des Anwenders gewahrt bleibt. Dieses Menü (a) sollte dem Benutzer ebenfalls direkt nach dem Erstellen einer entsprechenden Kante präsentiert werden, um sofort die Art der temporalen Relation festzulegen. Änderungen können jederzeit und ohne Aufwand vorgenommen werden. Nachdem nun ausgiebig die Verwendung von Menüs im Konzept vorgestellt wurde, soll es im nächsten Schritt um Optionen gehen, für die das Pie-Menü an seine Grenzen gerät und deshalb nicht geeignet ist.

4.1.5 Erweiterungsfenster

Wie bereits im vorherigen Abschnitt erwähnt gibt es Ausprägungen an einem Knoten, die zu mächtig sind, als dass sie mit einem Pie-Menü bedient werden können. Zu diesen zählt beispielsweise der Formeleditor für Bedingungen, der bei CTTE sehr stark ausgeprägt ist und viele Einstellmöglichkeiten bietet. Neben dem Formeleditor spielen in dieser Hinsicht aber auch noch am Knoten benötigte Objekte sowie das Bearbeiten seiner Bezeichnung und Zeitwerte gleichermaßen eine Rolle, für die ein sinnvolles Pie-Menü zu komplex würde.

Bildschirmtastatur

Für alle folgenden Fälle wird eine Texteingabe benötigt. Diese sollte – wie bei Touch-Anwendungen gängige Praxis – über ein sogenanntes On-Screen Keyboard erfolgen, einem Software Keyboard, das auf dem Bildschirm angezeigt und bedient wird. Zur Veranschaulichung wurde die Windows 7 eigene Bildschirmtastatur beispielhaft auf ein Minimum an benötigten Tasten reduziert und dient hier als Referenzmodell (s Abbildung 4.12). Verwendet werden kann aber auch jegliche andere Software Tastatur, bspw. die aus Googles Mobilbetriebssystem Android bekannte Tastatur für Tablets oder Smartphones.

Abbildung 4.12 - Beispiel On-Screen Keyboard auf Windows 7 Basis

Die Tastatur sollte bei Auswählen eines Texteingabefeldes automatisch zur Zeicheneingabe eingeblendet werden. Dazu ist es sinnvoll, diese Tastatur von unten auf dem Bildschirm einzublenden, da dieser Bereich garantiert vom Anwender erreicht werden kann. Die horizontale Position kann abhängig vom Knoten sein und somit variieren. Wichtig ist, dass der zu bearbeitende Eingabebereich sofort sichtbar ist, bspw. indem die unter der virtuellen Tastatur liegende Ansicht entsprechend automatisch verschoben und ein Textcursor eingeblendet wird – wie von Smartphones bekannt. Über die Taste SYM können Sonderzeichen und Zahlen erreicht werden und der nach unten gerichtete Pfeil rechts unten blendet die Tastatur bei Bedarf einfach wieder aus.

Price Tag

Für die generellen Einstellungen am Knoten wurde als Metapher eine Preisschild-Optik (engl. *price tag*) verwendet, da sie rein textueller Natur sind (s. Abbildung 4.13). Sie können über das Pie-Menü durch Auswahl des T-Symbols {S} erreicht werden. Zur Orientierung, welcher Knoten gerade bearbeitet wird, soll dieser im Tag enthalten sein. Innerhalb des Tags lässt sich der Name des Knotens verändern sowie die beiden Felder *Type* und *Frequency* parametrisieren, welche von CTTE übernommen wurden und nicht näher erläutert werden. Für die Auswahl dieser Felder stehen Bullet Points zur Verfügung, die mit dem Finger direkt ausgewählt werden können. Der Inhalt des Feldes Type ändert sich in Abhängigkeit des gewählten Knotentyps dynamisch.

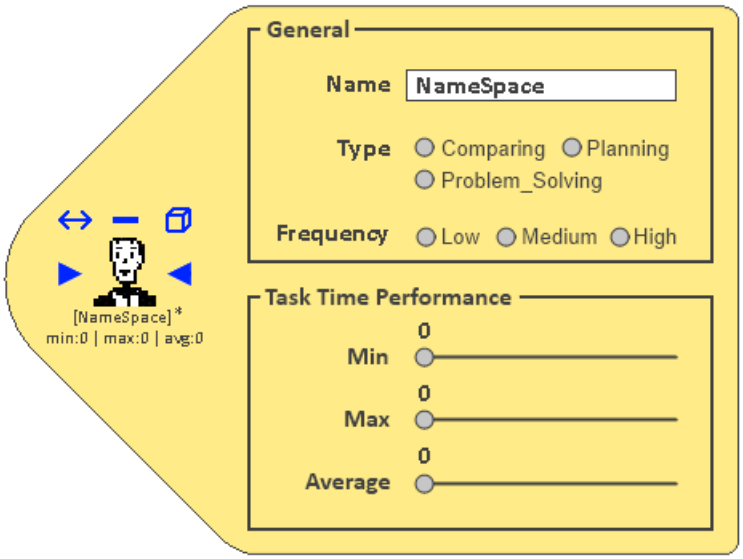


Abbildung 4.13 - Knoteneigenschaften "Price Tag"

Zeitangaben können mit Hilfe von Schiebereglern angepasst werden. Dabei wandert die Zahl oberhalb des Reglers an ihm ausgerichtet mit und zeigt gleichzeitig den momentan ausgewählten Wert in Sekunden an. Für den Average Schieberegler könnte die Einschränkung gelten, dass dieser nur Werte zwischen Min und Max annehmen kann; alles andere würde logisch keinen Sinn ergeben. Gleichzeitig gilt auch, dass Max immer größer oder gleich Min sein muss. Eine Änderung der Schieberegler soll – wie auch die Änderung des Namens – direkt am Knoten (links) sichtbar sein.

Denkbar ist, dass dieser Price Tag auch nach links ausgerichtet erscheinen kann (um 180° gedreht), wenn man einen Knoten am rechten Rand bearbeitet, neben dem nach rechts unmittelbar „kein Platz mehr“ ist. Um das Price Tag zu verlassen, reicht ein Berühren des angezeigten Knotens. Die gemachten Änderungen werden dabei sofort gespeichert und zurück zum Pie-Menü gewechselt.

Aufgabenobjekteditor

Als nächstes wird beschrieben, was nach Betätigen des Würfelsymbols {NO} geschieht. Zur Anlage von Aufgabenobjekten soll sich ein eigenes Fenster öffnen, in dem der Pool der Objekte verwaltet wird. Ein Entwurf dazu spiegelt sich in Abbildung 4.14 wieder. Sichtbar ist (wie zuvor) immer durchgängig der gerade bearbeitete Knoten oben links, mit dem auch durch Berühren gespeichert und zurückgegangen werden kann. Sein blaues Würfelicon wird erst dann aktiviert, wenn sich mindestens ein Aufgabenobjekt im Pool oben rechts befindet.

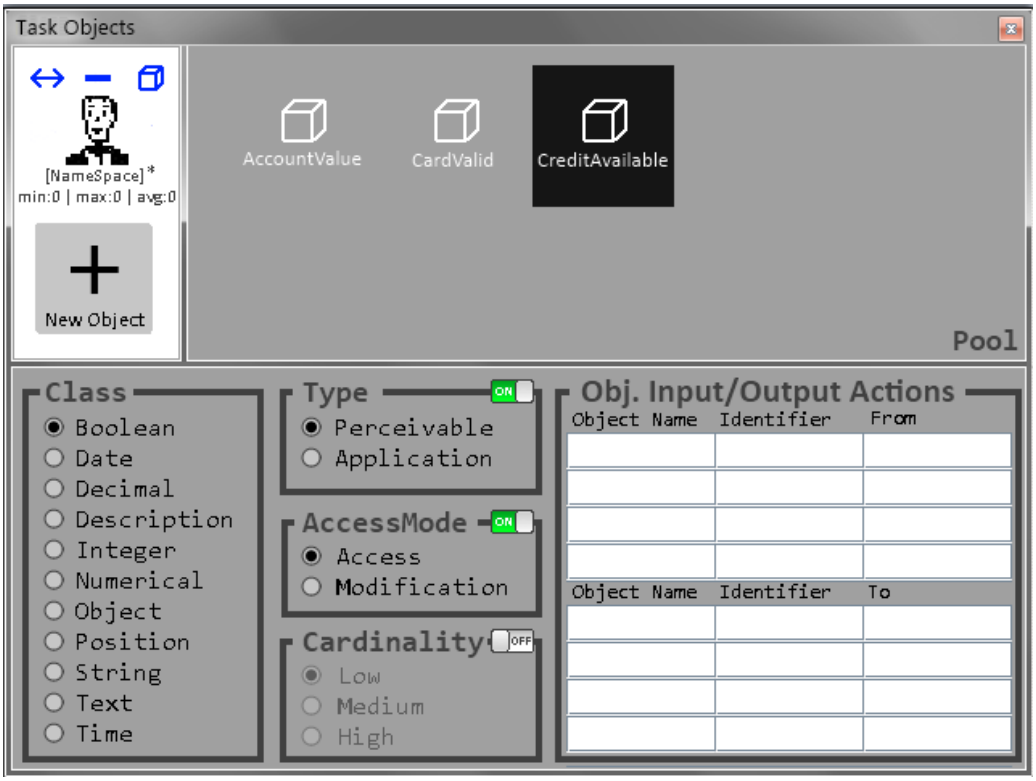


Abbildung 4.14 - Entwurf: Aufgabenobjekteditor

Zum Pool hinzufügen lassen sich neue Objekte mit Hilfe der Plus-Schaltfläche, die sich unterhalb des Knotensymbols im Fenster befindet. Bei Betätigung wird sogleich ein neuer Würfel innerhalb des Poolbereichs erzeugt, der mit Hilfe der eingeblendeten Tastatur benannt wird. Der momentan ausgewählte Würfel ist markiert, und seine Eigenschaften wie *Class*, *Type*, *AccessMode* und *Cardinality* lassen sich im unteren Bereich einstellen. Am wichtigsten ist dabei die Klassifizierung des Objekts, also um welche Klasse von Objekt es sich handelt. Diese Einstellung ist deshalb so wichtig, weil sie Auswirkungen auf den nachfolgend beschrieben Formeleditor hat. Alle anderen aus CTTE übernommenen Eigenschaften werden wegen geringer Relevanz nicht näher erläutert. Sie können aber auch mit Hilfe der Umschalter in der oberen rechten Ecke der Kästen ganz deaktiviert werden.

Formeleditor für Bedingungen

Für den Formeleditor der Bedingungen, der in CTTE sehr ausgeprägt ist, sollte sich nach Betätigen des Vor- {W} oder Nachbedingungssymbols {O} aus dem Pie-Menü (ebenso wie für die Aufgabenobjekte) ein gesondertes Fenster öffnen, in dem die Bedingungen definiert werden. Dieser sollte konsequenterweise auf dieselbe Art wie das Diagramm selbst aufgebaut sein und gesteuert werden, da es sich wiederum um eine Baumstruktur handelt, die jedoch aus andersartigen Knoten und einfacheren Kanten besteht. Grundsätzlich kommen als Knotentypen AND, OR und XOR in Frage und Kanten besitzen keine Typen,

sodass dort auf eine Symbolik verzichtet werden kann (s. Abbildung 4.15). Wieder ist oben links zur Orientierung der zugehörige Knoten fest angeordnet und die Arbeitsfläche soll sich innerhalb des Fensters verschieben lassen, um auf evtl. Komplexität der Regeln reagieren zu können. Die Pie-Menüs sollen auch innerhalb dieses Fensters zur Verfügung stehen.

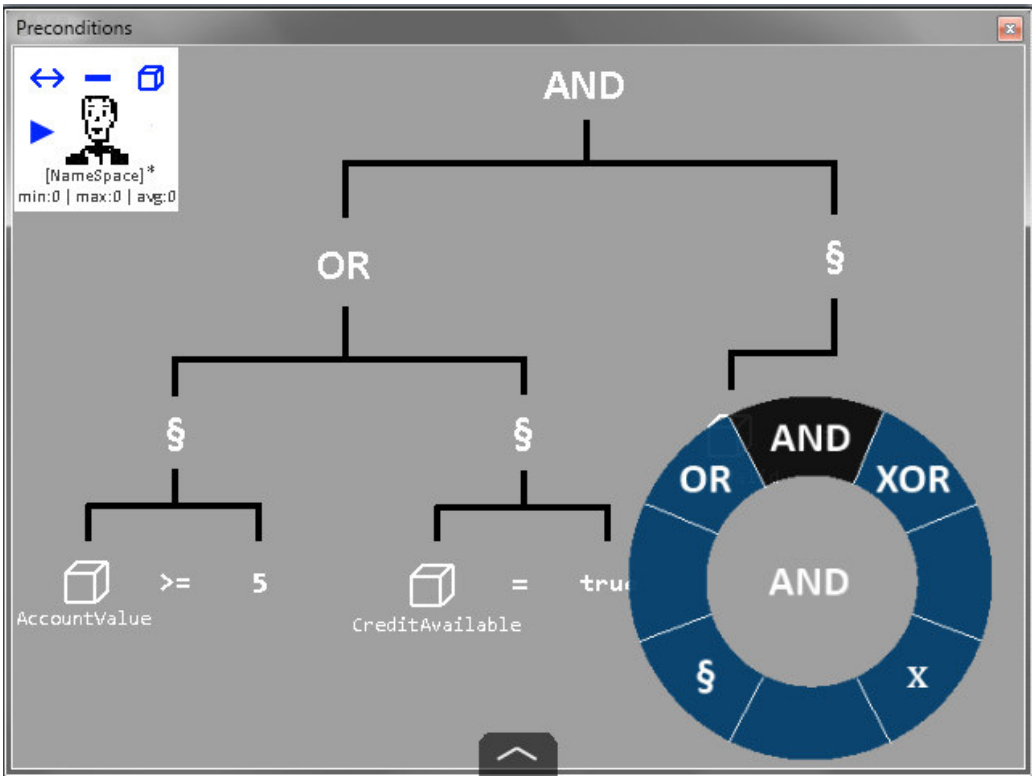


Abbildung 4.15 - Entwurf: Formeleditor für Vor- sowie Nachbedingungen

Neu eingeführt wurde der Knotentyp in Form eines Paragraphen (§), der eine Regel am Ende einer logischen Verknüpfung repräsentiert. Ein AND, OR oder XOR Knoten muss immer mit Paragraphenknoten oder weiteren logischen Verbindern enden. Da für die Definition von Regeln immer zuerst Aufgabenobjekte erzeugt worden sein müssen (wie zuvor mit dem Objekteditor beschrieben), können diese über ein Randmenü ausgewählt und eingefügt werden (Abbildung 4.16). Dieses lässt sich innerhalb des Formeleditorfensters von unten aufklappen, indem der angedeutete Pfeil am unteren Fensterrand ausgewählt wird (vgl. Abbildung 4.15).

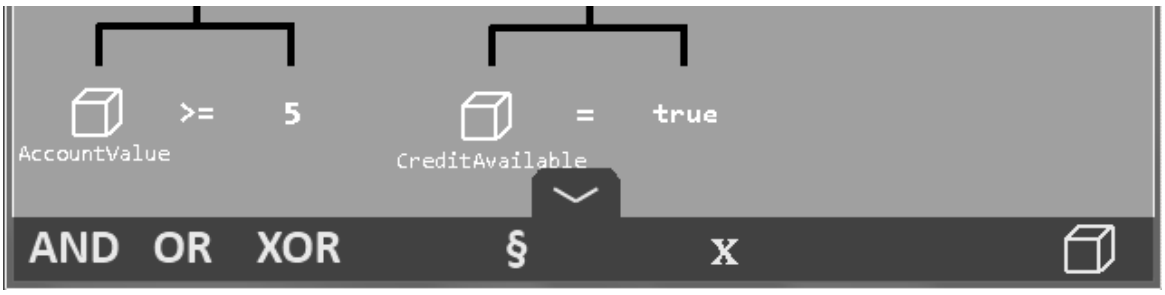


Abbildung 4.16 - Randmenü des Formeleditors

Daraufhin öffnet sich das in Abbildung 4.16 gezeigte Menü, das alle verfügbaren Elemente alternativ zum Einfügen per Drag&Drop beinhaltet (das Einfügen und Bearbeiten der Knoten und Kanten soll jedoch auch auf die gleiche Weise wie im Aufgabenmodell selbst durch Gestik funktionieren, was jedoch erst später in Kapitel 4.2.1 detailliert beschrieben wird). Werden bei einer Vergleichsoperation nicht zwei Objekte miteinander verglichen, sondern ein Objekt auf ein Literal geprüft, wird nach dem Objekt

zunächst das x-Symbol als Stellvertreter eingefügt, das dann weiter spezifiziert werden kann (wie im Beispiel „5“ oder „true“). Wie schon in CTTE richten sich die Vergleichsoperatoren dynamisch nach dem Datentyp des Objektes, auf dem die Operation durchgeführt wird. Deshalb sollen sie auch genauso abhängig davon wählbar sein, sodass man beispielsweise ein Boolesches Objekt nur auf gleich oder ungleich und nicht etwa auf größer oder kleiner prüfen kann, was dieser Datentyp gar nicht zulassen würde.

Um am Ende der Kette Aufgabenobjekte überhaupt einfügen zu können, muss eine Verbindung zwischen dem Aufgabenobjekteditor und dem Formeleditor bestehen. Durch Auswahl des Würfelsymbols erhält man Zugriff auf die im Objekteditor angelegten Objektinstanzen.

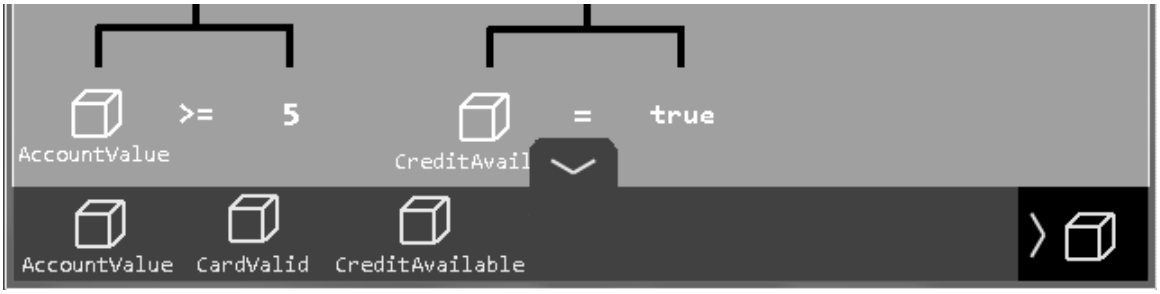


Abbildung 4.17 - Objektmenü des Formeleditors

Die Objekte aus dem Pool lassen sich nun benutzen, um an einem §-Knoten eine Regel aufzubauen (Abbildung 4.17). Dazu muss zunächst ein Würfel mit dem Paragraphen verbunden werden. Im Anschluss daran kann entweder ein weiteres Objekt (Abbildung 4.18a) für die Bedingung angehängt werden, oder das x um beispielsweise ein Literal (b&c) wie einen Zahlen- oder Wahrheitswert für den Vergleich zu definieren.

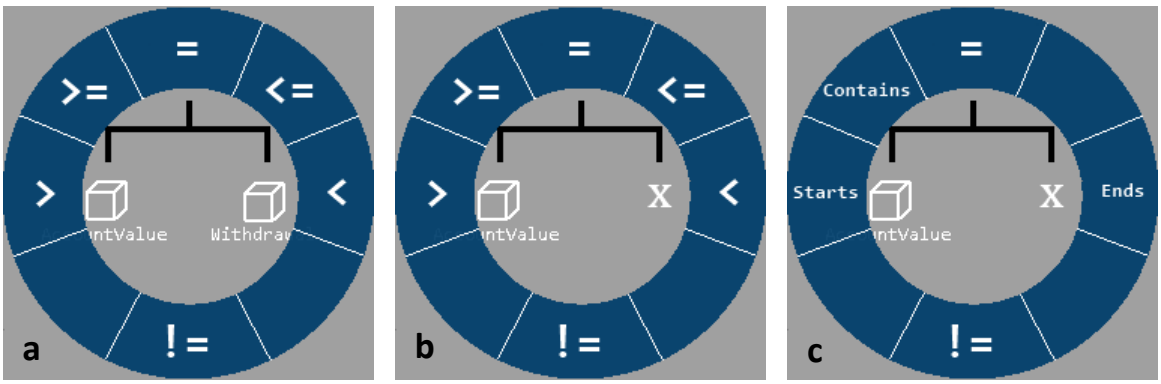


Abbildung 4.18 - Pie-Menü für Vergleichsoperatoren

Unabhängig davon, ob an einem Paragraphen-Knoten entweder zwei Objekte oder ein Objekt und ein x-Knoten untergeordnet wurden, öffnet sich das Kontextmenü (a&b). Dieses beinhaltet von der Klasse des ersten Objekts (linke Seite) abhängige Operatoren, sodass auch völlig unterschiedliche Optionen im Menü möglich sind (b&c) – je nach Art des Objekts. Der Operator kann nachträglich wieder geändert werden, indem das gerade erzeugte Symbol (z. B. =, !=, <, > etc.) wieder angewählt wird. Nachdem die Auswahl des Operators getroffen wurde, kann durch Anwählen des verknüpften x ein Literal an dem Knoten definiert werden (s. Abbildung 4.19). Dabei sind abhängig vom zu vergleichenden Objekttyp sinnvolle Werte vorgegeben, es kann aber auch eine manuelle Eingabe über die bekannte Textschaltfläche erfolgen (erforderlich z. B. bei Strings). Bei Tätigen einer Eingabe wird das x entsprechend ersetzt.

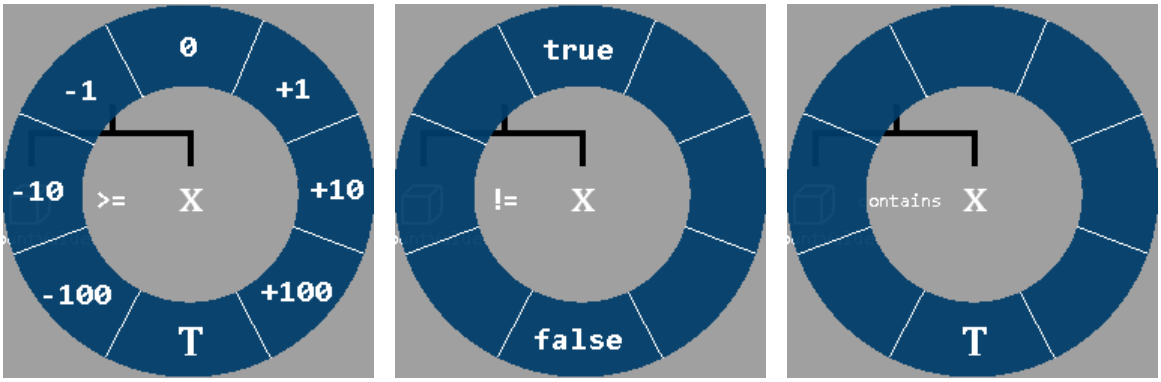


Abbildung 4.19 - Definition der Literale

Die definierten Regeln sollen durch Berühren des Icons des Ursprungsknotens in der oberen linken Ecke des Fensters gespeichert werden (vgl. Abbildung 4.15). Wird das Fenster wie sonst üblich geschlossen, werden die Änderungen nach einer Sicherheitsabfrage entweder gespeichert oder verworfen.

Rollenmanager

Den wichtigsten der virtuellen Bildschirme innerhalb der Applikation stellt der Rollenmanager dar. Auf ihm werden, für den besonderen Fall eines kooperativen Aufgabenmodells, der Metabaum erstellt und die dazugehörigen Rollen definiert. Zur Veranschaulichung wird in diesem Abschnitt das Beispiel *GuitarSalesOrder.cctt* verwendet, welchem ein kooperatives Modell zugrunde liegt und welches bei CTTE mitgeliefert wird (ohne der bisher in diesem Kapitel vorgeschlagenen Optimierungen).

Im Rollenmanager kann über eine Leiste am unteren Bildschirmrand das gesamte Projekt mit allen Diagramminstanzen und weiteren Optionen verwaltet und gespeichert werden. Innerhalb des Rollenmanagers besteht ein Pool der im Projekt vorhandenen Diagramme in Form einer Liste, die von unten eingeblendet wird. Gleichzeitig lässt sich auch von der Leiste die Rollenverwaltung einblenden, um Rollen anzulegen und diese Diagrammen zuzuweisen. Den Ausgangsbildschirm zeigt Abbildung 4.20. Sichtbar ist das kooperative Modell, das in dieser Ansicht bearbeitet wird, sowie am unteren Bildschirmrand die Steuerungsleiste.

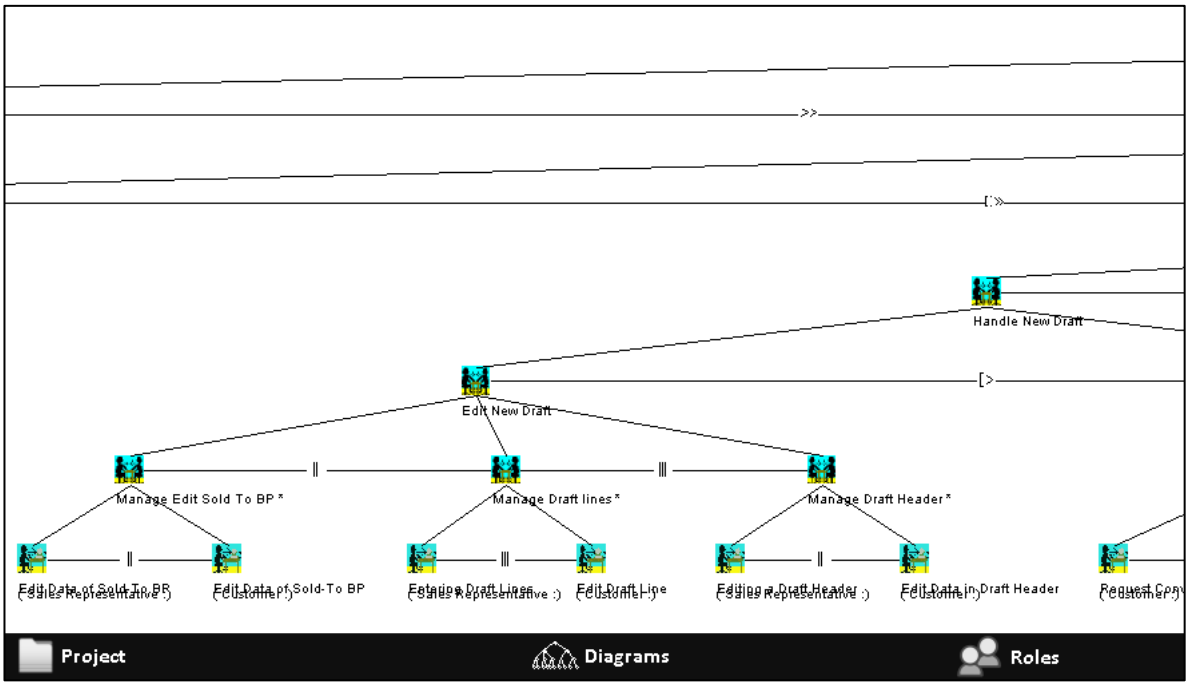


Abbildung 4.20 - Rollenmanager-Ansicht mit Bottom-Leiste

Über die Schaltfläche *Project* lassen sich grundsätzliche Einstellungen am Projekt vornehmen, wie das zusammenhängende Speichern der Diagramme, die Sprachauswahl oder das explizite Ein- oder Ausschalten des kooperativen Modus (s. Abbildung 4.21). Der Name des Projekts lässt sich ebenso verändern wie das Format, in dem das Modell abgelegt wird. Das Ändern der Sprache wirkt sich auf die gesamte Applikation aus und ändert alle Begriffe im User Interface entsprechend ab. Weiterhin könnte sich über dieses Menü die in Kapitel 4.1.1 angesprochene FLIP-Ansicht einschalten lassen, die die Diagramme umgedreht darstellt.

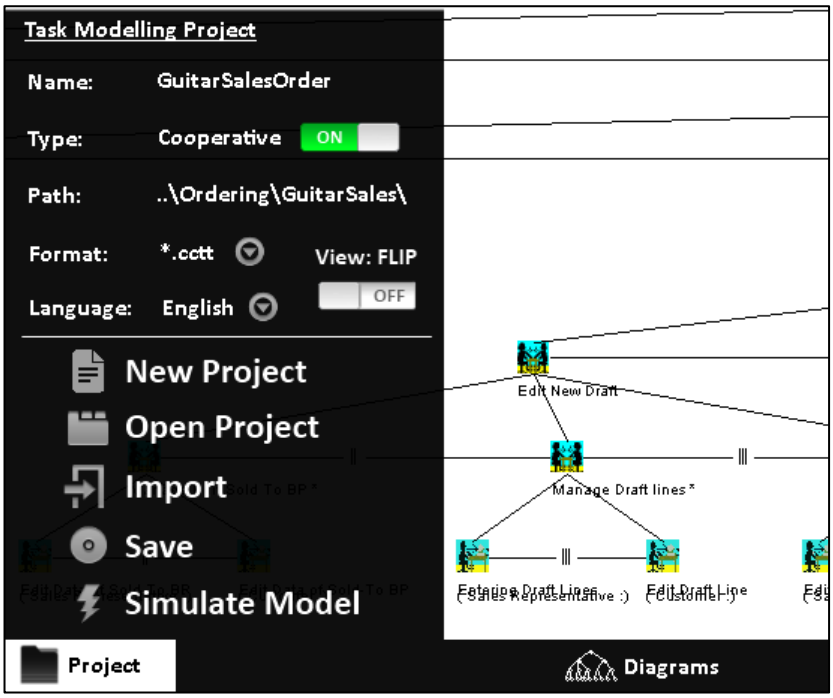


Abbildung 4.21 - Rollenmanager: Project Menü

Es soll über den Rollenmanager über das Project-Menü neben dem Speichern außerdem möglich sein, neue Projekte anzulegen, vorhandene zu öffnen oder in das bestehende Projekt die Diagramme anderer Projekte zu importieren. Diese werden dann einfach dem Diagrammpool hinzugefügt und stehen für weitere Bearbeitung zur Verfügung. An dieser Stelle ist weiterhin der sinnvollste Platz, um den Start der Modellsimulation zu initiieren. Beim kooperativen Modus ist wichtig, dass dieser automatisch aktiviert wird, sobald an irgendeinem der Knoten innerhalb des Projektes die Eigenschaft *ConnectionTask* gesetzt ist. Wird er deaktiviert, gibt es das Diagramm des Metabaums nicht, sodass der gesamte Arbeitsbereich im Rollenmanager leer bleibt.

Die innerhalb des Projektes bestehenden Diagramminstanzen können über *Diagrams* erreicht werden. Es öffnet sich eine Liste der im Projekt existierenden Diagramme, durch die gescrollt werden kann. (vgl. Abbildung 4.22). Falls die Arbeitsfläche für mehrere Benutzer aufgeteilt ist, wird angezeigt, auf welchem Bildschirmteil sich das Diagramm zu diesem Zeitpunkt befindet. Über die Buttons am unteren Rand können neue Diagramme erzeugt oder ausgewählte Diagramme kopiert oder gelöscht werden. Dazu können die Einträge aus der Liste mit dem Finger einfach auf das Duplizieren- oder das Löschen-Icon gezogen und abgelegt werden. Dieses Menü funktioniert unabhängig vom *Cooperative Mode.*

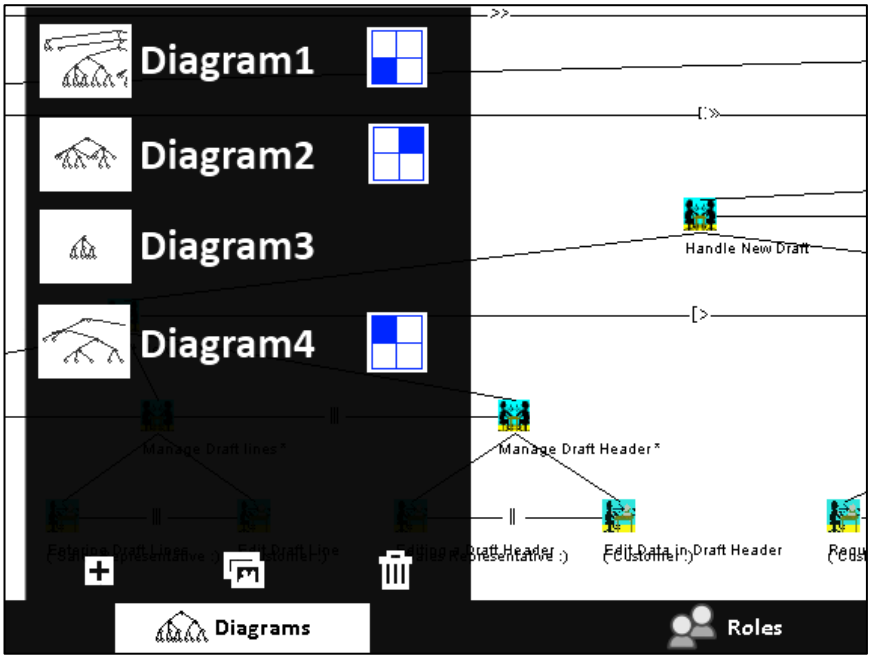


Abbildung 4.22 - Rollenmanager: Diagrams-Menü

Über das *Roles*-Menü lassen sich für den kooperativen Fall Rollen definieren und vorhandenen Diagrammen zuweisen (s. Abbildung 4.23). Im oberen Abschnitt werden die angelegten Rollen angezeigt. Eine symbolische Kette am Rand des Rolleneintrags zeigt die Verknüpfung mit einem Diagramm an. Sind auf dem verknüpften Diagramm *Connection Tasks* definiert, werden alle mit dem Connection-Symbol markierten Teilaufgaben automatisch als Einträge unterhalb in der Liste angezeigt. Eine Art Ampel signalisiert, ob die entsprechenden Aufgaben bereits im Metamodell verknüpft wurden (grün) oder noch nicht (rot). Auf diese Weise erhält man direkt einen Überblick über die noch zu tätigenden Verknüpfungen und die Komplexität des Modells.

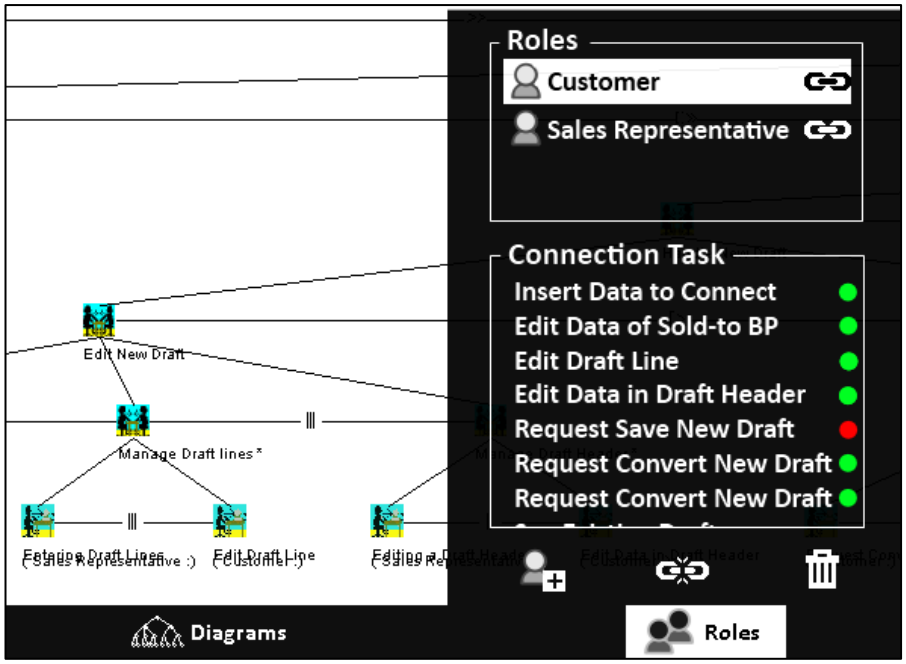


Abbildung 4.23 - Rollenmanager: Roles-Menü

Am unteren Rand besteht die Möglichkeit, neue Rollen anzulegen oder bestehende zu entfernen. Mit Hilfe der Unlink-Schaltfläche in der Mitte kann die Verknüpfung mit einem Diagramm wieder gelöst werden. Um eine Verknüpfung aber zunächst erst einmal herzustellen, müssen das Diagrams- sowie das

Roles-Menü parallel geöffnet werden (s. Abbildung 4.24). Per Drag&Drop kann nun die Rolle aus dem Menü rechts auf eine entsprechende Diagramminstanz nach links gezogen werden, um die Zuweisung zu vollenden. Zur Signalisierung wird im Roles-Menü das Kettensymbol an der Rolle aktiviert, während im Diagrams-Menü – oder besser gesagt in der gesamten Anwendung – der Name des Diagramms mit dem der Rolle ersetzt wird. Sobald dieser Schritt erfolgt ist, wird das ConnectionTask Feld im Roles-Menü, wie bereits erwähnt, automatisch mit Einträgen gefüllt (sofern sich auf dem verknüpften Diagramm ConnectionTask Markierungen befinden). Löst der Anwender die Verbindung wieder, erhält das Diagramm seinen ursprünglichen Namen zurück, und das Kettensymbol sowie alle ConnectionTask Verknüpfungen werden auch wieder entfernt.

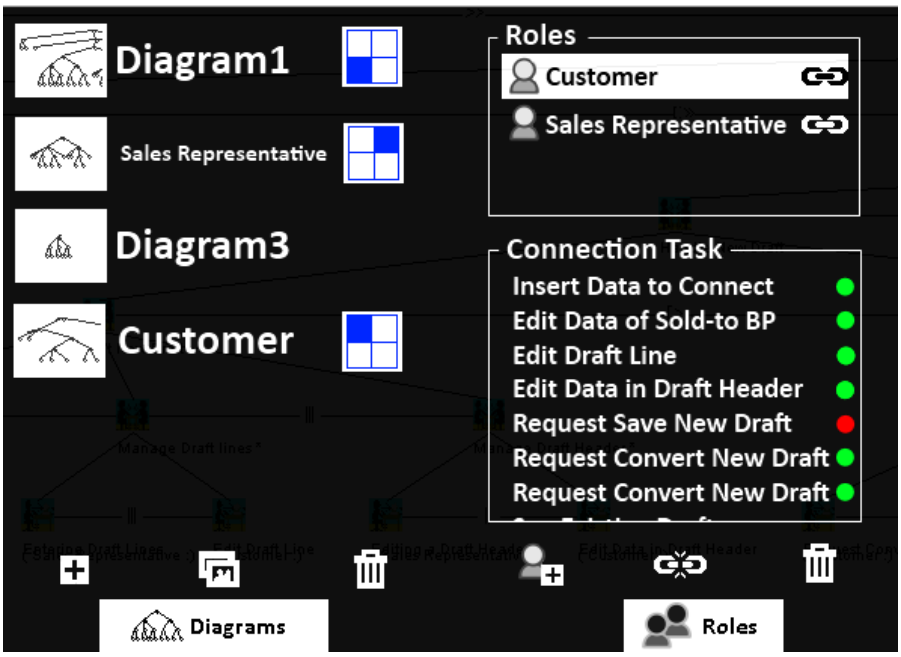


Abbildung 4.24 - Verlinkung von Diagrammen und Rollen

Als letztes fehlt noch das Vorgehen zur Verbindung der Metamodell-Knoten mit den ConnectionTask-Knoten der angelegten Rollen. Da im Allgemeinen die typische Vorgehensweise ist, zuerst das kooperative Modell aufzubauen, bevor seine Verfeinerung durch speziell angelegte Rollen erfolgt, muss es bereits dort möglich sein, Knoten schon vorab Rollen zuzuweisen, bevor konkrete Ausprägungen an ConnectionTasks aus den jeweiligen Rollenmodellen verfügbar sind. Dazu wurde im Pie-Menü innerhalb des kooperativen Modells das ConnectionTask Symbol durch ein Verkettungssymbol ersetzt (vgl. Abbildung 4.25, links). Wählt man diese Option an, öffnet sich die bekannte Liste der Connection Tasks aus dem Roles-Menü. Diese kann zunächst leer sein, wenn noch keine Rollen angelegt wurden (rechts). Das Anlegen einer Rolle kann direkt im Menü erfolgen.

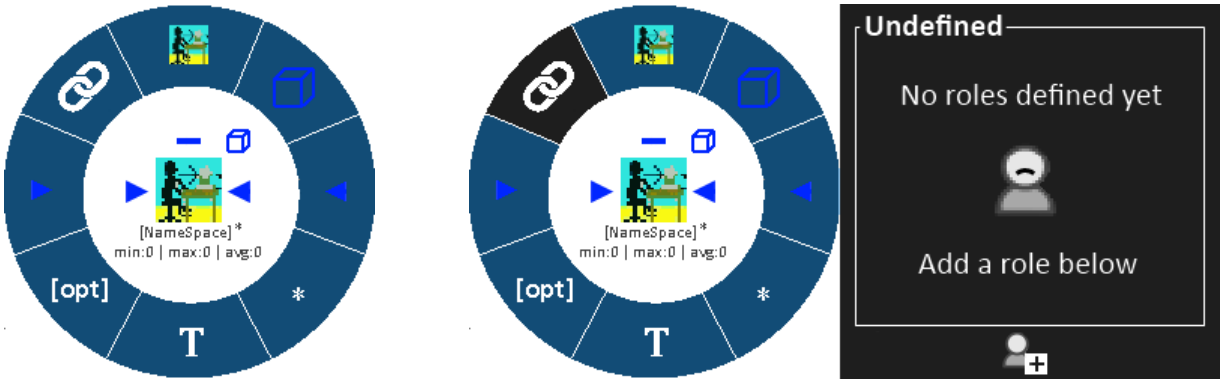


Abbildung 4.25 - Verändertes Pie-Menü im Coop Modus: Link Icon plus ConnectionTask-Liste

Auf diese Weise wird es möglich, Knoten zunächst lediglich eine Rolle zuzuweisen; ohne Kenntnis darüber, welche ConnectionTask später hinzugefügt wird. Diese Vorgehensweise hilft dabei, schon während des Erstellungsprozess des Metamodells festzulegen, welche Schritte durch welche Rolle ausgeführt werden sollen. Dazu wird die verknüpfte Rolle in Klammern unter dem Knoten in einer erweiterten, festen Zeile angezeigt – jedoch zunächst in roter Schrift, um zu signalisieren, dass noch keine Connection Task verknüpft wurde (s. Abbildung 4.26). Dadurch wird grafisches Clutter durch Überlagerung von Texten vermieden, wie es zuvor bei CTTE typisch war (vgl. bspw. Abbildung 4.20). Außerdem ist das Verkettungssymbol oben links am Knoten vorerst noch nicht aktiv. Diese Art der Darstellung schafft einen Überblick über die noch zu verknüpfenden Knoten innerhalb des Metamodells.

Abbildung 4.26 - Vorläufige Zuweisung der Rolle

Ist eine Verbindung der Rolle mit einem Diagramm hergestellt, das definierte ConnectionTasks besitzt, tauchen diese automatisch in der zuvor noch leeren Liste auf (idealerweise in Echtzeit, sollte bspw. ein anderer Teilnehmer weitere Knoten als Connection Tasks in seinem Diagramm markieren; vgl. Abbildung 4.27). Falls die Liste zu lang ist, kann darin vertikal gescrollt werden. Die Ampelsymbolik gibt wiederum an, welche Aufgaben noch nicht zugewiesen wurden und unterstützt den Benutzer somit unmittelbar bei der Auswahl. Selbstverständlich kann auch eine der grün markierten ConnectionTasks ausgewählt werden, um deren Verknüpfung mit einem anderen Knoten zu ändern.

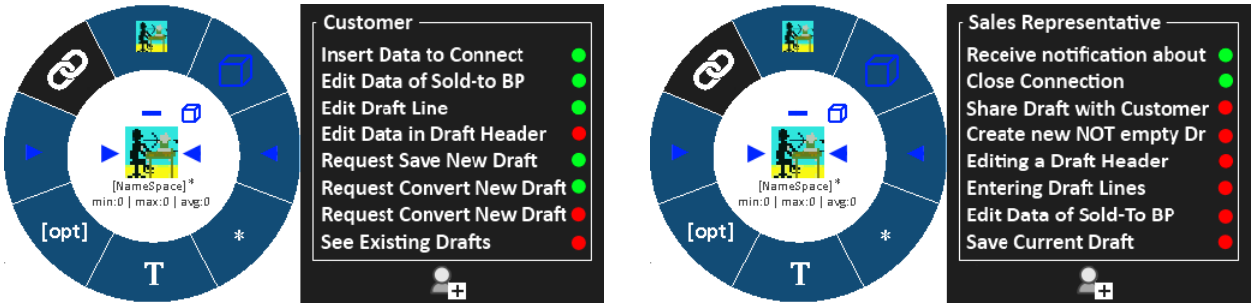


Abbildung 4.27 - Wechsel der Rolle

Am Knoten soll der Zugriff auf die Connection Tasks aller Rollen sichergestellt sein. Darum kann durch Wischen auf der erscheinenden Liste zwischen den Rollen hin und her gewechselt werden (Abbildung 4.27, rechts). Wird eine der Connection Tasks aus der Liste ausgewählt, wird am Knoten das Verknüpfungssymbol sichtbar (Abbildung 4.28, links). Die Farbe der Rollenbezeichnung unterhalb des Knotens ändert sich entsprechend auf schwarz und im Hintergrund ändert sich die Farbe der ConnectionTask-Ampelmarkierung auf grün. Die gerade getroffene Auswahl wird bei erneutem Öffnen der Liste am Knoten hervorgehoben, damit die Verlinkung nachvollzogen werden kann (rechts).

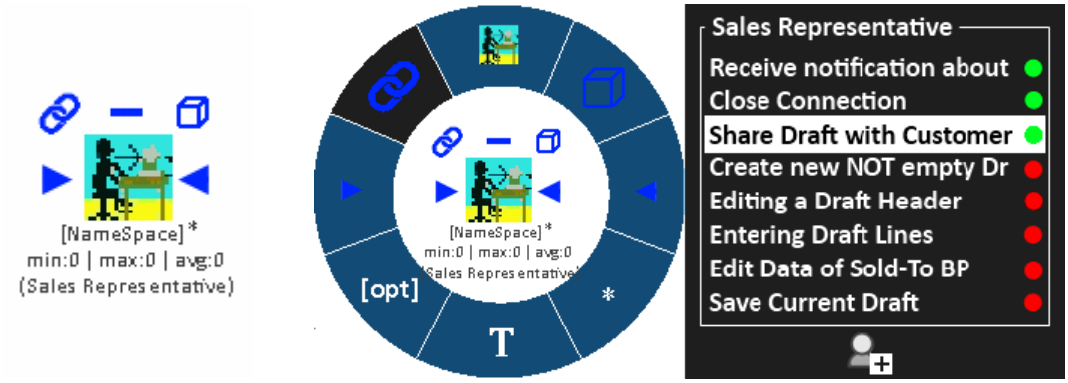


Abbildung 4.28 - Verbundene ConnectionTask & Finale Ansicht des Knotens

Darüber hinaus soll – bei Anwählen eines grün markierten Eintrags aus der ConnectionTask-Liste im Roles-Menü – direkt zu dem verknüpften Knoten im kooperativen Modell gesprungen werden können, um diesen nicht langwierig suchen zu müssen. Das kann z. B. dann hilfreich sein, wenn die Verknüpfung einer bestimmten ConnectionTask wieder gelöst werden soll. Die Verbindung kann durch erneutes Drücken der Schaltfläche im Pie-Menü wieder ausgeschaltet werden, nachdem das erste Betätigen die Liste zunächst wieder geöffnet hat. In Konsequenz werden das Verkettungssymbol oben links am Knoten sowie der Name der verknüpften Rolle schließlich wieder entfernt.

Nachdem nun detaillierte Überlegungen zum Rollenmanager vorgestellt wurden, soll es nun abschließend um Off-Screen Visualisierung gehen.

4.1.6 Off-Screen

Für die Anzeige von Objekten, die in der momentanen Ansicht des Viewports eines UML Klassendiagramms nicht verfügbar sind, haben FRISCH ET AL. eine effektive Methode der Off-Screen Visualisierung vorgestellt, die Elemente an einem interaktiven Rand sichtbar macht (s. 3.2.1.3). Die Einführung des von ihnen vorgestellten Randbereichs mit Proxys macht auch für Aufgabenmodelle Sinn, und ein Erweitern bzw. Anpassen der verwendeten Proxydarstellung liegt deshalb nahe. In einem ersten Entwurf wurde zunächst Wert darauf gelegt, dass alle potentiell vorkommenden Elemente als Proxys etabliert werden. Das Ergebnis zeigt Abbildung 4.29 in Analogie zum ursprünglichen Entwurf.

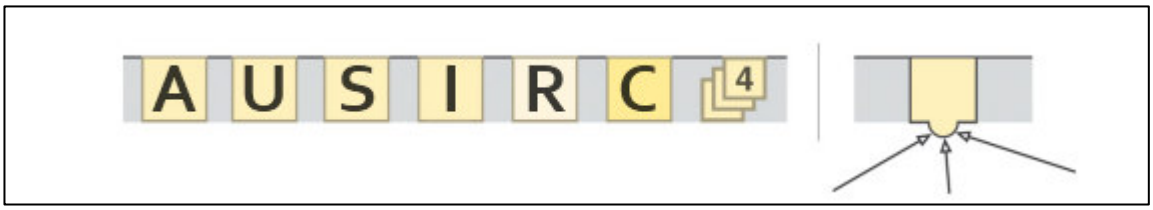


Abbildung 4.29 - Erster Entwurf der Aufgabenmodell-Proxys

Eingeführt wurden die Buchstaben *A, U, S, I* sowie *C* stellvertretend für *Abstract Task, User Task, System Task, Interaction Task* und *Cooperative Task*, die je nach Lage des Viewports allesamt am interaktiven Rand vorkommen können. Auffällig ist zudem die veränderte Farbgebung des C, da die Co-operative Task eine spezielle Rolle einnimmt und deshalb gesondert hervorgehoben werden soll (da sie bspw. auch nur im kooperativen Modus verfügbar ist). Weiterhin besticht die Einführung des Proxys mit dem Buchstaben *R*, der zu jedem Zeitpunkt die Position der Wurzel (*ROOT*) des Diagramms anzeigen soll, um so die Orientierung des Anwenders zu unterstützen. Auch dieser Proxy ist zur Abgrenzung farblich von den anderen abgestuft. Außerdem sollen wiederum gestapelte Proxys zu jedem Zeitpunkt am Viewport auf die Komplexität des Aufgabenmodells hinweisen.

All diese Überlegungen basieren jedoch noch auf der Vorlage eines UML Klassendiagramms, für das es erstens nur wenige unterscheidbare Elemente wie *Abstraction* und *Interface* gibt, zweitens aber auch keine definierte Icon Logik dieser Bezeichnungen vorhanden ist. Die Einführung der oben genannten Elemente AUSIRC hätte weiteren Lernaufwand bedeutet, obwohl mit der Verwendung der Anfangsbuchstaben des jeweiligen englischen Begriffs schon eine ausreichende Lösung gefunden worden wäre. Weil für die Aufgabenmodellierung aber bereits etablierte Icon Sets vorliegen, erscheint eine Integration dieser Icons in die Proxys als idealer Schritt. Das Ergebnis dieser Überlegung zeigt der zweite Entwurf in Abbildung 4.30 mit Hilfe der in CTTE verwendeten Symbole:

Abbildung 4.30 - Verbesserter Entwurf mit Icons

Durch diese Festlegung werden die bekannten Icons wiederverwendet, sodass seitens der Anwender nichts neu erlernt werden muss. Einzig für den Wurzelknoten wurde ein neues Symbol integriert, das sich metaphorisch passend in die vorhandene Landschaft einbettet. Bedauerlicherweise geht mit diesem Schritt die abgrenzende Farbgebung der Proxys sichtlich verloren, was aber trotzdem nicht zwingend einen Nachteil darstellt. Abbildung 4.31 zeigt anschaulich die Verwendung der Off-Screen Visualisierungstechnik im Anwendungsfall.

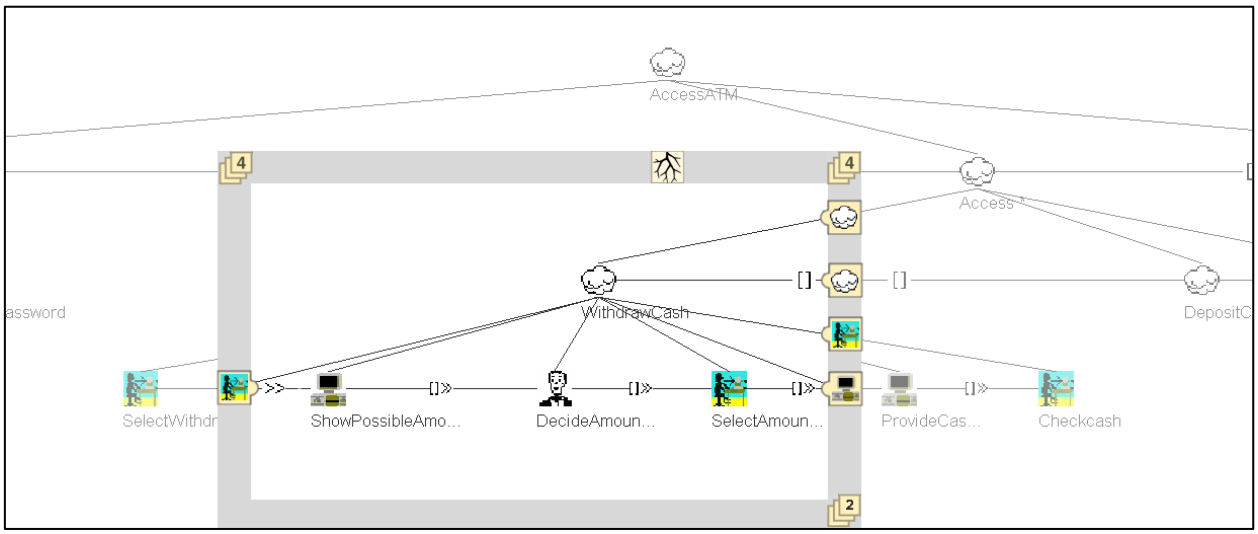


Abbildung 4.31 - Off-Screen Aufgabenmodellierung

Nachdem nun detailliert auf Aspekte der Darstellung eingegangen wurde, steht im nächsten Abschnitt die Interaktion mit dem Diagramm im Vordergrund.

4.2 Interaktion

Dieses Segment des Konzepts befasst sich intensiv mit der Interaktion von Aufgabenmodellen. Da sich deren Domäne in dieser Arbeit von einer Desktop-Anwendung auf ein Multi-Touch fähiges Display verlagert hat, spielt der Einsatz von Gestik eine übergeordnete Rolle für die Manipulation und Navigation von Aufgabenmodellen. Aus diesem Grund wird es in diesem Abschnitt vorwiegend um die Verwendung von Gesten für Aufgabenmodelle gehen (s. 4.2.1).

Ferner lassen sich Tangibles in die Interaktion beim Entwicklungsprozess einbeziehen. Diese sind greifbar und können die Gestik mit sinnvollen Metaphern und Verfahren unterstützen, sowie vielleicht die Kollaboration fördern. Daher werden in einer anschließenden Betrachtung geeignete Tangible-Methoden für die Aufgabenmodellierung identifiziert und in das Konzept integriert (s. 4.2.2).

4.2.1 Gesten

Hier sollen nun für das Konzept wichtige Gesten vorgestellt werden. Dabei wird Wert darauf gelegt, dass keine Konkurrenz zwischen bestehenden und neuen Gesten auftritt, und dass bestehende Gesten weitestgehend in ihrer Funktion aus anderen Bereichen übernommen werden, sodass kein immenser Lernaufwand bzgl. der Bedienung bei den Anwendern auftritt. Um dem gerecht zu werden, werden daher zunächst etablierte Gesten aus der Praxis beispielhaft für die Aufgabenmodellierung adaptiert und im Anschluss diese Basis durch neue Überlegungen erweitert bzw. ergänzt. Um die Funktionsweise der Bewegungen anschaulicher erklären zu können, wird in allen Grafiken auf die in Abschnitt 4.1.3 gemachten Vorschläge zur Diagrammgestaltung verzichtet und die Original CTTE Darstellung verwendet.

4.2.1.1 Etablierte Gesten

Bei der Definition von Gesten für eine neue Applikation trifft man sehr schnell auf einige Vertreter, die bereits als Standard anzusehen sind. RAINER DORAU hat in seinem Werk „Emotionales Interaktionsdesign“ die wichtigsten etablierten Gesten zusammengefasst und im Kapitel „Typische Multi-touchanwendungen“ ihre Funktionsweise definiert (Dorau 2011). Viele der im Folgenden verwendeten Grafiken stammen aus seinen Ausführungen oder wurden zur Veranschaulichung für Aufgabenmodelle angepasst. Jeder Smartphone- oder Tabletbenutzer wird die aufgezählten Varianten schon verinnerlicht haben, wenn er sein Gerät in vollem Umfang benutzen will. Aus diesem nicht zu unterschätzenden Grund sollen diese Gesten auch in ihrer Natur beibehalten und nicht verändert werden, da eine Zuweisung komplett anderer Funktionen eher Verwirrung und erhöhten Lernaufwand seitens der späteren Anwender dieses Konzepts hervorrufen würde.

Nicht zuletzt diesem Umstand ist es geschuldet, dass sich die Gesten auch in dem vorgestellten Set von FRISCH ET AL. wiedererkennen lassen. Es gilt an dieser Stelle also nicht, das „Rad neu zu erfinden“, sondern lediglich für die bereits etablierten Gesten den Bezug auf die Aufgabenmodelle herzustellen. Dazu werden die Gesten einzeln aufgegriffen und mit Beispielen auf die Aufgabendiagrammbearbeitung übertragen. Im Anschluss daran werden Überlegungen zu neuen Gesten oder einer leicht veränderten Interpretation vorgestellt (Kapitel 4.2.1.2).

Tippen

Obwohl von DORAU nicht explizit als Geste genannt, ist das Antippen mit dem Finger eine der wichtigsten Interaktionstechniken auf Multi-Touch Systemen. Das Antippen von Objekten kann zur Auswahl oder dem Markieren selbiger dienen, oder Interaktionen auf Ihnen hervorrufen. Dabei spielt auch eine Rolle, ob und wie lange der Finger auf dem Objekt verweilt (sog. *Zeitverhalten*). Damit keine Konflikte mit anderen Gesten entstehen, muss darauf besonders geachtet werden.

Abbildung 4.32 zeigt links die Tippen Geste (engl. *to tap*). Hierbei ist es entscheidend und charakterisierend, dass der Finger die Oberfläche nur kurz berührt und dann wieder weggenommen wird. Dies kann durch die Sensorik erkannt und als Event gewertet werden. Die Variante rechts zeigt das Tippen und Halten der Position (engl. *to hold*). Bei dieser Geste überschreitet die Verweildauer des Fingers einen voreingestellten Wert, sodass sie von der Tap-Geste unterschieden werden kann.

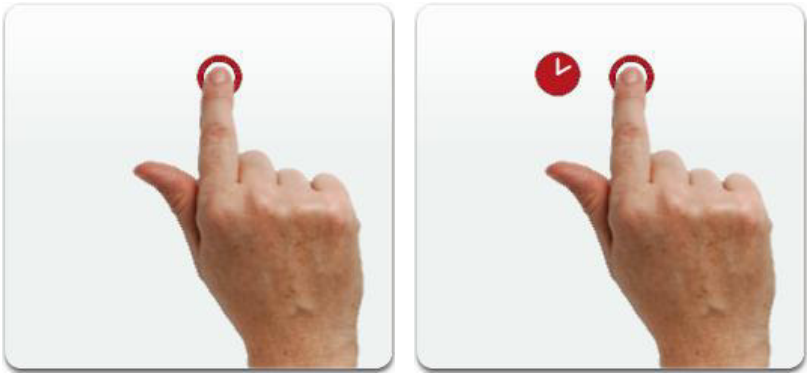


Abbildung 4.32 - TAP bzw. HOLD Geste[50]

Zum einfachen Erzeugen neuer Knoten soll die Hold Geste auf der leeren Fläche ausgeführt werden (vgl. Abbildung 4.33). Dabei sollte die Dauer des Haltens mindestens 1500 ms betragen, damit nicht von einem Tippen ausgegangen wird. Nachdem der Finger 1,5 Sekunden oder länger statisch auf der Fläche verweilt (a), soll ein halbtransparentes Icon mit Umrandung an der Position der Fingerspitze dem User Feedback darüber geben, was erstellt wird, wenn der Finger weggenommen wird.

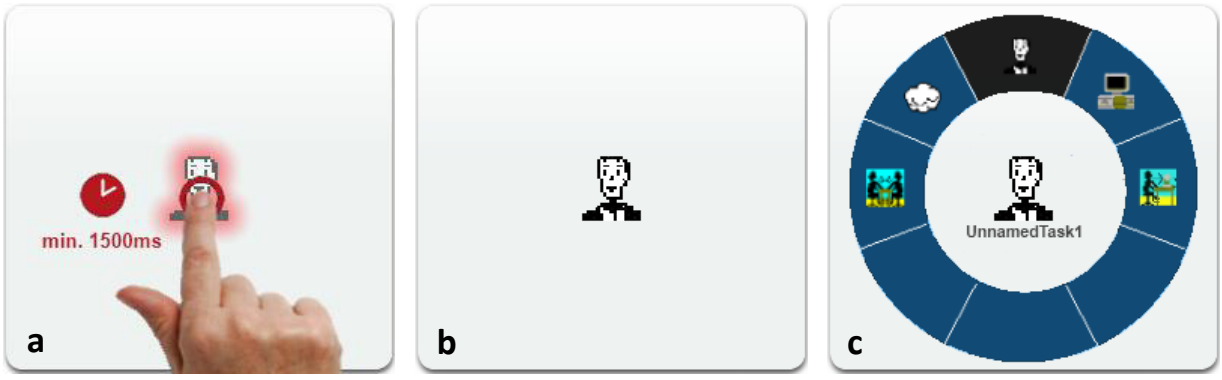


Abbildung 4.33 - Verwendung von Hold im Aufgabenmodell

Dabei soll der zuletzt verwendete Knotentyp vorausgewählt sein. Es kann entweder rein der Knoten ohne weitere Optionen angelegt werden (b), der dann durch Antippen weiter konfiguriert wird (s. Abbildung 4.34), oder der Knoten erscheint direkt mit dem Knotentyp-Auswahlmenü sowie der Bezeichnung „UnnamedTaskN“ (c), wobei N den Zähler erzeugter Knoten darstellt. Die erste Variante hat den Vorteil, dass sehr schnell viele Knoten erzeugt werden können, die jedoch alle gleich wären. Variante zwei bringt die sofortige Definition des Knoten als Vorteil mit sich, wenn diese aber zwingend durchgeführt werden muss verlangsamt es den Erstellungsprozess etwas. Aufgrund strukturierterer Vorgehensweise soll Variante zwei präferiert werden.

Wie bereits erwähnt soll einfaches Antippen eines Knotens sein kontextuelles Pie-Menü aus Abschnitt 4.1.4 aufrufen. Aus dem Menü können nun durch erneutes Antippen der entsprechenden Schaltflächen verschiedene knotenbezogene Funktionen aufgerufen werden. Im Menü kann durch Tippen auf die Mitte des Kreises zurückgegangen und die zuvor gemachten Einstellungen gespeichert werden. Wird außerhalb des Pie-Menüs auf die Fläche getippt, schließt sich das Menü, ohne die gemachten Einstellungen zu speichern.

[50] Quelle: (Dorau 2011), S. 138; Abb. 110, modifiziert

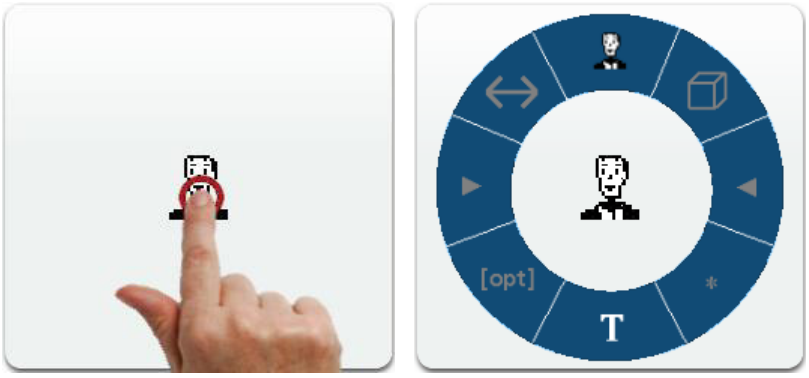


Abbildung 4.34 - Verwendung von TAP auf Knoten

Das gleiche Verhalten soll bei Antippen einer waagerechten Kante erfolgen, also einer Kante die eine temporale Relation beinhaltet (Abbildung 4.35). Da Kanten, die eine Unterordnung verkörpern, keine zeitlichen Beziehungen besitzen (und deshalb kein Icon), können diese auch nicht verändert werden und sind somit nicht antippbar. Für alle anderen Kanten öffnet das Antippen der Kante das Pie-Menü, um die Relationsart zu wechseln. Dies kann durch Tippen auf den Innenbereich des Kreises gespeichert werden.

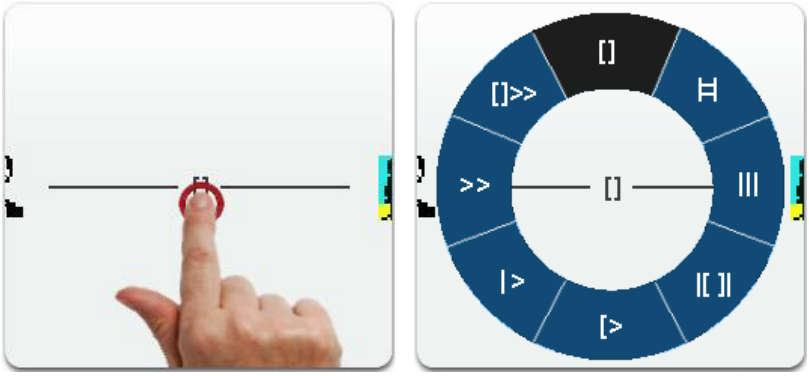


Abbildung 4.35 - Verwendung von TAP auf Kante

Ein einfaches Tippen auf die Arbeitsfläche ohne den Finger zu halten soll keine Wirkung besitzen. Umgekehrt soll ein Halten auf einem Knoten ein Verschieben initiieren (die hiernach folgende Funktion beschreibt dies genauer). Auch ein Halten auf einer Kante hat keinen gesonderten Effekt, beim Loslassen öffnet sich wie gewohnt das Knotenmenü.

Ziehen/Wischen

Die wohl bekannteste und natürlichste Geste ist das Ziehen mit der Hand (engl. *to drag*, s. Abbildung 4.36). Jedem gesunden Menschen steht diese Handbewegung uneingeschränkt zur Verfügung, und wer schon einmal kleine Kinder beim Spielen beobachtet hat, wird feststellen, dass sich diese Gestik des Ziehens und Schiebens (ebenso wie das Greifen und Heben) bereits früh entwickelt – wenn auch vielleicht noch nicht sofort mit einzelnen Fingern. Noch bevor Menschen laufen können, versuchen sie Dinge zu bewegen. Und im Laufe ihrer Entwicklung lernen sie beispielsweise, mit den Fingern Karten oder Steine bei diversen Brettspielen zu verschieben oder Gegenstände auf unterschiedlichsten Arten von Oberflächen anzuordnen; und immer geht es dabei um die Bewegung von Objekten auf einem Untergrund, oder das Bewegen des Untergrundes selbst.

Abbildung 4.36 - DRAG Geste[51]

Diese Geste soll weiterhin für sogenannte Drag&Drop Operationen vorgehalten werden, mit der Objekte an einer Stelle aufgenommen und an anderer Stelle wieder fallen gelassen werden können. Dabei soll es egal sein, ob einzelne Diagrammelemente per Finger verschoben werden (vgl. Abbildung 4.37a) oder das gesamte Diagramm selbst (vgl. Abbildung 4.37b). Der einzige Unterschied ist, dass ein Knoten nur dann verschoben wird, wenn die Bewegung durch Halten auf ihm gestartet wurde, was das Aufheben des Knotens suggeriert und ihn für die Dauer der Geste an den Finger „bindet“, bis er abgesetzt wird.

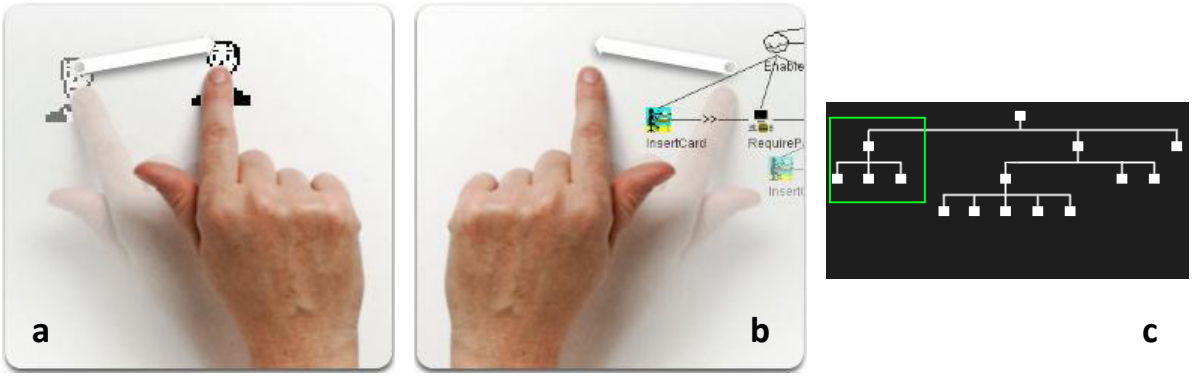


Abbildung 4.37 - Verwendung von Drag im Aufgabenmodell

Zu beachten ist beim Vorgang in Abbildung 4.37a außerdem, dass beim Verschieben von Knoten die jeweiligen verbundenen Kinder 1:1 an die neue Position mit verschoben werden. Dies ist deshalb zu bevorzugen, weil Objekte sowieso fest in ihrer Baumstruktur angeordnet werden. Ein einzelnes Verschieben des Elternknotens ohne Anpassung der Kinder würde diesen Zustand zerstören. Für (b) wäre es außerdem hilfreich, für die Dauer der Bewegung eine Miniaturkarte einzublenden (c), die Orientierung verschafft. Ein grüner Kasten zeigt darin den momentanen Bildausschnitt an.

Scrollen

Die Scroll-Geste ist mit ihrer Zwei-Finger Technik vorwiegend aus der Welt der Macs der Firma Apple bekannt. Auf den Trackpads von Mac Notebooks[52] sowie der Magic Mouse lässt sich das Blättern durch Dokumente und Webseiten standardmäßig nur durch das vollführen der Drag-Geste mit zwei Fingern bewerkstelligen (Touchpads anderer Hersteller bieten dafür spezielle Randbereiche der Fläche an). Dabei scrollt die vertikal ausgeführte Geste im Dokument selbst, während sie vertikal ausgeführt zwischen geöffneten Dokumenten wechselt. Man kann sich Scrollen wie das Abrollen einer Papierrolle vorstellen; führt man die Bewegung sehr schnell aus, gelangt man in kurzer Zeit ans Ende der Rolle. Auf die Frage, warum die Ein-Finger Variante dafür nicht ausreicht, soll folgendes als Antwort dienen.

[51] Quelle: (Dorau 2011), S. 138; Abb. 110
[52] Quelle: Apple http://support.apple.com/kb/HT3448

Abbildung 4.38 - 2-Finger DRAG Geste[53]

Im Vergleich zur Ein-Finger Variante eröffnen sich mit dieser Version neue Möglichkeiten der Abgrenzung. So kann sie im Falle komplexer Aufgabenmodelle dazu dienen, sich schnell innerhalb des Diagramms zu bewegen und verschiedene Enden des Raumes nach nur kurzer Verzögerung zu erreichen. Viel besser aber noch lässt sich mit dieser Geste – genauso wie durch Dokumente – auch durch unterschiedliche Aufgabenmodelle innerhalb eines Projekts blättern; vor allem wenn es sich bspw. bei dem Aufgabenmodell um ein kooperatives handelt, bei dem bereits Rollen definiert und zugewiesen wurden. Auf diese Weise könnte im Single-User Betrieb mit dieser einfachen Geste in horizontaler Ausführungsrichtung ad hoc durch die verschiedenen Instanzen (Diag1, Diag2 … DiagN) sowie bereits mit Rollen verknüpfte Diagramme (Role1, etc.) und den Rollenmanager (RoleMan) geblättert werden, ohne dass dafür eine Schaltfläche verwendet werden müsste (s. Abbildung 4.39).

Abbildung 4.39 - Verwendung von SCROLL im Aufgabenmodell

Das momentan auf dem Bildschirm angezeigte Diagramm ist dabei in der eingeblendeten Vorschau hervorgehoben. Die durch das Scrollen neu ausgewählte Instanz wird textuell mit Hilfe der vier Marker (im Bild in der Mitte) markiert. Solange die Finger noch nicht abgesetzt sind, wechselt diese Markierung immer weiter bis zur gewünschten Position, solange die Hand bewegt wird. Am Ende (z. B. beim Rollenmanager) soll wieder von vorne begonnen werden (in diesem Fall Diag1 bzw. Role1), sodass durchgängig immer weiter gescrollt werden kann. Die Bewegung funktioniert dabei entgegengesetzt in beide Richtungen. Lässt man los, wird das durch die Markierung ausgewählte Diagramm geöffnet.

Im Multi-User Fall mit geteiltem Bildschirm können mehrere Instanzen von in Bearbeitung befindlichen Diagrammen existieren. Durch Wischen mit zwei Fingern kann so durch verschiedene geöffnete Diagramme gescrollt werden. Sind diese bereits durch den Rollenmanager mit einer Rolle verknüpft, kann wieder der Name der Rolle mit angezeigt werden (Role1, Role2, …), ansonsten weiterhin „Diag1..N“ mit entsprechender hochlaufender Zahl.

[53] Quelle: (Dorau 2011), S. 139; Abb. 111

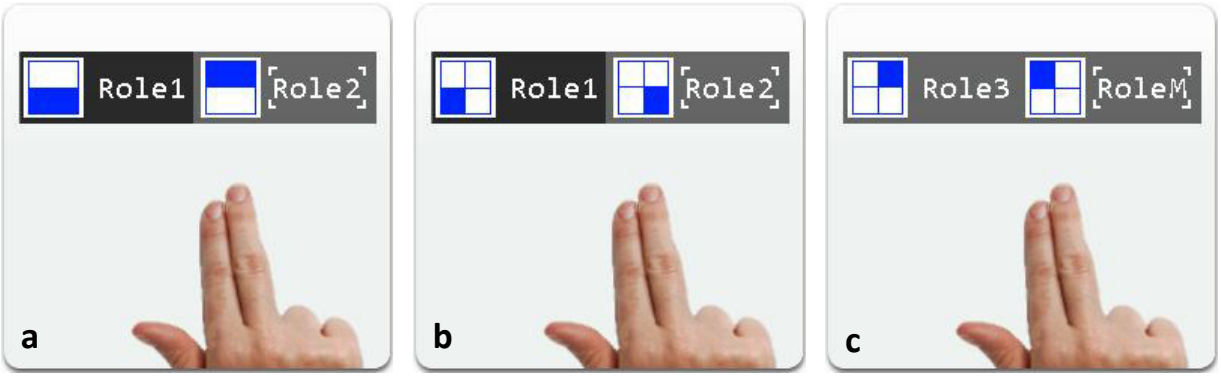


Abbildung 4.40 - Scroll bei Multi-User Anwendung

Zur Orientierung, welche der vorhandenen Diagramme sich derzeit in Bearbeitung befinden, kann deren Position auf dem geteilten Tabletop in der Vorschau angezeigt werden (vgl. Abbildung 4.40). Dabei ist eine Berücksichtigung der Teilung von Bedeutung: Im zweigeteilten Fall stehen sich die Personen gegenüber (a), im viergeteilten Fall können auch diagonale oder nebengelagerte Positionen entstehen (b&c). Ein Wechsel auf diese Instanzen ist in diesem Fall dann nicht möglich, weil sie sich gerade in Bearbeitung befinden. Falls weitere Diagramme oder Rollen existieren, oder zum Beispiel der Rollenmanager gerade auf keinem der Bildschirme geöffnet ist, werden diese Instanzen innerhalb des Balkenmenüs ohne entsprechende Grafik versehen, und stehen deshalb für die Auswahl zur Verfügung.

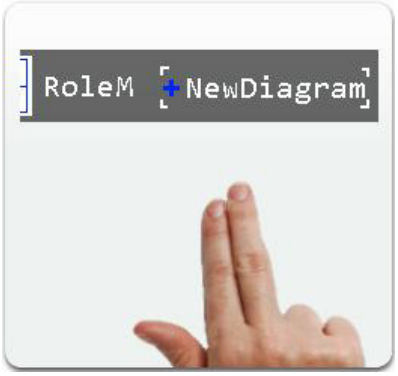


Abbildung 4.41 - Scroll: Neues Diagramm

Denkbar wäre auch, durch diese Geste ein neues Diagramm über das Balkenmenü zu erzeugen. Dazu wählt man den Eintrag aus dem Balkenmenü per Scroll aus und erhält eine neue leere Fläche zum Arbeiten (Abbildung 4.41). Dabei muss sichergestellt sein, dass das zuvor bearbeitete Diagramm gespeichert wird und für späteres Zusammenlegen, Zuweisen oder erneutes Bearbeiten erhalten bleibt. Dieser Eintrag zur Diagrammerzeugung sollte grundsätzlich immer erreicht werden können, da es oftmals von Vorteil sein kann, spontane Ideen unbeeinflusst auf einer neuen, leeren Grundlage zu entfalten – so als wenn man ein neues Blatt Papier zur Hand nehmen würde.

Zoomen

Die Spreizgeste (engl. *to pinch:* klemmen) ist eine der repräsentativsten der Multi-Touch Generation. Sie ist vorwiegend aus dem Bereich der Kartennavigation bekannt, und vollführt durch Aufziehen des Daumens und Zeigefingers einer Hand eine Vergrößerung (Zoom-In) des angezeigten Kartenausschnitts bzw. durch Zuziehen der beiden Finger eine Verkleinerung (Zoom-Out). Auf diese Weise werden in beide Richtungen neue Informationen zugänglich gemacht: Im ersten Fall werden detailliertere Informationen eines Bereichs ermittelt, in Gegenrichtung werden neue Gebiete außerhalb des ursprünglichen Bereiches erschlossen und man erhält einen besseren Überblick.

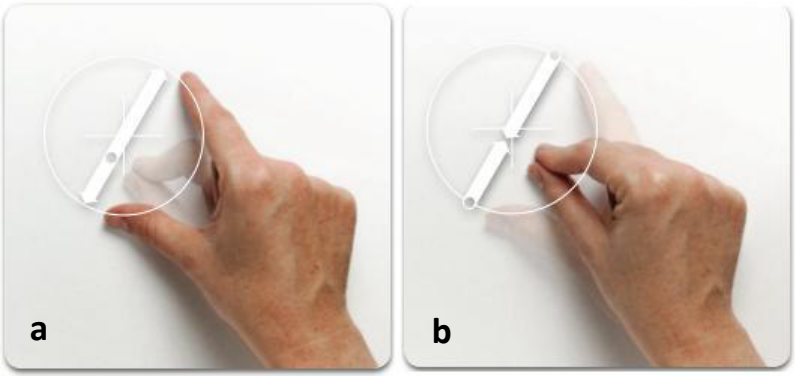


Abbildung 4.42 - PINCH Geste[54]

Diese Handbewegung lässt sich natürlich nicht nur für kartenbasierte Anwendungen benutzen und eignet sich deshalb hervorragend für die gleichen Operationen auf einem Diagramm. Vorteil dieser Geste ist vor allen Dingen der direkte Bezugspunkt des Zooms, was bei den untersuchten Editoren aus dem Desktopbereich ja bisher unbefriedigend gelöst wurde, da der Fokus beim Vergrößern ständig verloren ging. Diese Geste verhindert dieses Problem des Fokusverlusts, und kann mehrfach hintereinander ausgeführt werden, um immer tiefer in die Ebene hinein oder aus ihr heraus zu gehen. Jedoch muss dabei die Verwendung der Zoom Funktion nach oben und unten beschränkt werden, da die Abstraktion in beiden Richtungen ab einem gewissen Grad die Orientierung mindert anstatt sie zu verbessern. Bei zu starker Verkleinerung schwindet die Leserlichkeit ebenso sehr wie bei zu starker Vergrößerung.

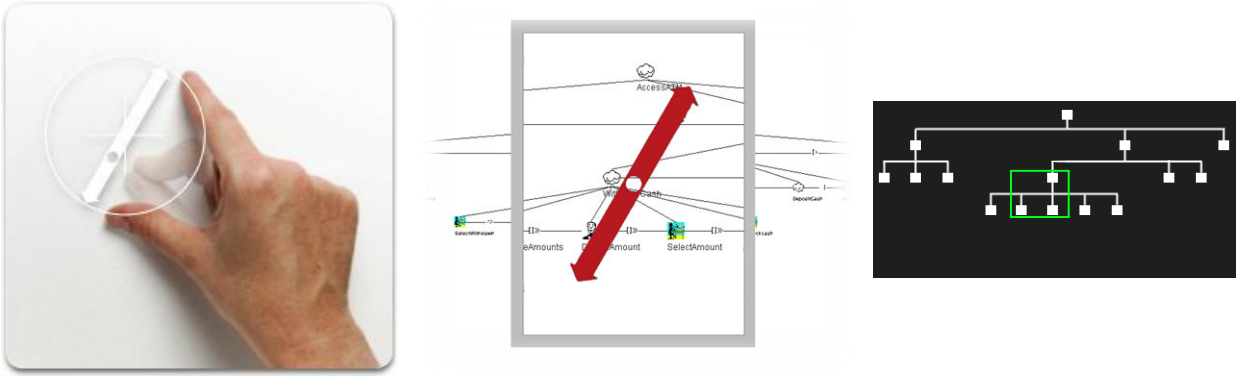


Abbildung 4.43 - Verwendung von Pinch im Aufgabenmodell (Zoom In)

Beim Aufziehen der beiden Finger soll der Bildausschnitt an dem Punkt vergrößert werden, von dem aus die Geste begonnen wurde; also dort, wo die beiden Finger zu Beginn nah zusammen waren (s. Abbildung 4.43, links). Der Bildausschnitt soll sich proportional zur Bewegung der Finger vergrößern, bis die Oberfläche losgelassen wird. Nach und nach werden nur noch wenige Elemente dargestellt, aber in detaillierterer Form. Ab einem bestimmten Vergrößerungsfaktor wird nicht mehr weiter hineingezoomt, um einen sinnvollen Arbeitsbereich zu gewährleisten (welchen Vorteil hätte es beispielsweise, nur noch ein Icon auf dem Bildschirm sehen zu können?). Auch hierbei ist wieder die Miniaturkarte hilfreich, wenn sie während des Vorgangs eingeblendet wird. Der grüne Kasten darin muss sich proportional immer weiter verkleinern, weil durch das Zoomen die Größe des Bildausschnittes stetig abnimmt.

[54] Quelle: (Dorau 2011), S. 145; Abb. 118 {a}, Abb. 119 {b}

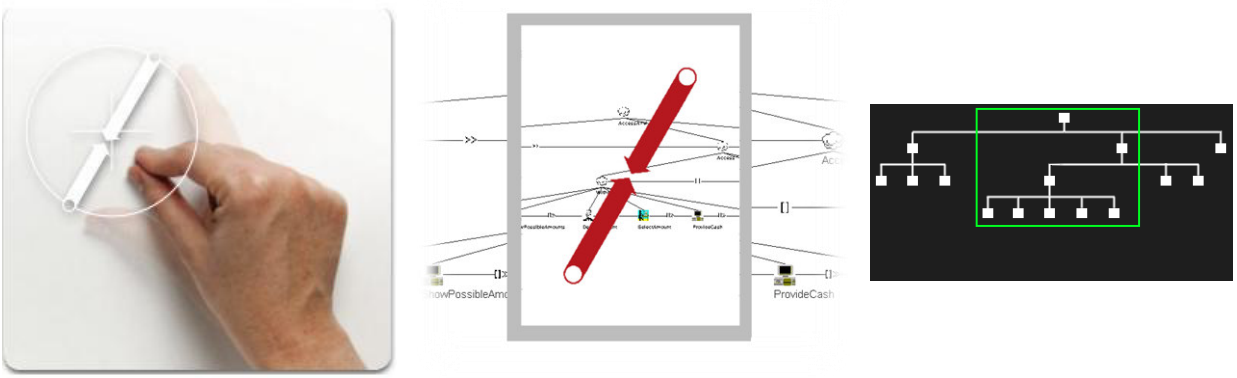


Abbildung 4.44 - Verwendung von Pinch im Aufgabenmodell (Zoom Out)

In entgegengesetzter Richtung der Fingerbewegung soll sich die Ansicht des Modells proportional verkleinern, sodass mehr Elemente und deren Zusammenhänge sichtbar werden – und dass ebenso innerhalb des Bereichs, in dem beide Finger die Oberfläche initial berührt haben (vgl. Abbildung 4.44, links). Dieser Überblick kann solange weiter ausgebaut werden, bis das gesamte Diagramm sichtbar ist bzw. man die Knotennamen noch weiterhin lesen kann. Sind in dieser Zoomstufe die horizontalen und vertikalen Dimensionen des Bildschirms aufgrund der Komplexität des Modells nicht mehr zur Anzeige ausreichend, kann nicht das gesamte Modell abgebildet werden und muss zur Exploration per Drag verschoben werden. In der Miniaturansicht wird der grüne Kasten wieder entsprechend vergrößert.

Drehen

Eine weitere intuitiv auszuführende Geste ist das Drehen. Sie kann entweder dafür eingesetzt werden, einzelne Objekte zu drehen, nachdem diese ausgewählt wurden, oder alternativ die gesamte Arbeitsfläche, wenn sie im freien Raum ausgeführt wird. Dabei lässt sich die Spin-Operation auf zwei Arten ausführen, wie Abbildung 4.45 demonstriert. Einerseits kann man eine kreisförmige Bewegung ausführen, indem Daumen und Zeigefinger einer Hand in entgegengesetzter Richtung simultan eine gekrümmte Linie formen. Andererseits kann man alternativ eine Kreisbewegung mit einem Finger vollführen, um die gleiche Reaktion hervorzurufen.

Abbildung 4.45 - SPIN Geste[55]

Bei letzterer Variante des Spin besteht jedoch das Problem, dass sich diese sehr ähnlich zu der Drag Operation verhält, was die Entscheidung, ob in Echtzeit gedreht oder verschoben werden soll, sehr erschwert. Diese Geste entstammt einem Zeitpunkt aus der Vergangenheit, zu dem die Technologie noch keine Multi-Touch-Unterstützung bot. Man könnte diese Gesten getrennt verwenden, und zwar in

[55] Quelle: (Dorau 2011), S. 149-150; Abb. 123 {a}, Abb. 124 {b}

der Art, dass mit der Zwei-Finger Variante stets die Arbeitsfläche gedreht wird und mit der Ein-Finger Variante nur Objekte gedreht werden können. Dies sollte jedoch nur nach vorherigem Auswählen möglich sein, und die Drehbewegung nicht über das Objekt hinausgehen, um den Spin von einem herkömmlichen Drag&Drop zu unterscheiden. Da ein Verdrehen einzelner Knoten in einem Aufgabenmodell nicht unbedingt sinnvoll erscheint, kann auf diese Technik auch ganz verzichtet werden, damit sie den Anwender schlussendlich nicht auch noch zusätzlich belastet.

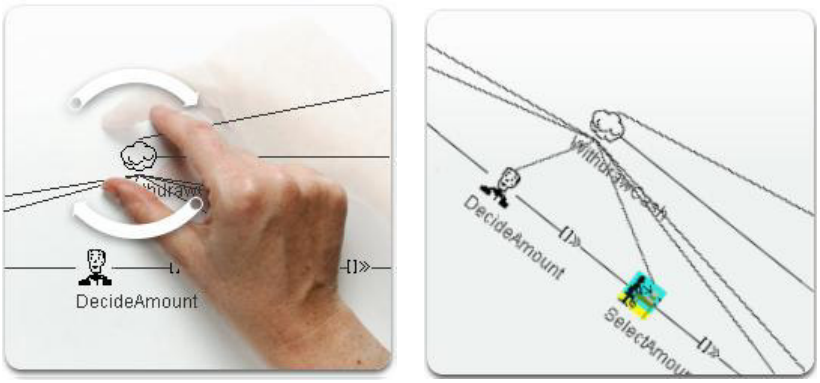


Abbildung 4.46 - Verwendung von Spin im Aufgabenmodell

Beschränkt man sich lediglich auf das Drehen der Arbeitsfläche, ist der einzige offensichtliche Fall sinnvoller Anwendung das Präsentieren des Aufgabenmodells für andere Personen, die sich in einer anderen Ausgangsposition als der Agierende befinden. Aus diesem Grund soll sich die Drehgeste allein auf dieses Szenario beschränken. Sinnvoll wäre hierbei ein kurzes Einrasten während der Drehbewegung entlang der rechten Winkel $\{0°, 90°, 180°, 270°\}$, um die Sicht an wichtigen und sinnvollen Positionen leichter zu fixieren.

4.2.1.2 Uminterpretierte und neue Gesten

Nachdem im vorherigen Abschnitt bekannte Standardgesten für die Verwendung auf Aufgabenmodellen interpretiert wurden, soll es in diesem Bereich um Neuschöpfungen speziell für diesen Themenbereich gehen. Um dies zu erreichen, werden bereits definierte Gesten in einen neuen Kontext gesetzt, oder von Grund auf neu entwickelt. Dieser Vorgang ist derart komplex, dass DORAU ihm ein beachtlich umfangreiches Kapitel gewidmet hat (Dorau 2011). Es wird nun versucht, bei der Gestaltung auf die dort vorgeschlagenen Aspekte zu achten.

Kanten erstellen

Für das Anlegen von Kanten zwischen zwei Knoten ist eine Zwei-Finger resp. Zwei-Hand Geste kaum vermeidbar. Um Verbindungen herzustellen muss es zwangsläufig einen Start- und einen Zielpunkt geben, an dem sich orientiert werden kann. Aus diesem Grund soll das Antippen zweier Knoten dieser Zielsuche entsprechen und die zu verbindenden Knoten selektieren, zwischen denen eine Kante erzeugt werden soll (Abbildung 4.47).

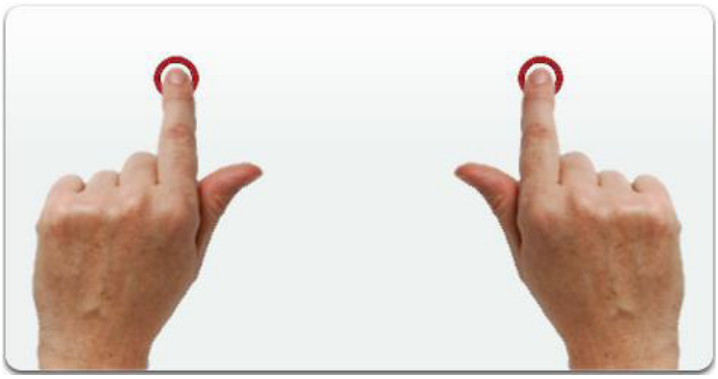


Abbildung 4.47 - 2-Hand TAP Geste[56]

Es ist hierbei egal, welcher der Knoten zuerst angetippt wird. Sobald zwei Knoten nahezu gleichzeitig angetippt werden, wird die Kante erstellt. Alternativ kann aber auch, solange einer der beiden Knoten mit dem Finger der einen Hand gehalten wird, durch Tippen mit der anderen Hand das Ziel der Kante bestimmt werden. Die anzulegende Kante kann bspw. in Rot vorgeblendet werden, solange der erste Knoten noch gehalten wird (vgl. Abbildung 4.48). Auf diese Weise ist es auch weiterhin möglich, die Kante noch zu verändern, in dem mit der zweiten Hand ein anderer Knoten angetippt wird, solange die erste Hand noch nicht freigegeben wurde (die rote Kante ändert sich dementsprechend). Lässt die erste Hand los, wird die Kante gezeichnet und wechselt ihre Linienfarbe auf schwarz.

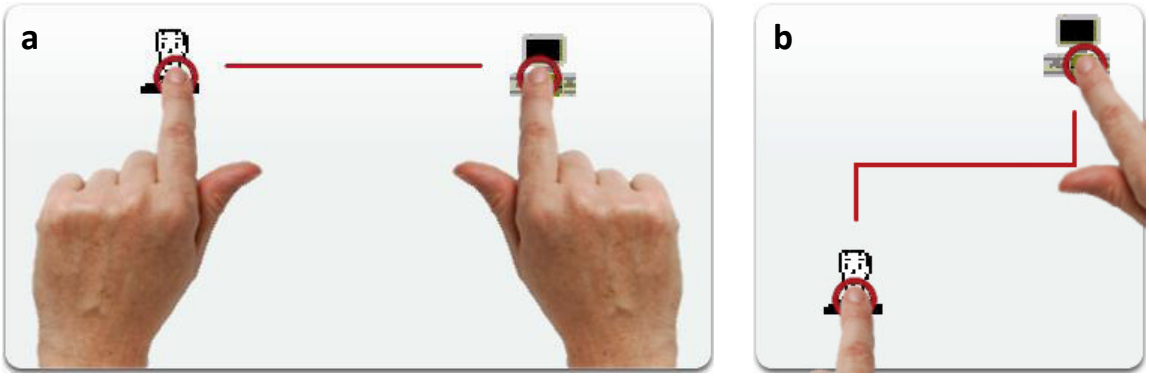


Abbildung 4.48 - Verwendung von 2-Hand Tap im Aufgabenmodell

Ob eine Zwischenkante mit temporaler Relation eingefügt werden soll (a) oder ein Knoten dem anderen untergeordnet wird (b), soll allein von der Position der Knoten abhängig sein. Berühren sich die Pixel der Knoten in der horizontalen Ebene noch, wird ein waagerechter Verbinder eingefügt, bei dem anschließend das Pie-Menü zur Wahl der zeitlichen Beziehung angezeigt wird (a); anderenfalls wird das geographisch weiter südlich liegende Symbol dem anderen untergeordnet (b). In beiden Fällen sollen die Knoten automatisch in eine skalierende, aber feste Baumstruktur eingebunden werden (s. Kapitel 4.1.3). Ebenfalls sollen sich auf diese Art ganze Teilbäume anderen Knoten unterordnen lassen. Ob der Anwender eine Unter- oder Nebenordnung wünscht, kann er durch vorheriges Verschieben der Knoten mit der Drag-Geste selbst bestimmen.

Auf-/Zuklappen eines Knotens

Um kurz- oder langfristig Übersicht im Modell herzustellen, kann es durchaus sehr sinnvoll sein, einzelne innere Knoten eines Baumes oder ganze Äste vorübergehend einzuklappen, um eine abstraktere Sicht auf die Aufgabe zu bekommen. Das Öffnen oder Schließen von inneren Knoten ließe sich naheliegender

[56] Quelle: (Dorau 2011), S. 138; Abb. 110, modifiziert und gespiegelt

Weise durch die Pinch-Geste vollziehen, indem der Knoten zusammengeklemmt wird. Da diese jedoch schon für das Zoomen reserviert ist, wird eine metaphorisch passende Geste eingeführt – das Zusammenziehen aller Finger einer Hand (s. Abbildung 4.49):

Abbildung 4.49 - Hand öffnen/schließen (GRAB)[57]

Nach DORAU lässt sich die Geste des Schließens einer Hand durch alle fünf Finger mit dem Zusammenfassen mehrerer Objekte assoziieren (Erstellen von Stapeln etc.). Alternativ nennt er aber auch das Verbergen von Objekten innerhalb der Hand, was ideal für das Einklappen innerer Knoten eines Baumes wäre. Führt man diese Geste auf einem Knoten aus, werden alle seine Kind-Elemente „im Knoten“ versteckt. Die entgegengesetzte Gestik, das Öffnen der Hand, gibt in Analogie die gesamten versteckten Objekte wieder preis.

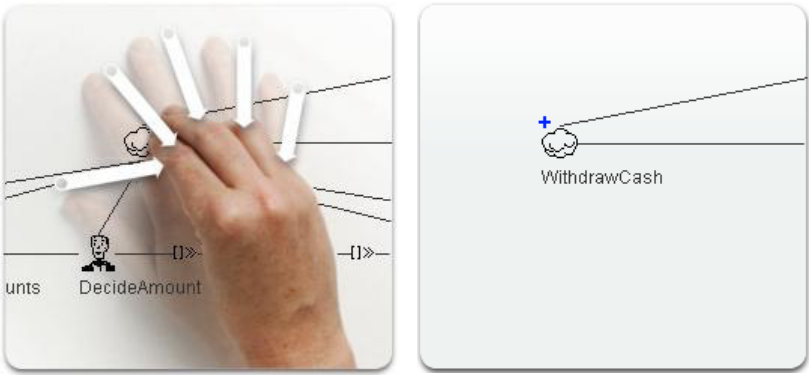


Abbildung 4.50 - Verwendung von Grab im Aufgabenmodell

Um den Anwender visuell zu unterstützen, muss eine Symbolik an diese Geste angeheftet werden. Das etablierte Pluszeichen für das Erweitern von Objekten, das aus Ordnerstrukturen oder auch Webanwendungen bekannt ist, sollte am eingeklappten Knoten als Markierung dienen und dem Benutzer helfen, eingeklappte Knoten zu erkennen. Es wurde auch bereits in CTTE für eingeklappte Knoten verwendet (und bereits in Kapitel 4.1.3 am Knoten integriert), das Auswählen des Einklappens konnte jedoch nicht direkt am Knoten durchgeführt werden. Diese Lücke wird durch die neue Geste geschlossen. Auf Knoten, die keine Kinder besitzen, soll die Geste keinen Einfluss haben – an diesen wird aber auch das Minus Symbol bis zur Unterordnung von Kinderknoten nicht angezeigt.

Entfernen von Knoten und Kanten

Das Durchstreichen oder Ausradieren von Dingen bei erkannten Fehlern oder nötigen Änderungen ist ein natürlicher und naheliegender Vorgang beim Skizzieren. DORAU und FRISCH ET AL. haben jeweils in ihren Auflistungen unabhängig voneinander auf diese Geste verwiesen. Die Schwierigkeit besteht dabei

[57] Quelle: (Dorau 2011), S. 153; Abb. 130

darin, die ausgeführte Bewegung von einem Drag zu unterscheiden. Nichtsdestotrotz kann diese Gestik dazu verwendet werden, um Knoten oder Kanten aus dem Diagramm zu entfernen.

Abbildung 4.51 - ERASE Geste[58]

Um ein Löschen zu beginnen, fängt man in der Nähe des zu entfernenden Objektes an, eine Bewegung auszuführen, als wenn man dieses Objektes wie auf einem Blatt Papier mit dem Finger ausradieren wollte. Während der Finger auf der Oberfläche verweilt kann beim Objekt als visuelles Feedback der Alphawert herabgesetzt werden, was es transparent erscheinen lässt. So kann man direkt sehen, was nach dem Loslassen des Fingers gelöscht wird. Nach Abschließen der Geste sollte das ausradierte Objekt auch unmittelbar verschwinden.

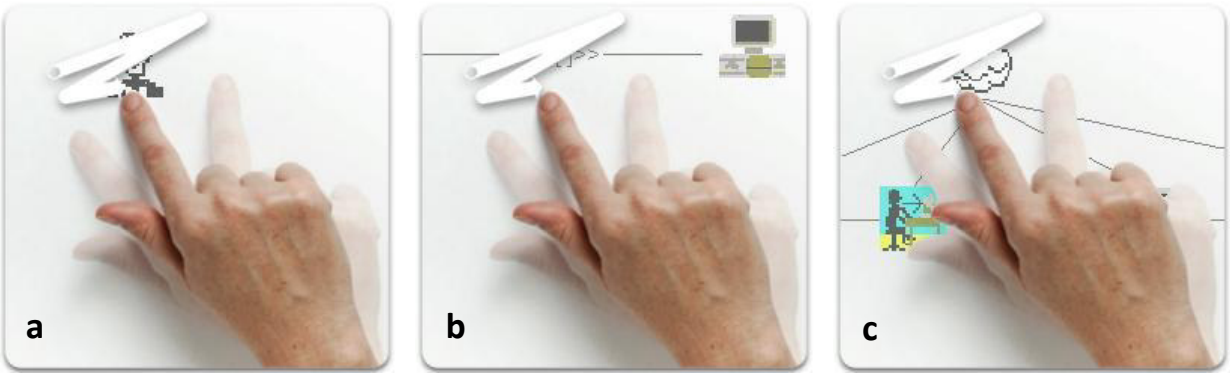


Abbildung 4.52 - Verwendung von ERASE im Aufgabenmodell

Die Geste soll sowohl auf Knoten (Abbildung 4.52a) als auch auf Kanten (Abbildung 4.52b) durchführbar sein. Die Bewegung lässt sich sinngemäß auch mit längeren Pfaden über mehrere Objekte gleichzeitig ausführen, die dann zusammen gelöscht werden. Wichtig ist, dass konsequenterweise beim Entfernen eines Elternknotens die zugehörigen Kinderknoten und Kanten mit entfernt werden, damit keine unbeabsichtigten Reste übrig bleiben. Man könnte argumentieren, dass dies ungewollt ganze Bäume vernichten kann, jedoch kann der Wurzelknoten eines Teilbaums einfacher über das Menü geändert werden, als ihn zu löschen und neu einzufügen. Außerdem ließe sich mit dieser Vorgabe das ganze Diagramm durch Ausradieren der obersten Wurzel komplett in einem Schritt leeren (sofern alle Knoten zusammenhängen), wenn man neu beginnen möchte.

Rückgängig

Von essentieller Wichtigkeit ist in vielen Softwareprogrammen eine Funktion, die die zuletzt durchgeführten Schritte wieder rückgängig macht. Diese Funktion kann sehr viel Arbeit ersparen, sollte man einmal unabsichtlich beispielsweise das gesamte Diagramm oder auch nur einen komplexen

[58] Quelle: (Dorau 2011), S. 160; Abb. 136

Teilbaum gelöscht haben. Die einzige Alternative bei diesem Szenario ohne eine klassische „Undo“ Funktion wäre das komplette erneute Anlegen der versehentlich entfernten Elemente. Um keine Konflikte mit anderen Gesten zu erhalten, wird eine Berührung eingeführt, bei der zunächst die drei mittleren Finger einer Hand benutzt werden (Abbildung 4.53a).

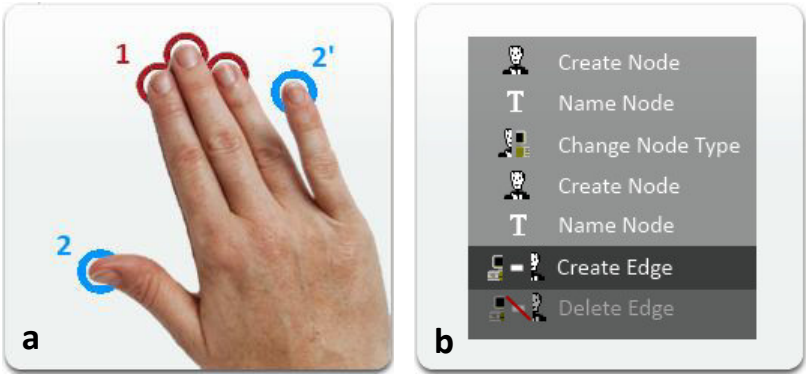


Abbildung 4.53 - 3-Finger HOLD + TAP der verbliebenen Finger | Undo Stack[59]

Ein Hold mit drei Fingern initiiert die Geste (1). Während die drei Finger halten, stellt das Tippen mit dem Daumen – kontinuierlich ausgeführt – sukzessive die vorangegangenen Zustände des Diagramms wieder her (2). Durch das Tippen ist sichergestellt, dass mehrere Rückschritte schnell hintereinander ausgeführt werden können. Im Gegensatz dazu soll das Tippen mit dem kleinen Finger, während die drei Finger weiterhin halten, den zuletzt zurückgenommenen Schritt wiederholen (2‘). Dies wurde bewusst so gewählt, um die Analogie beim Ausführen der Geste zu bewahren, gleichzeitig aber die schwieriger durchzuführende Bewegung auf die Funktion zu legen, die weniger häufig Benutzung findet. Zur besseren Übersicht sollte der Undo-Stack, also der Stapel auf dem die zuletzt getätigten Operationen gespeichert sind, zumindest temporär für den Anwender sichtbar sein (zum Beispiel in Form einer Liste, die am Rand eingeblendet wird, sobald die Geste ausgeführt wird, s. Abbildung 4.53b). Auf diese Weise kann man sich orientieren, wie viele Schritte man zurückgehen möchte und direkt entscheiden, ob die Geste erneut praktiziert wird.

Trennen und Zusammenlegen von Arbeitsbereichen

Speziell beim gemeinsamen Arbeiten kann zwischenzeitlich (oder auch dauerhaft) die Anforderung entstehen, die Aufgabenbereiche der Anwender aufzuteilen oder zusammenzulegen. Zu diesem Zweck soll eine Geste erstellt werden, die sich analog zu realen Maßstäben verhält und in ihrer Funktion offensichtlich ist. Vorgeschlagen wird hier deshalb das Auftrennen und Auseinanderschieben des Bildschirmes mit zwei Händen. Alle 10 Finger beider Hände berühren dabei die Oberfläche und sind zu Beginn der Geste nah beieinander (vgl. Abbildung 4.54).

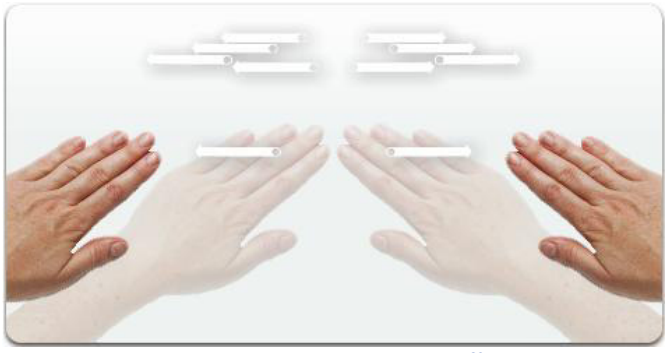


Abbildung 4.54 - TEAR-Geste[60]

[59] Quelle: (Dorau 2011), S. 141; Abb. 113, modifiziert

Durch das voneinander Wegbewegen der Hände nach außen entsteht eine Bewegung, die sehr stark an das Auseinanderreißen eines Blatt Papiers erinnert (noch stärker, wenn das Blatt perforierte Trennlinien aufweist). Darum wurde der Name TEAR für die Geste gewählt (engl. *to tear (apart)*) Während des Auftrennens des sichtbaren Bereichs soll ab einer bestimmten Schwelle zwischen den beiden Händen die Wirkung angedeutet werden (s. Abbildung 4.55), indem die zur Mitte gerichteten Ränder beider neuen Bereiche wie ein Riss von Papier aussehen. Auf beiden Hälften wird der momentane Bildschirmausschnitt angezeigt, und je weiter die Hände auseinander gezogen werden, desto größer wird der Leerbereich der Mitte. Werden die beiden Hände vor dem Loslassen wieder zusammengeschoben, erfolgt keine Auftrennung. Ebenso können durch entgegensetzte Gestik auch aufgetrennte Bildschirme auf die gleiche Weise wieder vereint werden. Für beide Fälle muss aber noch festgelegt werden, welches Diagramm dann auf dem neuen Bildschirm angezeigt wird (folgt etwas später in diesem Abschnitt, auch Teil von Kapitel 4.3).

Für den Trennvorgang ist es wichtig, dass man eine Orientierung darüber erhält, wie das Diagramm geteilt wird. Grundsätzlich erfolgt die Auftrennung an einer Linie zentriert zwischen den beiden Händen, wie man es in Abbildung 4.55 oben deutlich sehen kann. Problematisch dabei ist, dass auf diese Weise Äste zusammenhängende Bäume gekappt werden können, wenn diese sich auf beiden Seiten der Trennung befinden.

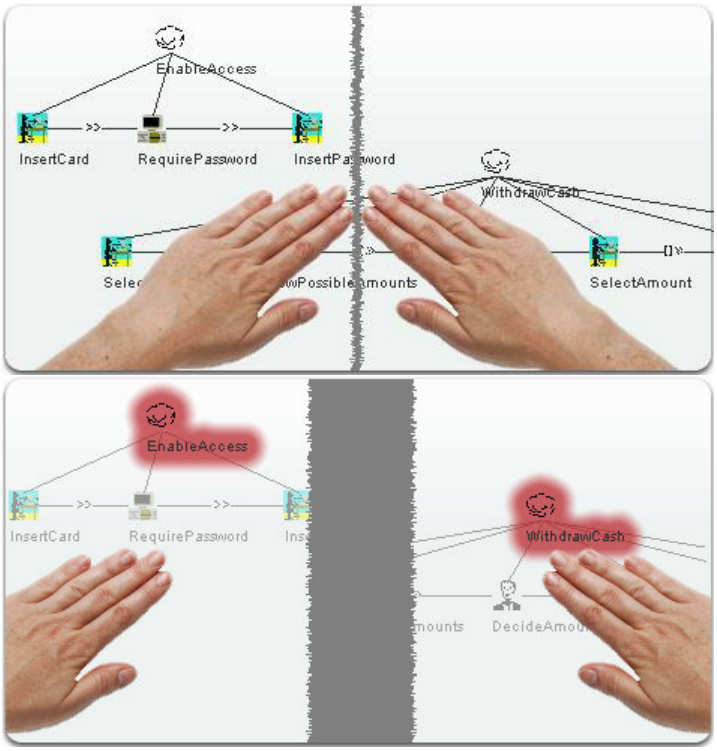


Abbildung 4.55 - Anwendung von TEAR im Aufgabenmodell

Um dieses Verhalten zu unterbinden, sollen Teilbäume stets anhand ihrer Wurzel aufgeteilt werden. Das bedeutet, dass beim Trennen die Kinder dieser Teilbäume vernachlässigbar sind und somit verblassen, während ihre Wurzeln hervorgehoben werden, um zu signalisieren, welche der Bäume mitsamt ihrer Kinder auf welche Seite der Teilung wechseln (s. Abbildung 4.55 unten). Für einzelne Knoten ist dies unproblematischer, da ihr resultierender Platz anhand der Trennlinie direkt sichtbar wird; sollen sie aber nach der Trennung auf einem bestimmten Bildschirmteil verfügbar sein, müssen sie vorher entsprechend verschoben werden.

---

[60] Quelle: (Dorau 2011), S. 163; Abb. 140, modifiziert und gespiegelt

Gemäß der späteren Ausrichtung der Bildschirme soll auch ein automatisches Drehen der Diagrammteile erfolgen (vgl. Abbildung 4.56). Auf diese Weise würde den Teilnehmern direkt angezeigt, wie sie sich optimal positionieren können (mehr zu Positionen in Kapitel 4.3). Bei Missfallen der Ausrichtung ließe sich die automatische Drehung selbstverständlich mit Hilfe der Spin-Geste wieder anpassen.

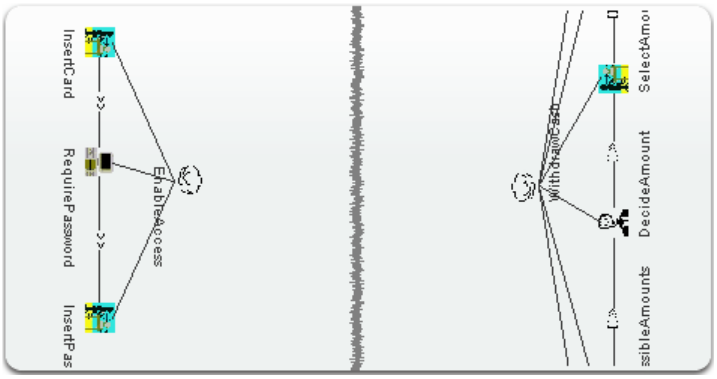


Abbildung 4.56 - TEAR: Automatische Ausrichtung

Weiterhin kann jedoch auch der Fall eintreten, dass lediglich der Bildschirm aufgeteilt werden soll, aber nicht das darunter liegende Diagramm. Aus diesem Grund ist es sinnvoll, nach dem Abschließen der Geste einen Dialog einzubringen, der nach dem gewünschten Ergebnis fragt. Einen Vorschlag dazu zeigt Abbildung 4.57.

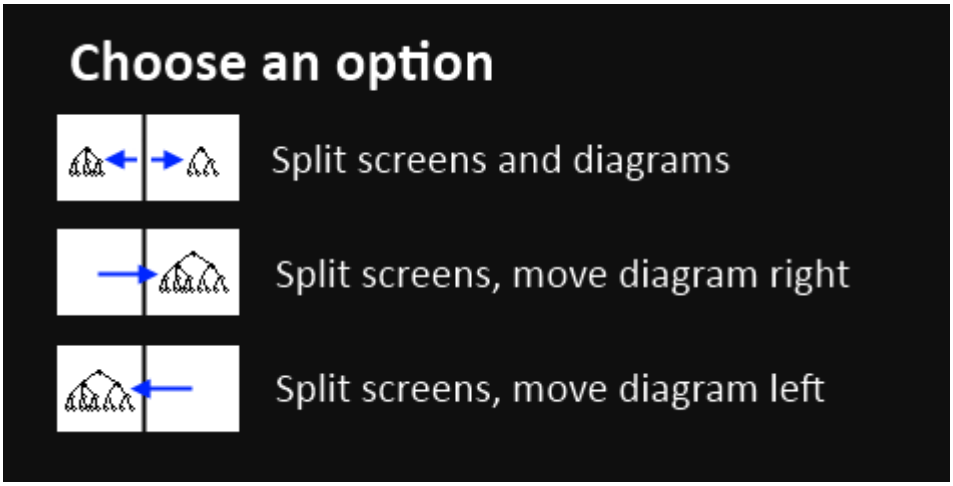


Abbildung 4.57 - Split Dialog

Grundsätzlich sollte diese Geste den Arbeitsbereich immer in der Mitte aufteilen, sodass gleichgroße Hälften für die Partizipierenden zum Arbeiten entstehen; unabhängig davon, wo die Trennlinie bei der Geste angedeutet wurde. Ein Anpassen der Größe der einzelnen Bildschirmteile erscheint an dieser Stelle zunächst vernachlässigbar, diese Thematik wird jedoch ebenso in Abschnitt 4.3 noch einmal rekapituliert.

4.2.2 Tangibles

Die folgenden Überlegungen beziehen sich auf weitere Bausteine im Erstellungsprozess von Aufgabenmodellen, die für die Verwendung der Natural UIs eine sinnvolle Ergänzung darstellen. Die Ideen aus Kapitel 3.2.2 zu Tangible UIs von MARCO ET AL. und OTT ET AL. werden für die effektive Benutzung auf Aufgabenmodellen modifiziert und verkörpern somit eine greifbare Methodik für das Erzeugen von Knoten und Kanten beim Skizzieren von Diagrammen. Dabei werden zunächst nach dem

Ansatz von Jabberstamp oder TOYVision Stempel eingeführt, die einen Abdruck der jeweiligen Knoten oder Kanten auf dem Tisch hinterlassen. Anschließend wird die interaktive Tablet-Stempeltechnik ebenfalls so adaptiert, dass sich Smartphones als dynamischen Stempel für Aufgabenmodelle verwenden lassen.

Beide Techniken sollen dafür auf dem reacTIVision Framework (Kaltenbrunner & Bencina 2007) des TUIO Protokolls beruhen, einem Protokoll für Tabletop Tangible User Interfaces (Kaltenbrunner 2009). reacTIVision nutzt die Fiducial-Technologie mit Amoeba Symbolen und kann – bei entsprechend hochwertiger Hardware bzw. Sensorik – eine Vielzahl verschiedener Amoeba Fiducials auf einem Tisch unterscheiden und verwenden (Reactable Systems S.L. 2009). Unter diesen Voraussetzungen entstehen die folgenden Ansätze.

4.2.2.1 Fiducials als Stempel zur Erstellung verschiedener Knoten- und Kantentypen

In Kapitel 4.2.1 wurde die Verwendung der Finger einer oder beider Hände für die Erstellung von Kanten und Knoten vorgestellt. Dabei kommt es zwangsläufig zu Verzögerungen, weil die vorgestellten Gesten eine gewisse Zeit zum Ausführen benötigen, und weiterhin noch die Art des Knotens oder der Kante im Anschluss bestimmt werden muss. Eine direktere und schnellere Erstellung von Knoten kann ergänzend durch Stempel erfolgen, die einfach auf die Oberfläche aufgesetzt werden und so den jeweiligen Knotentyp an gewünschter Stelle platzieren. Abbildung 4.58 zeigt einen Entwurf solcher Stempel, die als blaue Würfel konzipiert wurden, unter denen ein Fiducial aufgeklebt ist. Jedem Knotentyp ist ein Fiducial mit anderer ID zugewiesen, sodass unterschiedlich gelabelte Würfel verschiedenartige Knoten erzeugen, aber auch mehrere Stempel desselben Knotentyps gleichzeitig verwendet werden können.

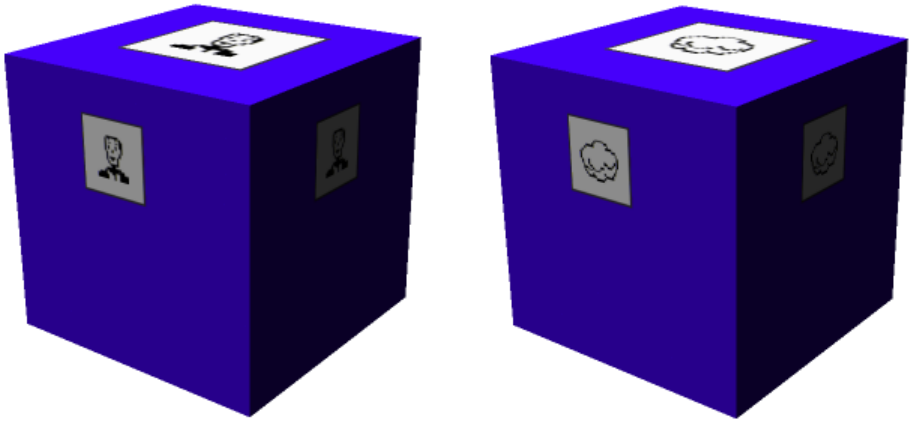


Abbildung 4.58 - Entwurf: Beispiel-Stempel für Diagrammknoten

Auf den Stempeln ist der Knotentyp als Label aufgedruckt, sodass der Anwender direkt den Knotentyp auswählen kann, den er erzeugen möchte. Auf der oberen Fläche des Würfels ist das Icon des Knotentyps in groß angebracht, während an den Seiten dasselbe Icon verkleinert dargestellt wird. Dies ist bewusst so gewählt, um die richtige Orientierung des Würfels direkt zu sehen (die große Seite oben, da man beim Arbeiten am MTT ja auch von oben darauf schaut). Würde der Würfel auf der Seite liegen wäre eines der kleinen Icons nach oben gerichtet; steht er auf dem Kopf, zeigt das Fiducial nach oben. Auf diese Weise wird vom Anwender unmittelbar wahrgenommen, ob der Würfel falsch platziert ist und somit nicht erkannt werden kann.

Wird der Würfel auf die Arbeitsfläche gesetzt, entsteht unter ihm der auf dem Würfel angezeigte Knoten. Dieser wird in Orientierung der Arbeitsfläche ausgerichtet, egal in welcher Drehung der Würfel auf der Arbeitsfläche steht, da das Drehen von Objekten nicht vorgesehen ist. Solange der Würfel nicht

aufgehoben wird, kann er auf der Fläche verschoben werden. Nach dem Absetzen kann er nur dann verschoben werden, wenn ein Stempel des gleichen Typs wieder auf ihn aufgesetzt wird. Sollte er Kinder besitzen, werden diese in ihrer Baumstruktur mit verschoben. Wird die Bewegung schnell ausgeführt, entsteht im Sinne des „Toy Clicks“ sofort ein Knoten auf der Fläche, was sehr schnell wiederholt werden kann, um das Diagramm rapide zu füllen. Durch Ausführen dieser Klickbewegung auf einem Knoten mit einem Stempel anderen Typs wird dessen Typ direkt ohne Auswahlmenü geändert. Wird der Stempel anstatt zu Klicken nicht sofort aufgehoben, kann der Knoten mit verändertem Typ wieder direkt verschoben werden.

In Analogie lassen sich auch Stempel dazu verwenden, Kanten mit temporalen Relationen zwischen den Knoten zu erzeugen. Diese wurden zur Abgrenzung in der Farbe Rot und mit kleineren Abmessungen entworfen, um sie leichter von den Knotenstempeln unterscheiden zu können (vgl. Abbildung 4.59). Auch hier soll wieder gelten, dass die Oberfläche größere Symbole besitzt als die Seiten, was sich aber aufgrund der kleineren Abmessungen als schwierig herausstellt, sollen sie noch lesbar sein. Alternativ könnte die Gesamte obere Fläche ausgenutzt werden, um eine bessere Abgrenzung herzustellen, was dort jedoch einen Verlust der Farbe zur Folge hätte.

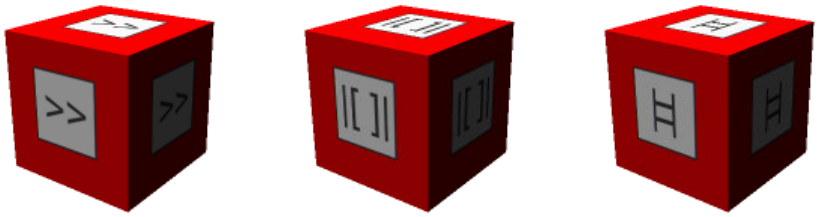


Abbildung 4.59 - Entwurf: Beispiel-Stempel für Diagrammkanten

Um mit diesen Stempeln nun eine Kante zu erzeugen, deren zeitliche Beziehung direkt definiert ist, müssen sie horizontal mittig zwischen zwei Knoten gestellt werden. Auf diese Weise werden die direkten Nachbarn mit der auf dem Stempel angezeigten Relation verbunden. Die Kante kann solange zwischen zwei beliebige Knoten verschoben werden, solange der Stempel aufgesetzt bleibt. Will man die Kante ändern, wird einfach ein anderer Stempel auf sie gestellt, wodurch der Kantentyp unmittelbar geändert wird. Dadurch wird ein Klick eingespart, mit dem man sonst erst das Menü öffnen müsste. Für das Unterordnen von Knoten ist kein Stempel vorgesehen, sodass dies weiterhin durch Gestik erreicht werden muss. Alternativ ließe sich die Unterordnung realisieren, indem zwei beliebige rote Würfel auf einen höher und einen tiefer gelegenen Knoten gestellt werden, sodass nicht extra eine weitere Stempelart dafür eingeführt werden muss.

Wie das gesamte Zusammenspiel in der Realität aussehen könnte, zeigt Abbildung 4.60 in einer dreidimensionalen Skizze. Zu sehen ist der Tabletop mit einem Diagramm, auf dem die verschiedenen Würfel liegen. Man sieht die unter den Stempeln angebrachten Amobea Fiducials, durch die die Erkennung stattfindet.

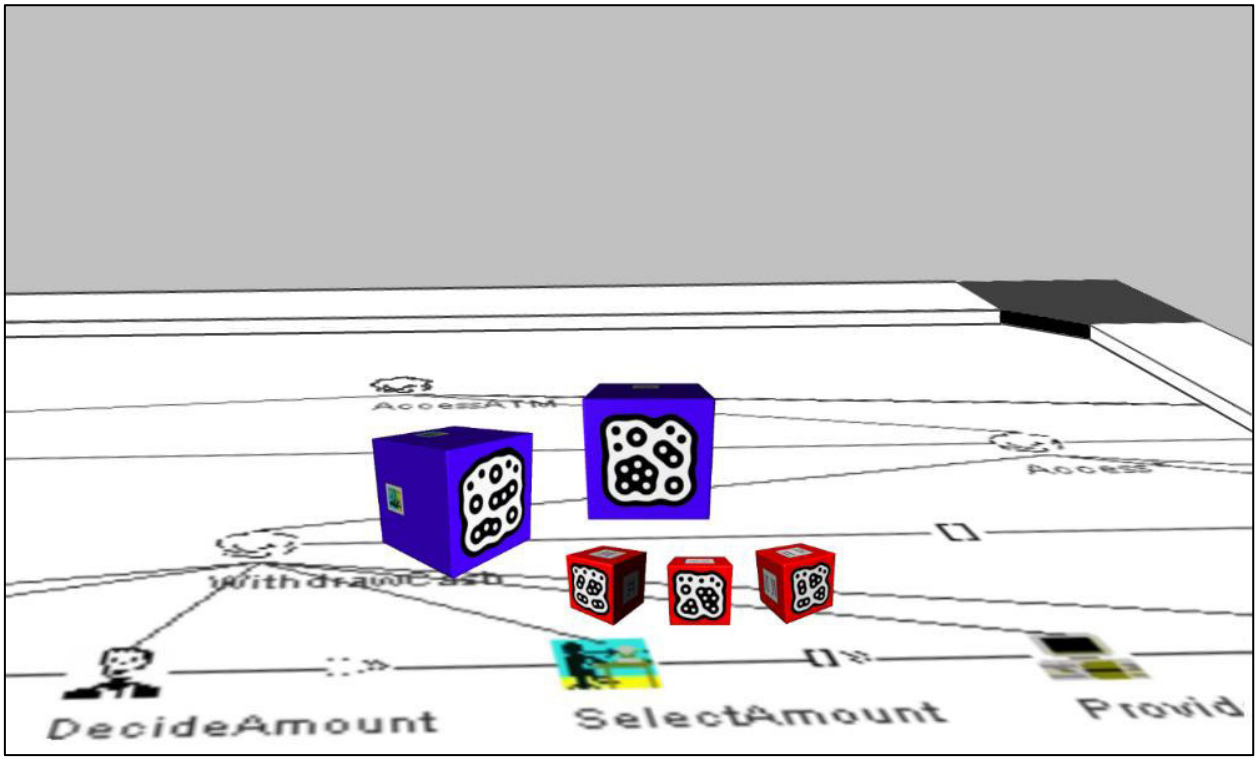


Abbildung 4.60 - 3D-Skizze: Einbettung der Fiducial Stamps im Kontext

Die vorgestellten Stempel können praktisch parallel zur in Kapitel 4.2.1 beschriebenen Gestik verwendet werden. In manchen Aspekten sind sie schneller als ihr Pendant aus der Gestik, sie sind jedoch auch auf nur eine bestimmte Funktion beschränkt und bieten dem Benutzer so weniger Spielraum. Aber gerade diese Beschränktheit macht ihre Benutzung für die gezeigten Fälle durchaus attraktiv. Auch wurde darüber nachgedacht, wie bei TOYVision realer wirkende Figuren als Stempel zu verwenden. Ein Mock-Up zu diesen Überlegungen zeigt Abbildung 4.61.

Abbildung 4.61 - Mock-Up: User- und System-Stempel[61]

4.2.2.2 Smartphone als Objektpalette und Stempel

Um die eben angesprochene Beschränktheit zu umgehen, wird der Ansatz von OTT ET AL. aufgegriffen und ein Stempel entwickelt, mit dem mehr als nur ein Knotentyp erzeugt werden kann. Dazu soll ein Smartphone als Träger dienen, dem zur Erkennung auf der Rückseite zentriert ein Fiducial aufgeklebt ist (s. Abbildung 4.62). Auf diesem Smartphone läuft eine Applikation, die unterschiedliche Stempelfunktionen ermöglicht.

[61] Körper von www.dumini.de – Business Man #1 (Dumini-Business #1)
Computer von www.oldradioworld.de – Computer Radio (modifiziert)

Abbildung 4.62 - Pairing Vorgang für die Erkennung des mobilen Endgerätes[62]

Das Handy wird über WLAN drahtlos an den Tabletop gekoppelt. Dazu muss zunächst ein sogenanntes *Pairing* stattfinden, ein Bekanntmachen von Tisch und Smartphone. Dabei geht es jedoch nicht um die Kopplung der beiden Geräte selbst, da dies über das Client-Server Paradigma gelöst werden kann, wenn der Tisch die Serverrolle einnimmt. Es geht eher um die Zuweisung einer Fiducial ID zur MAC-Adresse des Smartphones, damit beim Auflegen erkannt werden kann, welches Gerät sich nun auf dem Tisch befindet. Dies ist vor allem dann wichtig, wenn mehrere Benutzer am Tisch arbeiten und jeder davon ein Smartphone verwendet. Aus diesem Grund muss auf jedem der Mobilgeräte ein anderes Fiducial aufgeklebt sein. Weiterhin muss sichergestellt werden, dass die Benutzer nacheinander ihre Gerät pairen, da es sonst zu falschen Zuweisungen kommen kann.

Ist das Smartphone noch nicht gepairt, erhält der Benutzer direkt nach dem Starten der App die Aufforderung, sein Smartphone mit dem MTT zu pairen, bevor weitere Optionen verfügbar werden (Abbildung 4.62, rechts). Diese Verknüpfung wird serverseitig gespeichert und soll solange aufrechterhalten bleiben, bis die Anwendung auf dem Tisch beendet wird. Auf diese Weise wird vermieden, dass das Smartphone jedes Mal neu gepairt werden muss, falls versehentlich die App durch den Nutzer auf dem Smartphone beendet wird. Ein manuelles erneutes Pairen soll aber weiterhin über die App angestoßen werden können.

[62] Quelle: Samsung, Modell Galaxy Ace GT-S5830; Bildmaterial von Amazon (http://www.amazon.de), modifiziert

Nachdem der Pairing-Vorgang abgeschlossen ist, erscheint auf dem Display des Smartphones die Auswahl möglicher Knotentypen (Abbildung 4.63, links). Um einen Knoten auf dem Tisch zu erzeugen, muss zunächst der gewünschte Typ ausgewählt werden. In einer neuen Sicht erhält der Anwender den Hinweis, das Smartphone nun auf den Tisch zu legen, um den angezeigten Knoten zu stempeln (Abbildung 4.63, Mitte).

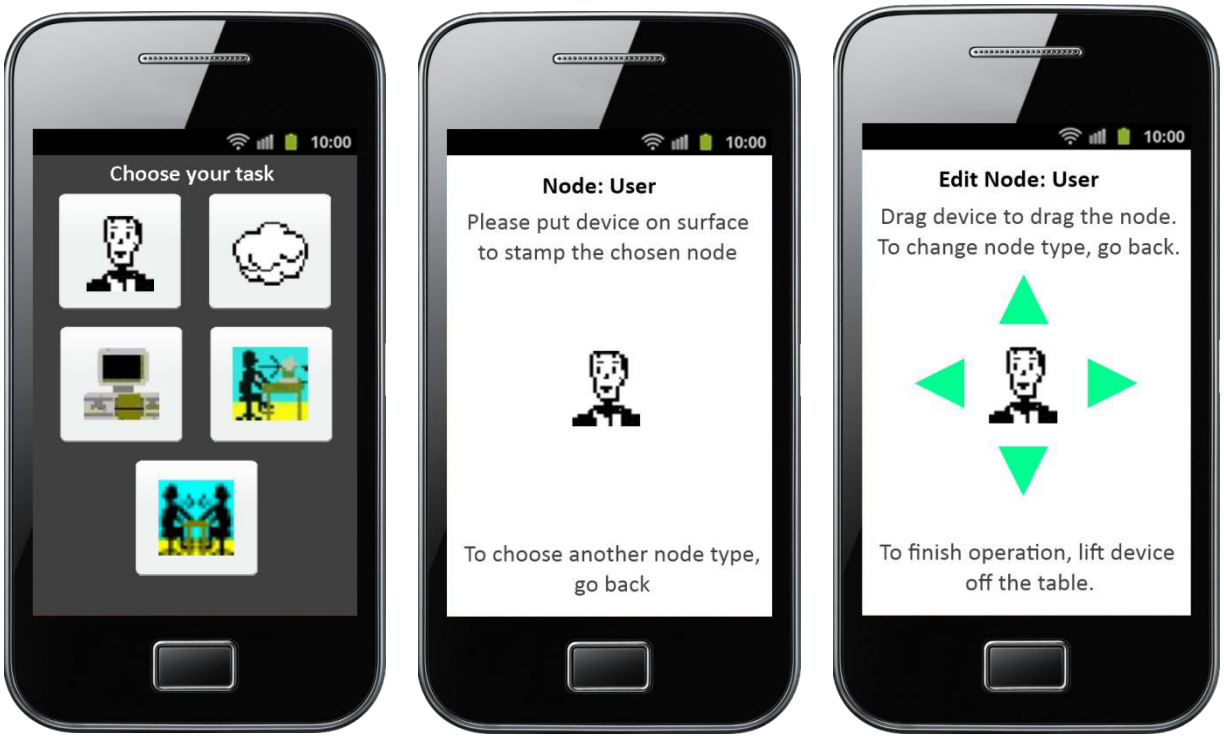


Abbildung 4.63 - Entwurf einer Smartphone-App als TUI: Knoten stempeln

Liegt das Smartphone auf, ist der Knoten erzeugt. In der App wird nun in den Editiermodus gewechselt (s. Abbildung 4.63, rechts), d. h. es werden auf dem Bildschirm weitere Möglichkeiten angezeigt, wie das Verschieben des Knotens durch Verschieben des Gerätes auf dem Tisch; oder das Ändern des Knotentyps durch Zurückgehen über den Sensorschalter des Smartphones und erneutes Auswählen eines anderen Typs. Solange das Smartphone auf dem Tisch liegen bleibt, kann der gerade erzeugte Knoten beliebig verschoben oder sein Typ verändert werden. Ein Verbotsschild könnte zusätzlich anzeigen, dass der Knoten an der momentanen Position nicht abgelegt werden kann (z. B. wenn ein anderer Knoten darunter liegt, auf den der neue geschoben wurde). Ideal wäre die Vorstellung, dass der Untergrund, auf dem sich das Smartphone gerade befindet, in Echtzeit auf seinem Display dargestellt würde, um Konflikte mit anderen Knoten beim Draggen direkt sehen zu können. Nimmt der Anwender abschließend das Gerät wieder auf, werden die Änderungen auf dem Tisch „festgeschrieben“, also der Knoten an gewünschter Position mit dem angezeigten Typ abgelegt (solange er sich nicht auf einem anderen befindet).

Der beschriebene Editiermodus soll auch erreicht werden, wenn das Smartphone einfach auf einen vorhandenen Knoten aufgelegt wird, während es sich im Knotenauswahlmenü befindet. Auf diese Weise kann ebenso das Verschieben dieses Knotens oder das Verändern seines Typs angestoßen werden. Solange ein Knoten „fokussiert“ ist, ist auch denkbar, eine Funktion zum Löschen des Knotens innerhalb der App zu integrieren.

Das gleiche Verhalten gilt für die Erstellung von Kanten, mit dem kleinen Unterschied, dass das mobile Gerät horizontal zwischen zwei bestehende Knoten gelegt werden muss, um eine Verknüpfung der beiden durch die gewählte Beziehung zu erreichen. Dazu wählt der Benutzer zunächst den Kantentyp respektive die temporale Relation in der App aus (Abbildung 4.64, links).

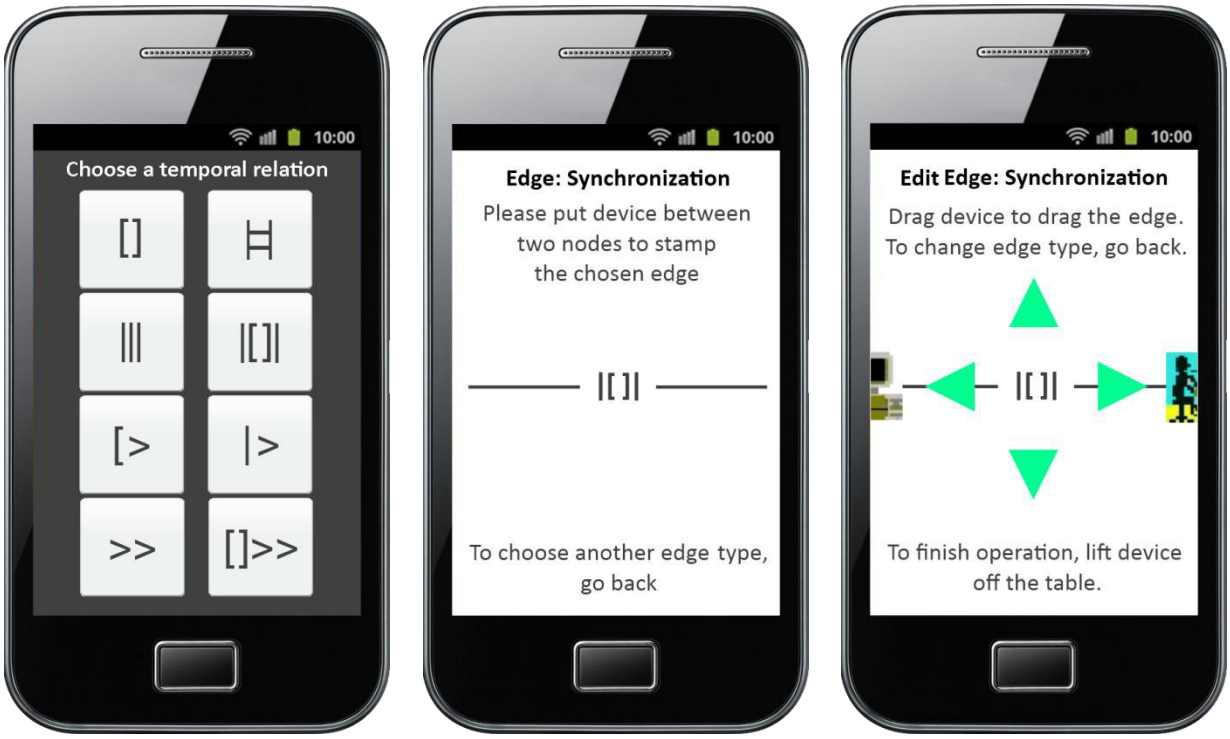


Abbildung 4.64 - Entwurf einer Smartphone-App als TUI: Kanten stempeln

Auf dem Bildschirm wechselt die Ansicht nun wieder auf den Hinweis, das Telefon mittig zwischen zwei bestehenden Knoten zu positionieren (Abbildung 4.64, Mitte). Die Kantenlinien auf dem Display dienen der Orientierung zum Anpeilen der beiden gewünschten Knoten. Ist das Telefon platziert, wird die Kante erzeugt und die App wechselt – analog zur Knotenerstellung – wieder in den beschriebenen Editiermodus (Abbildung 4.64, rechts), der solange erhalten bleibt, wie es auf dem Tisch aufliegt.

Innerhalb des Editiermodus kann die Kante mit dem Telefon verschoben werden, jedoch unter der Prämisse, dass sie dann nur zwischen zwei anderen Knoten platziert werden kann. Wie in der Abbildung rechts zu sehen, werden die beiden verbundenen Knoten zur Orientierung des Anwenders angezeigt. Beim Bewegen des Telefons aus dem Bereich zwischen zwei Knoten heraus verschwindet die Kante auf dem Tisch wieder; ebenso die verbundenen Knoten auf dem Display des Telefons, bis sich das Gerät wieder zwischen zwei anderen Knoten befindet.

Genau wie bei der Knotenbearbeitung soll es auch möglich sein, durch Auflegen des Smartphones auf eine vorhandene Kante diese in den Editiermodus aufzunehmen, um die gerade beschriebenen Schritte auch auf bereits vorhandenen Kanten auszuführen. Nach Veränderung der Kante können die Eigenschaften festgeschrieben werden, indem das Telefon vom Tisch entfernt wird. Einzig die Unterordnung von Knoten durch Kanten kann durch die vorgestellte Technik nicht realisiert werden.

Der gerade beschriebene Vorgang steht in direktem Bezug zu den Stempeln aus Kapitel 4.2.2.1. Jedoch ist es von der Analogie zur Realität her intuitiver, Kanten zu zeichnen anstatt sie zu stempeln, da sie nun einmal Verbindungslinien darstellen. Um diesen Umstand gerecht zu werden, könnte alternativ nach Auswahl der Kantenart anstelle des Stempelvorgangs ein Zeichenvorgang eingeräumt werden, bei dem der Anwender mit einer Ecke seines Smartphones eine Linie zwischen den beiden gewünschten Knoten zieht (s. Abbildung 4.65).

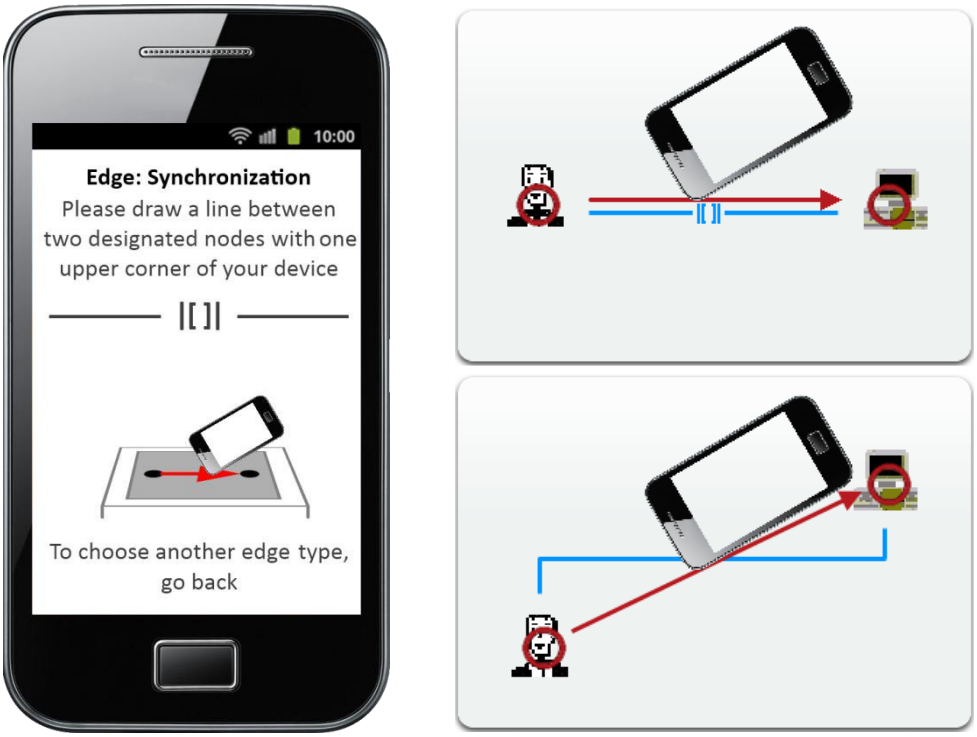


Abbildung 4.65 - Smartphone: Kanten zeichnen

Auf diese Weise könnte wie mit einem Stift agiert werden, um Kanten direkt mit entsprechendem Typ zu zeichnen (rechts oben). Die rote Linie gibt dabei die Bewegung mit dem Smartphone an, und die blaue Linie verkörpert die resultierende Kante. Durch Verwendung dieser Methode könnte auch das Problem der Unterordnung einfach gelöst werden, indem man topografisch unterhalb liegende Knoten mit dem Smartphone verbindet (rechts unten). Obwohl die Knoten per „Luftlinie“ verbunden werden, ordnet sich die Kante wie beim 2-Finger-Tap rechtwinklig in die Struktur ein, wie an der blauen Linie erkennbar ist. Wie diese Erkennung möglich wird, ist abhängig vom Lagesensor des Telefons, und wird ein weiteres Mal in Kapitel 4.2.2.3 aufgegriffen.

Um zwischen den verschiedenen Modi innerhalb der App wechseln zu können, soll eine einfache Wischgeste ausreichen (vgl. Abbildung 4.66). Durch Wischen – egal in welche Richtung ausgeführt – wechselt der Bildschirm der App sukzessive von einem Bereich zum nächsten, und fängt am Ende wieder mit dem ersten an, sodass ein geschlossener Kreis entsteht, der intuitiv durchlaufen werden kann.

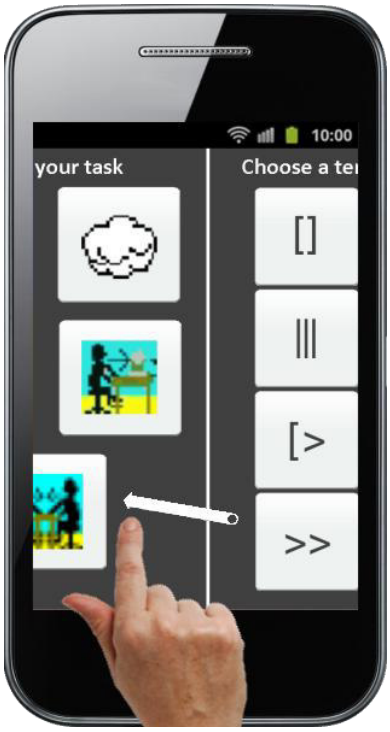


Abbildung 4.66 - Bildschirmwechsel in der App

Funktionen abbrechen kann der User jederzeit durch betätigen der Zurücktaste (bspw. das Erstellen von Knoten oder Kanten) und auch die App wird geschlossen, wenn die Zurücktaste in einem der Auswahlmenüs betätigt wird. Abschließend lässt sich sagen, dass die vorgestellten Ansätze der Unterstützung des Erstellungsprozesses mit Hilfe der Gesten dienen, da sie eine direkt definierte Erstellung von Diagrammelementen ermöglichen. Natürlich kann jeder der Schritte auch alternativ durch die Verwendung einer Geste ersetzt oder ergänzt werden. Da das Verschieben des Smartphones nach einiger Zeit etwas mühsam sein kann, ist die Tendenz sehr hoch, dass die Technik rein für das Stempeln der Objekte eingesetzt wird, das spätere Editieren aber komplett auf die Gesten entfällt. Eine weitere Technik zum Editieren von vorwiegend textuellen Knoteneigenschaften wird im folgenden Abschnitt beschrieben.

4.2.2.3 Smartphone zum Bearbeiten textueller Knoteneigenschaften

Für die Texteingabe ist bisher, wie in Kapitel 4.1.5 dargestellt, eine OSD Tastatur vorgesehen, die temporär für die Benennung von Knoten bzw. ihrer Eigenschaften eingeblendet wird. Die Eingabe von Texten über so eine Tastatur an einem großen Display kann sehr lange dauern und auch ermüdend sein, weshalb eine Alternative zu den bisher gemachten Äußerungen aufgezeigt werden soll. Weiterhin ist nicht zu unterschätzen, wie viel Platz solch eine Softwaretastatur bei sinnvoller Größe auf einem MTT einnimmt. Dadurch wird ein Großteil des darunter liegenden Diagramms verdeckt, sodass der Überblick leicht verloren gehen kann.

Texteingaben auf dem Smartphone gehören zu den alltäglichen Dingen: Man schreibt damit SMS oder Nachrichten in sozialen Netzwerken, verfasst unterwegs Blogeinträge oder Twitter-Meldungen oder gibt einfach Begriffe in Suchmaschinen ein. Warum also nicht das Handy für Texteingaben benutzen, wenn es uns sonst im Alltag in dieser Hinsicht auch permanent gute Dienste leistet?!

Um Texte auf dem Smartphone eingeben zu können, müssen Elemente vom Tisch mit dem Telefon „aufgehoben“ werden können. OTT ET AL. haben diesen Vorgang in ihrer Veröffentlichung als „fokussieren“ beschrieben (s. 3.2.2.2). Um das zu erreichen, muss in der App ein weiterer Bildschirm

neben den Stempelvorgängen eingefügt werden, der das Aufnehmen eines Knotens initiieren kann und per Wischen erreicht wird. Bei Aktivierung wird über WLAN ein Request an den Tisch gesendet, dass mit dem Smartphone ein Knoten zum Editieren vom MTT aufgenommen werden soll (Abbildung 4.67, links).

Abbildung 4.67 - Knotentexte bearbeiten per TAP mit dem Smartphone: Pick Node

Der Anwender erhält dann (nach Bestätigung vom Tisch) die Aufforderung, mit einer der oberen Ecken des Smartphones den gewünschten Knoten auf dem Tisch zu tippen, um diesen aufzunehmen (s. Abbildung 4.67, rechts). Der designierte Knoten kann beispielsweise dadurch ermittelt werden, dass gleichzeitig ein TAP auf dem Tisch in Verbindung mit entsprechenden Werten des Neigungssensors des Telefons registriert wird, die während der Aufnahmephase kontinuierlich an den Tisch übermittelt werden.

Da die Ecken des Telefons schlecht mit Fiducials ausgestattet werden können und vor allem im Multi-User Fall deshalb kaum unterscheidbar wäre, mit welchem Smartphone gerade auf den Tisch getippt wird, sollte immer nur ein aktiver Request an den MTT möglich sein. Ist dieser abgeschlossen und das Objekt aufgenommen, können weitere Knoten von anderen Teilnehmern aufgehoben werden – auch während der zuvor aufgenommene Knoten noch am Smartphone bearbeitet wird. Denkbar wäre hierfür eine Art Warteschlange, die die Requests der Reihe nach speichert und die Anwender (z. B. per Vibration) benachrichtigt, sobald sie ihre Pick-Operationen durchführen können. Die Warteschlange müsste natürlich auch einen Timeout besitzen, sodass Personen, die einen Request gestartet haben, auch den Tisch einfach verlassen können ohne die anderen Teilnehmer zu blockieren.

Wie die Verwendung in der Praxis aussehen könnte, zeigt die 3D Skizze in Abbildung 4.68. Es ist die Aufforderungsnachricht erkennbar, und durch das Tippen mit dem Smartphone auf den Tisch wird der Interaction Task Knoten „SelectAmount“ ausgewählt und zur Bearbeitung aufgehoben. Die App wechselt dann in den Bearbeitungsmodus, in dem der angetippte Knoten nun angezeigt wird.

Abbildung 4.68 - 3D Skizze: Pick Node in der Praxis

Auf dem Bildschirm des Smartphones können nun alle in Kapitel 4.1.5 genannten Eigenschaften, die Texteingabe erfordern, bequem mit der Handytastatur durchgeführt werden. Dabei werden als Wiedererkennungswerte Design und Symbolik der Menüs und der beschriebenen Erweiterungsfenster des Tisches auf dem Telefonbildschirm beibehalten (Abbildung 4.69). Weiterhin muss sichergestellt sein, dass während des Bearbeitungsvorgangs auf dem Telefon kein anderer Teilnehmer den Knoten parallel am Tisch verändern kann. Denkbar wäre, dass der Knoten – während der Bearbeitung auf dem Smartphone – auf dem MTT weder verschoben noch angetippt werden kann. Die Sperre könnte visuell unterstützend durch eine temporäre Verblassung des Icons auf dem Tisch angezeigt werden.

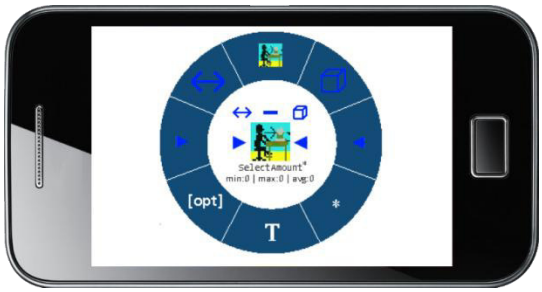


Abbildung 4.69 - Knoteneditierung Smartphone

Beispielhaft wird die Texteingabe in Abbildung 4.70 dargestellt. Nachdem auf dem Smartphone im Pie-Menü auf das Price Tag gewechselt wurde, lässt sich die Bezeichnung des Knotens mit Hilfe der Handytastatur einfach ändern (oben). Auch die Aufgabenobjekte lassen sich – nachdem der Objekteditor über das Menü ausgewählt wurde – einfach durch Antippen des Textes direkt in der App umbenennen (unten), aber auch wie auf dem Tisch über die Schaltfläche anlegen. Entsprechend werden analog alle anderen Textfelder auf dem Smartphone behandelt. Ebenso ist die Veränderung nicht textueller Eigenschaften über Schieberegler und Bullet Points effektiv nutzbar. Da die Ansicht auf dem Display des Telefons eingeschränkt ist, sollte darüber hinaus das Zoomen auf den gezeigten Bildschirmen möglich sein.

Abbildung 4.70 - Bearbeitung des Knotennames / von Objektnamen

Nachdem alle Änderungen getätigt wurden, kann der Benutzer innerhalb der App mit dem Zurückschalter des Smartphones zum Ursprungsbildschirm zurückkehren und erhält durch ein weiteres Zurück die Frage, ob die Änderungen des Knotens gespeichert werden sollen. Unabhängig davon, ob die Änderungen übernommen oder verworfen werden, wird der Knoten anschließend auf dem Tisch wieder für andere freigegeben.

Nach den Betrachtungen zur Interaktion via Gesten und Tangibles widmet sich jetzt abschließend das nächste Kapitel der Kollaboration bei der Bearbeitung von Aufgabenmodellen, bevor die gewonnenen Erkenntnisse in einem Fazit zusammengefasst und reflektiert werden.

4.3 Kollaboration

Ein großer Vorteil eines Multi-Touch Tabletops liegt in seiner physikalischen Größe. Die Dimensionen des Bildschirms, respektive der Arbeitsfläche eröffnen neue Wege, miteinander effektiv an ein und demselben Modell zu arbeiten. Um die nun folgende Vorgehensweise zu verdeutlichen, wird erst vom Single-User Fall ausgegangen und dieser dann sukzessive erweitert, um die Verteilungen der Kollaborateure genau nachvollziehen zu können.

Dabei wird eine Aufteilung der Arbeitsfläche vorgesehen, um zielgerichtetes Arbeiten an einem Aufgabenmodellierungsprojekt sicherzustellen. Die Aufteilungsschritte finden einmal auf räumlicher

Ebene, aber anschließend auch auf fachlicher Ebene statt, sodass eine zusammenhängende Vorstellung des verteilten und doch gemeinsamen Erstellungsprozesses deutlich wird.

4.3.1 Räumliche Aufteilung

Geht man zunächst vom Single-User Fall aus, wird sich der Anwender dabei üblicherweise allein an einer der längeren Seiten der rechteckigen Fläche aufhalten (vgl. Abbildung 4.71). Dies liegt darin begründet, dass man beim Arbeiten mit Computern diese Ansicht gewohnt ist, da standardmäßig nahezu alle Bildschirme dieses Format und diese Ausrichtung aufweisen. Hinzu kommt außerdem, dass die Länge der Arme eines Menschen begrenzt ist und der Aktionsradius durch Gestik sich bei normalem Stehen halbkreisförmig um den Standpunkt legt. Da die Arbeitsfläche im Querformat vorliegt, ist es durch Beugen und Strecken des Körpers wesentlich einfacher, die gesamte Fläche zu erreichen, als dies im Hochformat der Fall wäre.

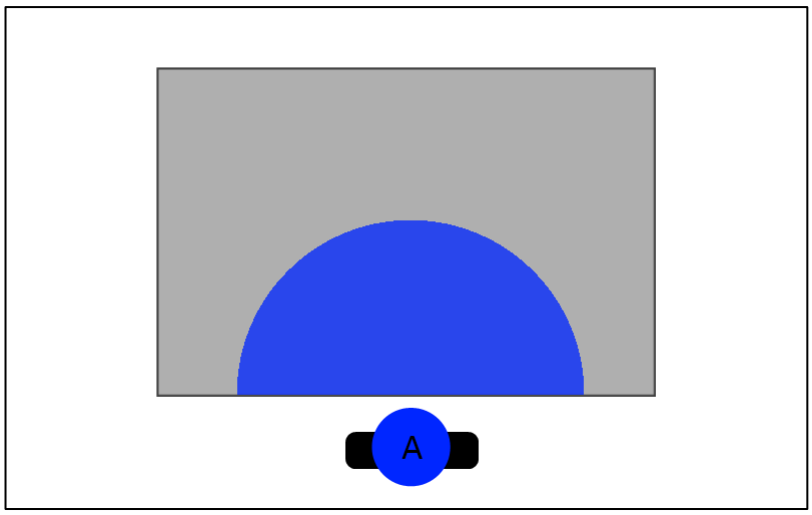


Abbildung 4.71 - Single User Szenario

Durch die gewählte Ausrichtung ist sichergestellt, dass alle Elemente des Diagramms auf dem Bildschirm erreicht werden können; und obwohl sie auch durch Verschieben in den zentralen Bereich vor den Benutzer zu bringen wären, wirkt diese Orientierung dennoch vertrauter und gewohnter in der Anwendung. Aus diesem Grund sollte sie auch für weitere Partizipierende beibehalten werde, wie nun im Folgenden weiter beschrieben wird.

Abbildung 4.72 zeigt den typischen Fall, wenn ein zweiter Benutzer (B) für die Bearbeitung hinzukommt. Zunächst wird er sich neben dem existierenden Benutzer (A) anordnen, um sich einen Überblick über das, was auf dem Tisch dargestellt wird, zu verschaffen. Diese These wird dadurch bekräftigt, dass das Diagramm eine dem Benutzer A zugewandte Orientierung besitzt; und deshalb auf dem Kopf oder von der Seite erschwert zu betrachten wäre. Doch würden jetzt beide Benutzer parallel anfangen wollen zu arbeiten, entstehen nicht unerhebliche Konflikte, die es zu vermeiden gilt. Das Erzeugen von Knoten und Kanten, sowie deren Verschieben auf der Arbeitsfläche gestaltet sich zunächst problemlos. A und B können beide innerhalb ihres Aktionsradius beliebig Elemente erzeugen und versetzen (auch wenn sich die Aktionsradien bereits überschneiden).

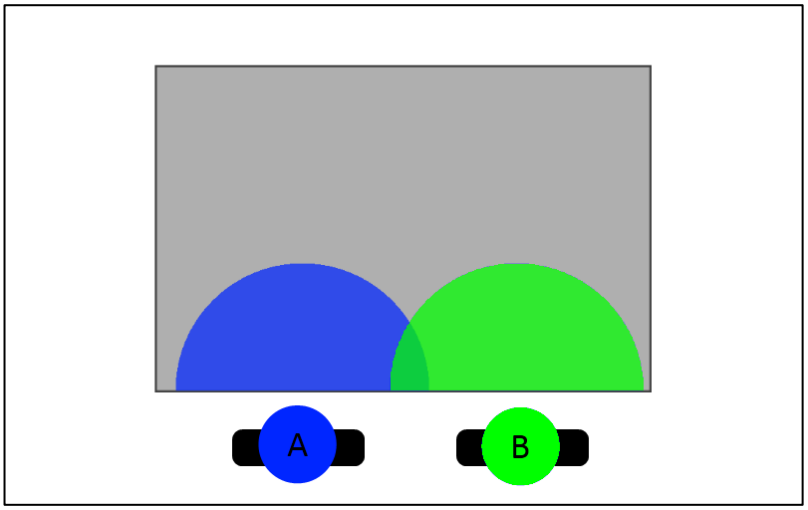


Abbildung 4.72 - Zwei Benutzer Standardszenario

Aber schon der einfache Fall des Verschiebens der Arbeitsfläche würde wahrscheinlich einen der beiden Akteure stören oder behindern, sollte die Aktion nicht zuvor verbal abgesprochen durchgeführt werden. So könnte im schlimmsten Fall User A die erzeugten Elemente von B aus dem Arbeitsbereich herausschieben, wenn er den Bildbereich nach rechts verschiebt, um weiter links Platz für neue Elemente zu schaffen. Ähnlich verhält es sich mit dem Zoomen: Wenn User B die Ansicht vergrößern will, User A dies aber nicht benötigt, entsteht erneut ein Konflikt. Das ist auch der Grund, warum auf eine Abdunkelung der Arbeitsfläche beim Aufrufen des Knoten-Kontextmenüs aus Kapitel 4.1.4 verzichtet wurde, um nicht ungewollt anderen Teilnehmern den Bildschirm zu verdunkeln.

Weiterhin ist noch das Zwei-Finger Wischen durch verschiedene Ansichten zu nennen, was bei einem ungetrennten Bildschirm naheliegend für beide Akteure die Ansicht wechselt, auch wenn lediglich einer der beiden diese Änderung wünscht. Als letztes Beispiel wäre noch die Rückgängig-Funktion zu nennen, bei der nicht unterscheidbar wäre, welcher der letzten Schritte von welchem der verschiedenen Benutzer zurückgegangen wird, da keine separate Trennung vorliegt. Mit dieser Problematik haben sich SEIFRIED & SCOTT beschäftigt, indem sie Techniken für regionales Undo/Redo auf großen Displays einführen (Seifried & Scott 2012); sie ist jedoch nicht Gegenstand der weiteren Betrachtungen.

Aus allen zuvor genannten Gründen erscheint die Aufteilung des Bildschirms zunehmend unausweichlich, wodurch auch die in Kapitel 4.2.1.2 etablierte TEAR-Geste zum Teilen des Bildschirms ihren Zweck erhält. Durch Anwendung der TEAR-Geste wird also der Bildschirm bspw. durch Benutzer A aufgeteilt (sog. *Split Screen*). Dabei soll nach Durchführen der Geste der aus 4.2.1.2 bekannte Split-Dialog erscheinen, der nach der gewünschten Aufteilungsoption fragt (vgl. Abbildung 4.56). Entweder wird nun das existente Diagramm gemäß der Startlinie der TEAR-Geste aufgetrennt (sodass A und B anschließend mit Teilen des Ursprungsdiagramms weiterarbeiten), oder es erfolgt lediglich die Aufteilung des Arbeitsbereiches – wobei das Ursprungsdiagramm dann schlussendlich auf einem der beiden Teile angezeigt wird, während der andere eine leere Fläche zum Arbeiten erhält. Das spätere Zusammenlegen (Merge) der getrennten Diagramme oder Arbeitsbereiche ist dann durch entgegengesetzte Gestik realisierbar, wobei beide Teile gleich ausgerichtet und nebeneinander abgelegt werden müssen (später in Kapitel 4.3.2).

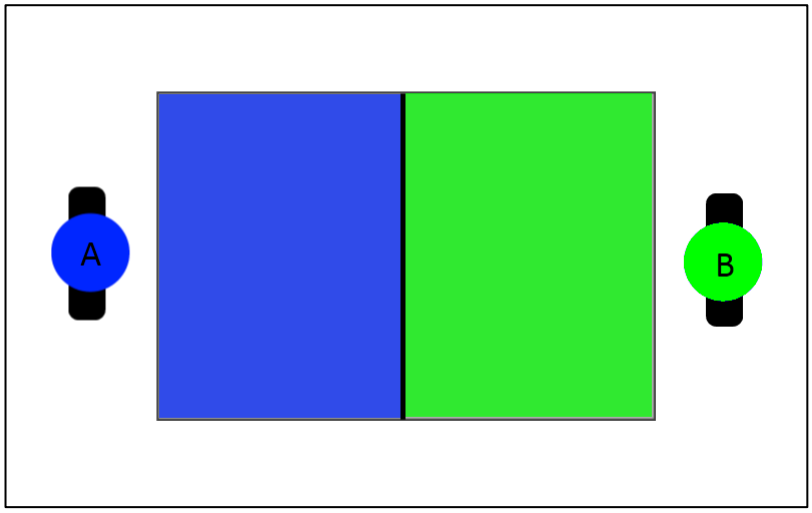


Abbildung 4.73 - Zwei Benutzer Split Screen

Um beiden Akteuren das gleiche Ursprungsformat in verkleinerter Form zu ermöglichen, müssen A und B nun an die kurzen Seiten des Tisches wechseln (Abbildung 4.73) – wie durch die automatische Ausrichtung der Tear-Geste vorgesehen. Mit Hilfe dieser Separierung ist es nun für beide möglich, unabhängig voneinander konfliktfrei Diagramme erstellen zu können. Der einzige Konflikt, der bestehen bliebe, ist das gleichzeitige Anzeigen desselben Inhaltes (Diagramme bzw. Rollen; Rollenmanager). Wird durch die Zwei-Finger Wischgeste auf eine andere Ansicht gewechselt, muss aber sichergestellt werden, dass nicht beide Anwender auf dieselbe Sicht Zugriff erhalten, was bereits in Abschnitt 4.2.1.1 im Zuge der Scroll-Geste angesprochen wurde.

Dies könnte beispielsweise durch ein Verbotsschild auf nicht verfügbaren Ansichten während des Scrollens durch die Liste der existierenden Ansichten noch stärker signalisiert werden (s. Abbildung 4.74). Alternativ wäre auch das einfache Ausblenden der derzeit in Benutzung befindlichen Instanzen möglich, was aber im Umkehrschluss auch das unmittelbare Verständnis für die Komplexität des Projektes reduziert.

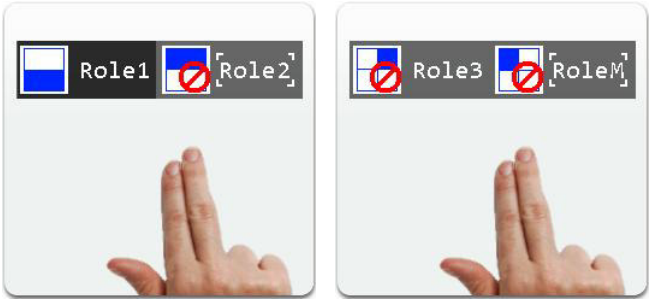


Abbildung 4.74 - Blockierte Ansichten bei Multi-User: links zwei, rechts vier

Kommen weitere Nutzer hinzu, können die soeben aufgeteilten Bildschirmfragmente weiter aufgeteilt werden. Um ein effektives Arbeiten zu erhalten, sollte die Anzahl von vier Personen beim gleichzeitigen Arbeiten nicht überschritten werden, da es sonst zu Platzproblemen auf der Oberfläche kommt und weiterhin nicht mehr jedem der Benutzer das gleiche Format zum Erstellen präsentiert werden kann. Die maximal mögliche Aufteilung zeigt Abbildung 4.75. Nachdem User A und B ihre Bildschirme mit der TEAR-Geste ein weiteres Mal aufgetrennt haben, entstehen weitere Plätze für C und D.

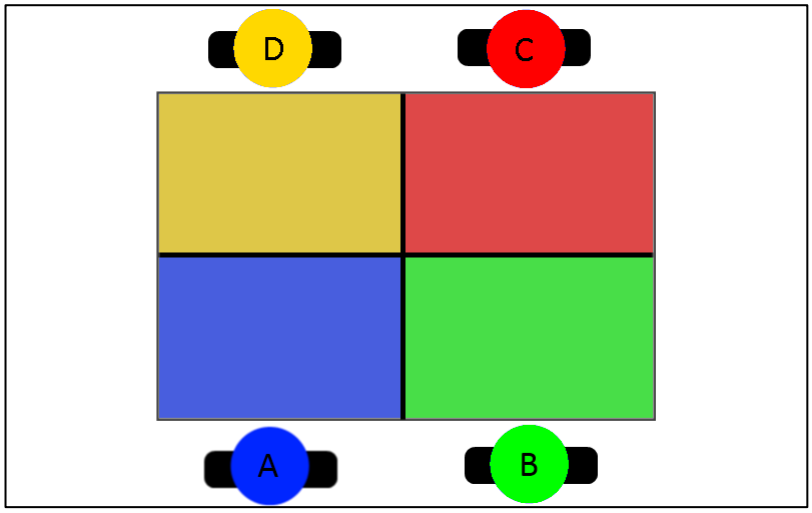


Abbildung 4.75 - Vier Benutzer Split Screen

A und B wechseln zurück auf ihre Ausgangsposition, womit wieder sichergestellt ist, dass alle Teilnehmer dieselbe Ausrichtung und dasselbe Format erhalten wie im Einzelbenutzerbetrieb – um den Faktor vier verkleinert. Analog zum Zwei-Benutzerbetrieb soll auch hier wieder die Orientierung der Teilbildschirme automatisch durch die TEAR-Geste vorgegeben werden. Für den Fall, dass drei Benutzer zeitgleich arbeiten, ergeben sich verschiedene Möglichkeiten der Bildschirmaufteilung. Da konsequent am Seitenverhältnis festgehalten werden soll, bleibt als einzige Möglichkeit die folgende Variante übrig, dargestellt in Abbildung 4.76. Sie kann selbstverständlich auch symmetrisch gespiegelt realisiert werden.

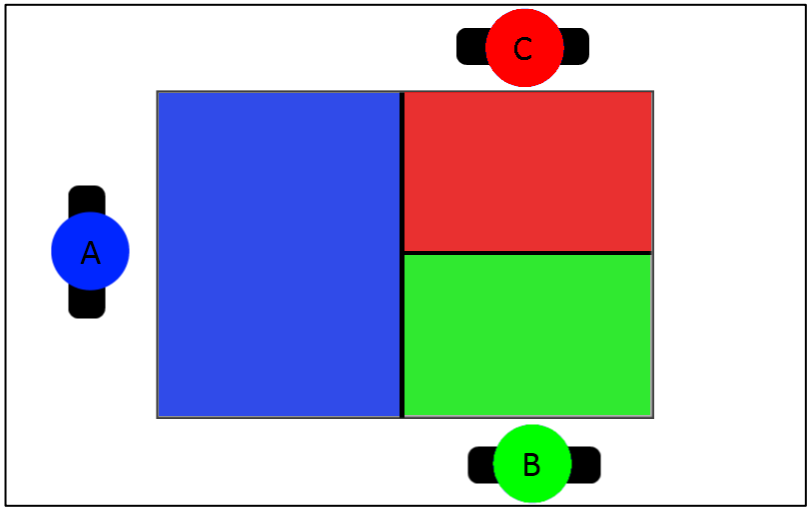


Abbildung 4.76 - Drei Benutzer Split Screen

Schlussendlich wäre es von Vorteil, die Grenzen der Aufteilung frei verschieben zu können. Wenn sich beispielsweise durch initiierendes HOLD und anschließendem DRAG die inneren Trennlinien der gesplitteten Bildschirme verschieben ließen, könnte auf verschiedene Anforderungen von Benutzern eingegangen werden, z. B. wenn einer der Teilnehmer mehr Platz benötigt als die anderen, weil er ein komplexeres Modell bearbeitet. Außerdem ist so bei drei Benutzern durch Verschieben der Vertikalgrenze eine fairere Aufteilung der Bildschirme möglich, bei der alle Anwesenden die gleiche Teilgröße in etwas breiterem Seitenverhältnis erhalten (vgl. Abbildung 4.77).

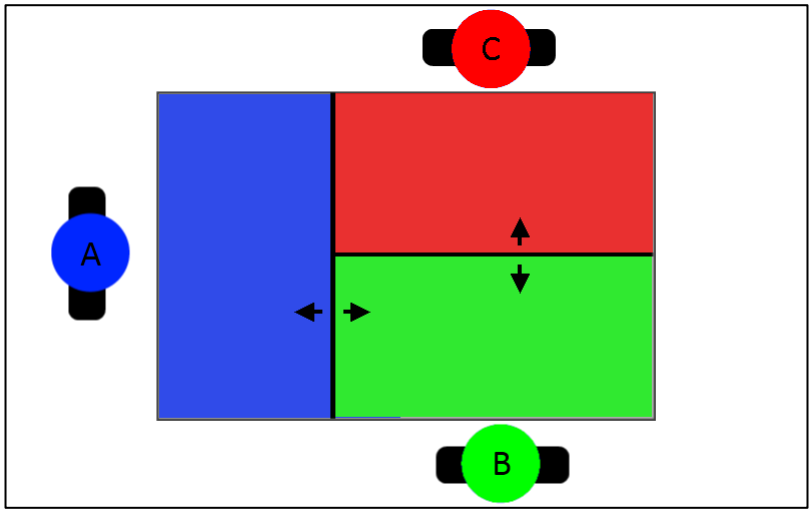


Abbildung 4.77 - Faire Aufteilung durch Grenzverschiebung bei drei Benutzern

Das Versetzen der horizontalen Trennlinie ist nur im beschränkten Maße sinnvoll geeignet, da sie den Arbeitsbereich der Personen auf einer Seite sehr stark dezimiert, sodass im schlimmsten Fall lediglich eine breite Linie zur Diagrammbearbeitung übrig bleibt. Obwohl eine Verschiebung dieser Linie vielleicht besser gar nicht zu erwägen sein sollte, darf sie trotzdem der Vollständigkeit halber im Konzept nicht fehlen, da sie unter Umständen in einem speziellen Szenario eben doch gerechtfertigt sein kann.

Nachdem nun detailliert die räumliche Aufteilung dargestellt wurde, bei der vorwiegend die Positionen der Benutzer und ihre zugewiesenen Arbeitsbereiche eine Rolle spielen, soll nun im nächsten Abschnitt über fachliche Aufteilungsaspekte referiert werden.

4.3.2 Fachliche Aufteilung

Fachliche Aufteilung spielt neben der räumlichen Partitionierung eine wichtige Rolle bei der kooperativen Zusammenarbeit. Nachdem der Tisch für die Benutzer in eigene Arbeitsbereiche eingeteilt wurde, stehen maximal vier neue, getrennt bedienbare Sichten zur Verfügung, die einer gemeinsamen Zielführung dienen. Da nicht alle Teilnehmer am selben Diagramm arbeiten können, macht eine Aufteilung besonders für den Fall des *Cooperative Mode* Sinn, bei dem Rollen definiert und getrennt bearbeitet werden können.

Ein erzwungenes Definieren der Rollen zu Beginn würde aber ein ungehindertes Starten in den Entwicklungsprozess unterbinden, da zunächst durch einen der Anwender festgelegt werden müsste, welche Rollen es gibt, bevor die anderen Teilnehmer ihre Diagramme erstellen können. Aus diesem Grund soll es – wie bereits geschildert – möglich sein, Diagrammen erst später eine Rolle zuzuweisen, sodass direkt alle Partizipierenden mit ihrem Beitrag beginnen, sobald die Applikation gestartet ist und die Aufteilung stattgefunden hat. Derjenige, der das kooperative Modell bearbeitet, muss parallel auch die Rollen anlegen und diese den bereits vorhandenen Diagrammen zuweisen.

Für mögliche Aufteilungen gibt es dabei viele Schemata, sodass keine eindeutige fachliche Aufteilung vorgegeben werden kann. Es kann lediglich vorher verbal abgestimmt werden, wer welche Rolle bei der Erstellung übernimmt – auch wenn diese noch nicht im Projekt angelegt wurden. Dieser Ansatz begünstigt das spontane Anpassen der Situation an mögliche hinzukommende Teilnehmer. Zu Beginn sollte derjenige bestimmt werden, der das Projekt verwaltet und somit vorwiegend mit dem Rollenmanager arbeitet. Diese Aufgabe birgt die meiste Verantwortung und erfordert erweiterte Koordinationsarbeit, da Rollen und Connection Tasks zugewiesen werden müssen. Die weiteren Rollen können beliebig auf die anderen Teilnehmer verteilt werden.

Im Single-User Betrieb müssten bei einem kooperativen Modell ohnehin alle Rollen sowie die Verknüpfungen durch ein und dieselbe Person getätigt werden. Die möglichen Verteilungen bei verschiedenen Useranzahlen (zwei bis vier) zeigt Abbildung 4.78. Auffallend ist, dass der Rollenmanager bei allen Konstellationen vertreten ist. Die Position der verschiedenen Rollen bzw. des Rollenmanagers ist dabei aber variabel und kann durch die in Kapitel 4.2.1 beschriebene Scroll-Geste beliebig auf jedem virtuellen Bildschirm angepasst werden, sodass individuell rotiert werden kann.

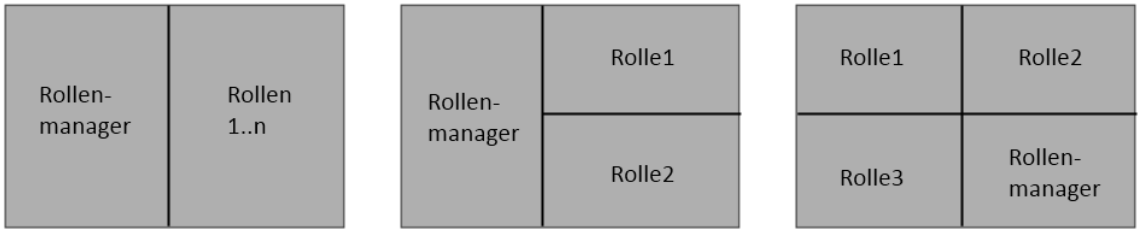


Abbildung 4.78 - Mögliche Beispielschemata für die fachliche Aufteilung

Da es häufiger vorkommen kann, dass es weniger Rollen als Teilnehmer gibt (beim nicht kooperativen Modus gar keine), wäre es durchaus von Vorteil, Teilbäume eines Diagramms durch verschiedene Teilnehmer zu bearbeiten. Die Trennung aus Abbildung 4.78 würde dann nicht verschiedene Rollen beinhalten, sondern Teildiagramme bzw. -bäume. Für das Teilen und Zusammenlegen von Arbeitsbereichen ist eine entsprechende Gestik etabliert worden und der Zusammenlegungsprozess soll nun näher erläutert werden. Angenommen, auf zwei benachbarten Teilbildschirmen befinden sich Teilbäume eines größeren Aufgabenbaums, die nun zusammengelegt werden sollen (s. Abbildung 4.79). Es ist dabei unerheblich, ob die beiden Bildschirmteile gegenüber oder nebeneinander angeordnet sind.

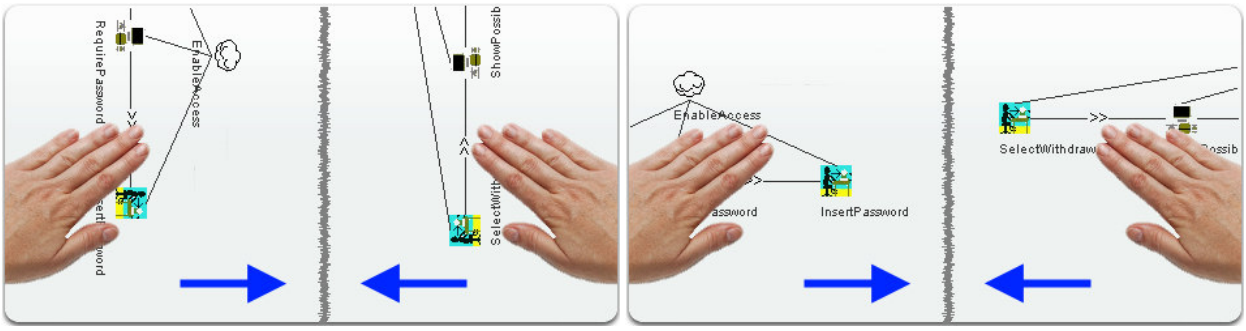


Abbildung 4.79 - Zusammenlegen von Diagrammen (Merge)

Durch Zusammenschieben der geteilten Bildschirme soll ein Dialog geöffnet werden, der die Anwender nach dem Zweck der Zusammenlegung fragt. Dabei kann entweder einfach eine Zusammenlegung der Arbeitsbereiche intendiert gewesen sein, weil bspw. jemand die Teilnahme beendet, oder aber auch das Zusammenlegen zweier Diagramme oder beides. Von essentieller Wichtigkeit ist dabei, dass die Anwendung die Diagramme so miteinander kombiniert, dass keine Überlagerungen der Teilbäume auftreten und diese auch gleich ausgerichtet werden, sie also im Endeffekt nebeneinander auf demselben Hintergrund abgelegt sind.

Ein entsprechender Dialog, der als Overlay über dem Bereich eingeblendet wird, könnte wie in Abbildung 4.80 (links) aussehen. Dabei kann – neben dem reinen Zusammenlegen der Arbeitsbereiche – beim Verschmelzen der Diagramme die Aufteilung der Arbeitsfläche weiterhin bestehen bleiben oder gleichermaßen mit zusammengelegt werden. Optional kann auch mit einem leeren Diagramm weitergearbeitet werden, wobei das verschmolzene Diagramm dann im Pool hinterlegt wird.

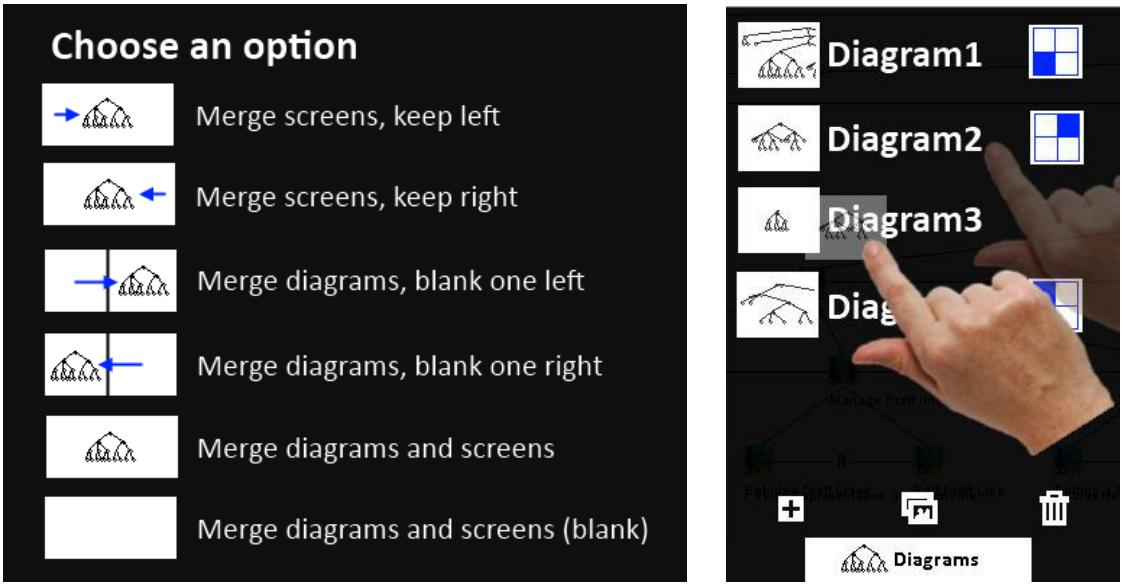


Abbildung 4.80 - Merge-Dialog bzw. Merge über Diagrams-Menü

Alternativ wäre es für den gleichen Zweck erforderlich, dass ein Merge über das Diagrams-Menü des Rollenmanagers durchgeführt werden kann. Dazu muss lediglich eins der beiden zu verschmelzenden Diagramme mit dem Finger ausgewählt und auf das Zieldiagramm geschoben werden (Abbildung 4.80, rechts), um die Verschmelzung nach einer Sicherheitsabfrage abzuschließen. Sollte das mit dem Finger aufgenommene Diagramm beim Kombinieren noch geöffnet gewesen sein, erhält der Benutzer dieses Teilbildschirms ein neues, leeres Diagramm, während sein zugehöriger Anteil im Zieldiagramm integriert wird. Dieser Vorgang macht besonders dann Sinn, wenn z. B. im Entwicklungsprozess der Fall entsteht, dass bei vier Teilnehmern auf diagonal entgegengesetzten Arbeitsbereichen zusammengehörige Diagrammteile entstanden sind, bei denen dann das Zusammenlegen über Merge nicht mehr greift.

Für beide Varianten wird noch einmal betont, dass bei der Verschmelzung keine Überlagerungen stattfinden und seitens der Anwendung automatisiert auf die Orientierung geachtet werden muss. Schlussendlich soll ein Anwendungsbeispiel die Verwendung des Konzepts stärker verdeutlichen, bevor abschließend die Zusammenfassung erfolgt.

4.4 Anwendungsbeispiel

In diesem Abschnitt soll einmal vorgestellt werden, wie die Zusammenarbeit mehrerer Benutzer für ein kooperatives Modell aussehen könnte (s. Abbildung 4.81). Als Grundlage dient wieder das Modell *GuitarSalesOrder.cctt*. Die Arbeitsfläche wurde mit Hilfe der Tear-Geste aufgeteilt, sodass vier Benutzer gleichzeitig arbeiten können. Während der Erstellung der Skizze ist deutlich geworden, dass zur besseren Orientierung für die Anwender der Name des jeweiligen Diagramms (oder der bereits verknüpften Rolle) halbtransparent innerhalb der Teilbildschirme eingeblendet werden sollte. Auf diese Weise ist direkt ersichtlich, welches Diagramm nun gerade angezeigt und bearbeitet wird. Dieser Aspekt wurde in die Abbildung eingearbeitet.

Zu sehen ist der Rollenmanager unten rechts, auf dem das kooperative Metamodell aufgebaut wird. Auf ihm sieht man, dass die Rollen *Sales Representative* sowie *Customer* bereits definiert sind, aber die Rolle Sales Representative noch keine oder noch zu wenige ConnectionTasks enthält, mit denen verknüpft werden kann (erkennbar an der roten Schrift und dem fehlendem Link Symbol). Dies rührt daher, dass die Erstellung des Modells für den Sales Representative auf die beiden gegenüberliegenden Flächen verteilt ist. Hier arbeiten zwei Personen an unterschiedlichen Teilbäumen des Diagramms, das

abschließend zusammengelegt wird. Es lässt sich sehr gut die Verwendung des Pie-Menüs erkennen (Diagram2, oben rechts), bei dem gerade die Art des Knotens „Terminate Query“ ausgewählt wurde. Weiterhin ist ein eingeklappter Knoten vorhanden, um kurzzeitigen Überblick herzustellen.

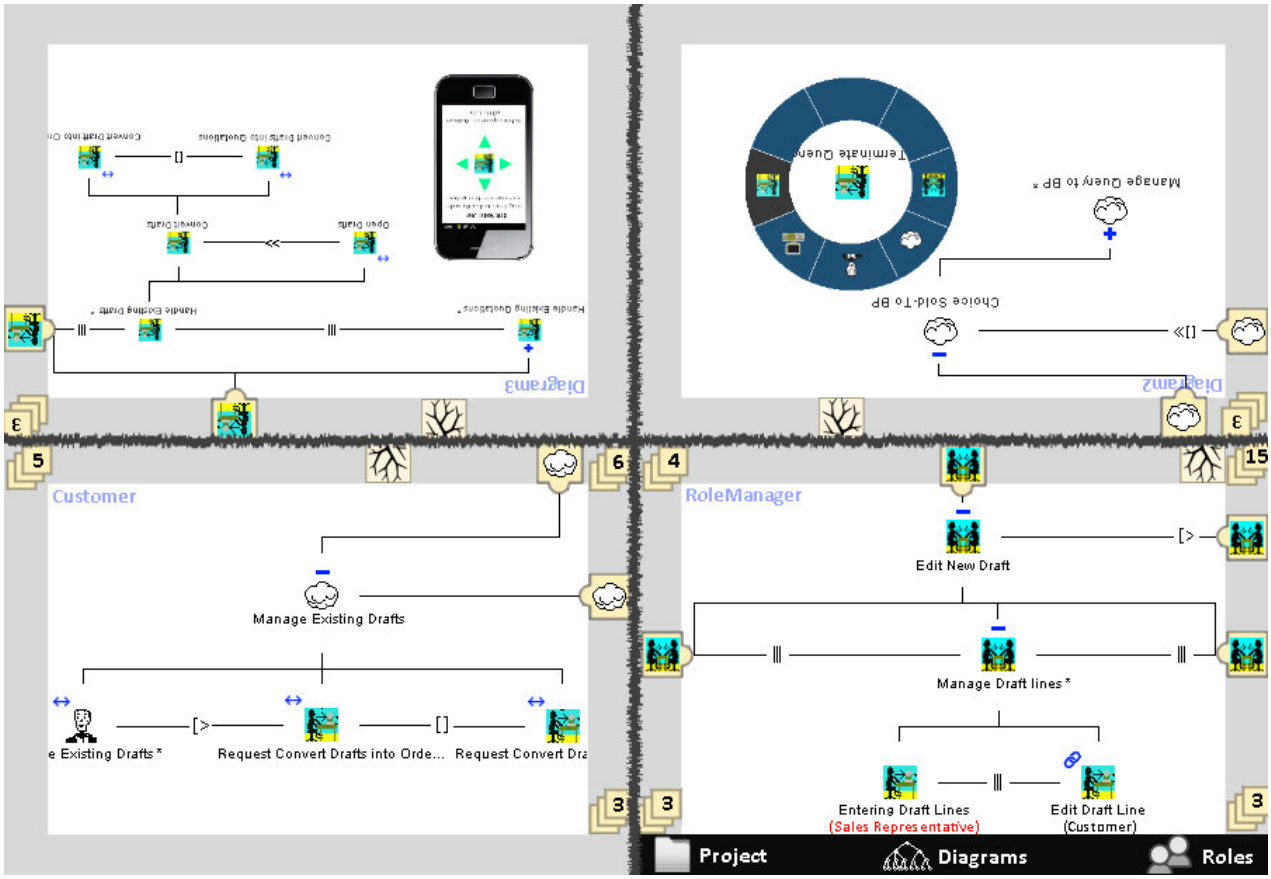


Abbildung 4.81 - Anwendungsbeispiel vier Benutzer (kooperatives Modell)

Auf der anderen Hälfte (Diagram3, oben links) wurde zur besseren Übersicht eine verkleinerte Zoomstufe des Baumes gewählt. Der Anwender benutzt sein Smartphone, um einen neuen Knoten auf der Oberfläche zu erzeugen. Nimmt er das Telefon hoch, wird der Knoten an dieser Stelle platziert. Bis zu diesem Zeitpunkt kann der Knoten durch Verschieben des Gerätes mit verschoben werden; danach kann die weitere Bearbeitung alternativ auch mit Gesten erfolgen.

Auf dem letzten verbleibenden Bildschirmteil entsteht durch den vierten Anwender die Rolle Customer (unten links). Dort sind sehr schön die ConnectionTasks zu erkennen, die vielleicht bereits im Metamodell rechts verknüpft sind. Der Knoten „Edit Draft Line“ deutet bereits mindestens eine Verknüpfung an, weil dort das Link Symbol aktiv ist und darunter die Rolle „Customer“ in schwarz angezeigt wird.

Für alle Bildschirmteile ist unabhängig voneinander die Off-Screen Visualisierung aktiv, die Elemente außerhalb des eingeschränkten Sichtbereichs mit Hilfe von Proxys zugänglicher macht. Beim Verschieben des Diagramms ändert sich der Randbereich entsprechend ab. Zoomstufen sind auf allen Teilen individuell einstellbar, und es können von allen Teilnehmern je nach Bedürfnis Smartphones, Stempel oder Gestik eingesetzt werden, um zu arbeiten. Diagramme können zusammengelegt werden, und eine Zuweisung der Diagramme zu Rollen findet innerhalb des Rollenmanagers statt, mit dem dann auch die Verknüpfung der ConnectionTasks vollzogen wird, die über die anderen Teilnehmer eingestellt wurden. Mit Hilfe dieses kleinen Beispiels ist schlussendlich die Verwendung des Konzepts etwas näher gebracht worden. Im Anschluss folgt nun die finale Zusammenfassung der Resultate.

5 Zusammenfassung und Ausblick

Abschließend soll ein letztes Mal zusammenhängend reflektiert werden, welche Aspekte im Verlauf der Konzeptentwicklung entstanden sind und welche Details in Bezug auf Darstellung, Interaktion und Kollaboration innerhalb der Domänen *Task Modelling* und *Interactive Displays* harmonisiert wurden.

Nachdem grundsätzliche Begriffe aus beiden Bereichen erläutert wurden, erfolgte eine Bestandsaufnahme heute existenter Werkzeuge, mit denen die Aufgabenmodellierung auf dem bisherigen Stand der Technik durchzuführen ist. Am Ende der Evaluation konnte CTTE als aussichtsreichstes Werkzeug identifiziert werden, um als Grundlage für das in dieser Arbeit entwickelte Konzept zu dienen. Die parallel untersuchten Werkzeuge leisteten trotzdem ihren Beitrag durch andere interessante Denkansätze, die an verschiedenen Stellen im Konzept mehr oder weniger Berücksichtigung fanden.

Im Anschluss daran erfolgte die Analyse gängiger Techniken für die Arbeit mit Diagrammen auf großen, berührungsempfindlichen Bildschirmen. Neben vorhandenen Gestensets für Node-Link Diagramme wurden aus dem Bereich der Interaktion zahlreiche Visualisierungs- und Explorationstechniken für die Multi-Touch Eingabe untersucht, die Potenziale für das Konzept boten. Als Stichworte sind hierbei Off-Screen Visualisierung sowie TouchPlucking, -Strumming und -Pinning erwähnenswert. Darauf folgend wurden würdige Vertreter der Tangible UIs auf Tauglichkeit überprüft und insbesondere Stempeltechniken sowohl statischer als auch dynamischer Natur für die Arbeit mit Diagrammen als vielversprechend festgestellt. Aus dem Bereich der Kollaboration mit anderen Teilnehmern in der Gruppe lag schlussendlich die Verwendung von Territorien als Aufteilung der Arbeitsfläche für das Konzept nahe.

Die so gewonnen Erkenntnisse wurden anschließend in einem Zwischenbericht kritisch hinterfragt und eine Auswahl der Elemente für die Verwendung im Konzept getroffen, die als geeignete Bausteine für den weiteren Verlauf empfunden wurden. Die Vorbereitungen waren somit abgeschlossen und der Grundstein für die Konzeptentwicklung gelegt. Schritt für Schritt wurde im Folgenden das Konzept etabliert.

Zu Beginn der Konzeptentwicklung stand zunächst die Definition von Darstellungsaspekten auf dem Programm. Nachdem sich die bereits übliche Baumstruktur manifestierte, wurden Vorschläge zur Anpassung der Symbolik unterbreitet und die Gestaltung des Diagramms durch den Einsatz rechtwinkliger Kanten und eines definierten Satzes an Zusatzsymbolen für Knoten weiter optimiert. Zusätzlich wurde ein eigenes Pie-Menü entwickelt, das für den Einsatz auf einem Touchscreen als ideal zu sehen ist und auf die Verwendung herkömmlicher Menügestaltung vollständig verzichtet. Abgerundet wurde dieser Abschnitt durch die Einführung notwendiger Erweiterungsfenster, um komplexere Funktionen an einem Knoten wahrnehmen zu können. Besonders die Entwicklung des Rollenmanagers stellt dabei einen essentiellen Schritt dar, weil dieser als die mächtigste Sicht innerhalb der verfügbaren Ansichten angesehen werden kann.

Als Folgeschritt wurde an Aspekten der Interaktion aufgezeigt, welche Möglichkeiten sich aus dem Bereich der Natural und Tangible UIs für die Gestaltung von Aufgabenmodellen zu Nutze gemacht werden können. Dabei wurde für die Gestik innerhalb der NUIs zunächst auf bewährte Bewegungsabläufe aus anderweitig bekannten Anwendungen zurückgegriffen und diese auf die Aufgabenmodellierung bzw. Diagrammerstellung übertragen. Spezielle Gesten wie Undo, Tear oder Grab wurden anschließend neben den herkömmlichen als sinnvolle Ergänzungen in das Konzept

integriert, sodass ein umfassendes Set entstand. Die aus dem Bereich der TUIs angeregten Stempeltechniken dienen als Unterstützung dieses Sets und können durch direkte Eingabemethoden den Erstellungsprozess teilweise beschleunigen. Ein Smartphone als dynamischer Träger verschiedener Knoten- und Kantentypen sowie zur Aufnahme von Knoten für die Eingabe textueller Informationen rundet die vorgestellte Interaktionsmethodik ab.

Im letzten Abschnitt lag der Fokus dann auf der gemeinsamen Erstellung der Modelle innerhalb eines zusammenhängenden Projektes durch verschiedene Teilnehmer. Aspekte der räumlichen und fachlichen Aufteilung spielten dabei gleichermaßen eine Rolle, um eine idealisierte Kollaboration während der Entwicklung zu gewährleisten. Dazu wurden verschiedene Schemata der Einrichtung von Territorien für die Belegung von zwei bis vier Benutzern diskutiert und Wege geschaffen, den Erstellungsprozess sinnvoll aufzutrennen sowie – je nach Komplexität des Modells – die einmal voneinander gelösten Diagrammteile sinnvoll und einfach wieder zusammenzufügen. Einzig der Vorgang der anschließenden Simulation der Aufgabenmodelle ist innerhalb dieser Arbeit noch nicht integriert worden, was jedoch auch nicht Teil der Aufgabenstellung war.

Die Verwendung und Gestaltung von User Interfaces unterliegt derzeit einem starken Wandel, sodass nicht genau vorhergesagt werden kann, in welche Richtung sich in der Zukunft die Schnittstellen zwischen Menschen und Computern bewegen. Ansätze dafür, jegliche Oberflächen als Basis für Eingaben mit den Fingern zu benutzen, werden bereits diskutiert (Stichwort: *Magic Finger* (Yang et al. 2012)) und erwecken bald nicht mehr den Eindruck der Science Fiction. Darüber hinaus ist es vielleicht in Zukunft ebenso möglich, Eingaben auch komplett ohne Berührung einer Oberfläche durchzuführen, wie es bereits im Bereich der Spielekonsolen schon länger der Vergangenheit angehört (Nintendo Wii, Microsoft Kinect, Sony Move). Ansätze für Tabletops sind dafür bereits vorhanden und bieten vielversprechende Möglichkeiten auf diesem Gebiet (Huppmann et al. 2012). Andererseits sind auch auf dem Gebiet der Sprachsteuerung große Fortschritte zu verzeichnen, sodass irgendwann einmal vielleicht die menschliche Stimme ausreicht, um Modelle auf einem Bildschirm zu erzeugen.

Entstanden ist die Vision von einer Aufgabenmodellierung, die den Anforderungen der heutigen Zeit gerecht wird. Die Verwendung von Natural und Tangible User Interfaces ist für die Gegenwart typisch und wird sich weiter als Standard etablieren, soviel ist sicher. Bereits jetzt wurde auf der CES 2013 in Las Vegas vom Hersteller *Lenovo* ein Table-PC („IdeaCentre Horizon 27“) für den Heimgebrauch vorgestellt (Lenovo 2013); ein Indiz dafür, dass für Tabletops der Schritt vom Labor ins heimische Wohnzimmer nicht mehr allzu weit weg ist. Dieser besitzt zwar nur eine physikalische Größe von 27“, aber mit der Zeit werden auch größere Dimensionen für den Heimanwender finanziell attraktiv – besonders wenn andere Hersteller in diesem Geschäftsbereich aufholen und miteinander konkurrieren.

Somit kann am Ende festgestellt werden, dass die zu Beginn formulierte Problemstellung im Zuge der Arbeit eine hinreichende Lösung fand und – im Sinne des Titels der Arbeit – ein ‚Konzept zur Erstellung von Aufgabenmodellen an einem interaktiven Display‘ erreicht wurde. Der nächste Schritt wäre nun die Implementierung des vorgeschlagenen Konzepts, sodass der Wandel von der Vision zur Realität vollzogen werden kann.

6 Literaturverzeichnis

Ardaiz, O. et al., 2010. Virtual collaborative environments with distributed multitouch support. *Proceedings of the 2nd ACM SIGCHI symposium on Engineering interactive computing systems - EICS '10*, p.235.

Bollhoefer, K.W. et al., 2009. White Paper Microsoft Surface und das Natural User Interface (NUI).

Dachselt, R., Frisch, M. & Decker, E., 2008. Enhancing UML sketch tools with digital pens and paper. *Proceedings of the 4th ACM symposium on Software visuallization - SoftVis '08*, p.203.

Dorau, R., 2011. *Emotionales Interaktionsdesign*, Berlin, Heidelberg: Springer Berlin Heidelberg.

Forschungsportal Sachsen-Anhalt, 2012. Diagramm-interaktion, Node-link, Off-screen Visualisierung. , pp.2011–2012.

Frisch, M. & Dachselt, R., 2010. Off-screen visualization techniques for class diagrams. *Proceedings of the 5th international symposium on Software visualization - SOFTVIS '10*, p.163.

Frisch, M., Heydekorn, J. & Dachselt, R., 2009. Diagram editing on interactive displays using multi-touch and pen gestures. *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces ITS 09 (2009)*.

Huppmann, D. et al., 2012. Exploring and Evaluating the Combined Multi-Touch and In-the-Air Tabletop Interaction Space. , p.12.

K-MADe, 2012. K-MADe - an user task oriented editor. , p.2012. Available at: http://kmade.sourceforge.net/.

Kaltenbrunner, M., 2009. reacTIVision and TUIO : A Tangible Tabletop Toolkit. , pp.9–16.

Kaltenbrunner, M. & Bencina, R., 2007. reacTIVision : A Computer-Vision Framework for Table- Based Tangible Interaction. , pp.15–17.

Lenovo, 2013. IdeaCentre Horizon | Table-PC with 27". , 8, pp.1–2.

Marco, J. et al., 2009. Bringing Tabletop Technologies to Kindergarten Children. *HCI 2009 – People and Computers XXIII – Celebrating people and technology*, pp.103–111.

Marco, J., Cerezo, E. & Baldassarri, S., 2012. ToyVision : A Toolkit for Prototyping Tabletop Tangible Games. , pp.71–80.

Montero, F. & López-Jaquero, V., 2007. IdealXML: an interaction design tool. *Computer-Aided Design of User Interfaces …*, pp.245–252.

Müller-Tomfelde, C., 2010. *Tabletops - Horizontal Interactive Displays*, Springer London.

Ott, M. et al., 2012. Enhancing Interactive Tabletop Workspaces with Tangible Contexts. In *Mensch & Computer 2012: interaktiv informiert – allgegenwärtig und allumfassend!?* pp. 53–62.

Paternò, F., 2001. ConcurTaskTrees: An Engineered Approach to Model-based Design of Interactive Systems. *ISTI-CNR, Pisa*, pp.1–18.

Raffle, H. et al., 2007. Jabberstamp : Embedding sound and voice in traditional drawings.

Reactable Systems S.L., 2009. Fiducial Symbols for reacTIVision.

Schmidt, S. et al., 2010. A set of multi-touch graph interaction techniques. *ACM International Conference on Interactive Tabletops and Surfaces - ITS '10*, p.113.

Scott, S.D., 2003. Territory-based interaction techniques for tabletop collaboration. *Conference Companion of the ACM Symposium on …*, pp.3–6.

Seifried, T. & Scott, S.D., 2012. Regional Undo/Redo Techniques for Large Interactive Surfaces. , pp.2855–2864.

Szwillus, G., 2011. Task Models in the Context of User Interface. In pp. 277–302.

Szwillus, G., 2012. User Interface Modelling. *Universität Paderborn Fakultät EIM , Institut für Informatik*, p.230.

Tse, E. et al., 2007. Multimodal Split View Tabletop Interaction Over Existing Applications. *Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP'07)*, pp.129–136.

Tuddenham, P. & Robinson, P., 2009. Territorial coordination and workspace awareness in remote tabletop collaboration. *Proceedings of the 27th international conference on Human factors in computing systems - CHI 09*, p.2139.

Vanderdonckt, J. & Simarro, F.M., 2010. Generative pattern-based design of user interfaces. *Proceedings of the 1st International Workshop on Pattern-Driven Engineering of Interactive Computing Systems - PEICS '10*, pp.12–19.

Yamashita, N. et al., 2011. Supporting fluid tabletop collaboration across distances. *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*, p.2827.

Yang, X. et al., 2012. Magic Finger : Always-Available Input through Finger Instrumentation. , pp.147–156.

7 Anhang

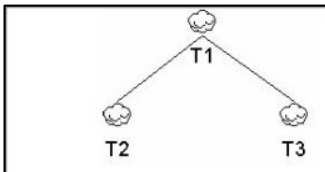
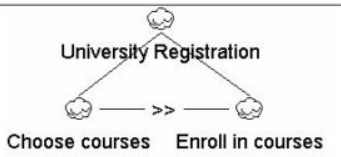
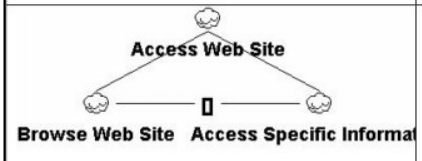
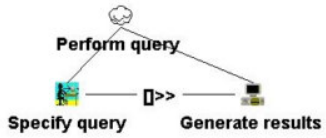
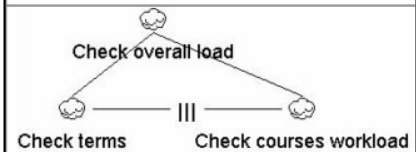
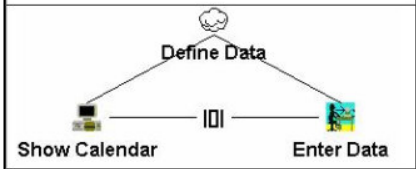
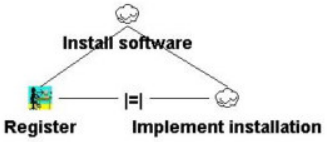
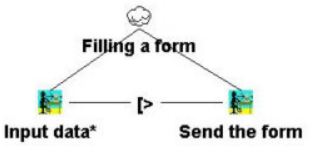
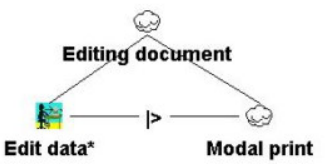
	**Hierarchy** Tasks at same level represent different options or different tasks at the same abstraction level that have to be performed. Read levels as "In order to do T1, I need to do T2 and T3", or "In order to do T1, I need to do T2 or T3"
	**Enabling** Specifies second task cannot begin until first task performed. Example: I cannot enroll at university before I have chosen which courses to take.
	**Choice** Specifies two tasks enabled, then once one has started the other one is no longer enabled. Example: When accessing a web site it is possible either to browse it or to access some detailed information.
	**Enabling with information passing** Specifies second task cannot be performed until first task is performed, and that information produced in first task is used as input for the second one. Example: The system generates results only after that the user specifies a query and the results will depend on the query specified.
	**Concurrent tasks** Tasks can be performed in any order, or at same time, including the possibility of starting a task before the other one has been completed. Example: In order to check the load of a set of courses, I need to consider what terms they fall in and to consider how much work each course represents
	**Concurrent Communicating Tasks** Tasks that can exchange information while performed concurrently Example: An application where the system displays a calendar where it is highlighted the data that is entered in the meantime by the user.
	**Task independence** Tasks can be performed in any order, but when one starts then it has to finish before the other one can start. Example: When people install new software they can start by either registering or implementing the installation but if they start one task they have to finish it before moving to the other one.
	**Disabling** The first task (usually an iterative task) is completely interrupted by the second task. Example: A user can iteratively input data in a form until the form is sent.
	**Suspende-Resume** First task can be interrupted by the second one. When the second terminates then the first one can be reactivated from the state reached before Example: Editing some data and then enabling the possibility of printing them in an environment where when printing is performed then it is no possible to edit.

Abbildung 7.1 - Paternò: temporale Relationen[63]

[63] Quelle: (Paternò 2001), S. 8-9

- *sequence* denotes that first A and then B have to be performed to achieve T,
- *random sequence* denotes that A and B have to be performed one after the other to achieve T, but the order is irrelevant,
- *parallel* means A and B have to be performed simultaneously, meaning that there must be at least one point in time where A and B are both "running" at the same time,
- *unrestricted* means that A and B have to be performed, but there is no restriction whatsoever about their mutual timing, and
- *selection* means that only one of the subtasks needs to be done to perform T.

In addition, a task can be specified to be

- *optional*, hence it can, but need not, be executed,
- or an *iteration*, hence it can be repeated several times.

Figure 2 contains some additions specifying the necessary sequencing rules, and including the optional addition of a photograph to the letter, which can either be put into the envelope as first or as second step.

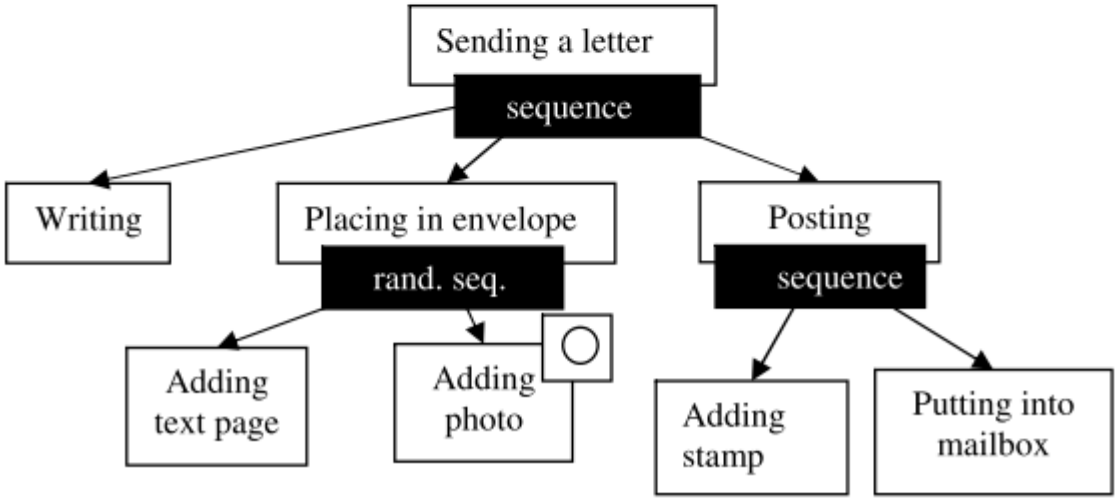


Fig. 2 Some temporal relations added to the "Sending a letter" task tree

Abbildung 7.2 - Szwillus: temporale Relationen[64]

[64] Quelle: (Szwillus 2011), S. 279

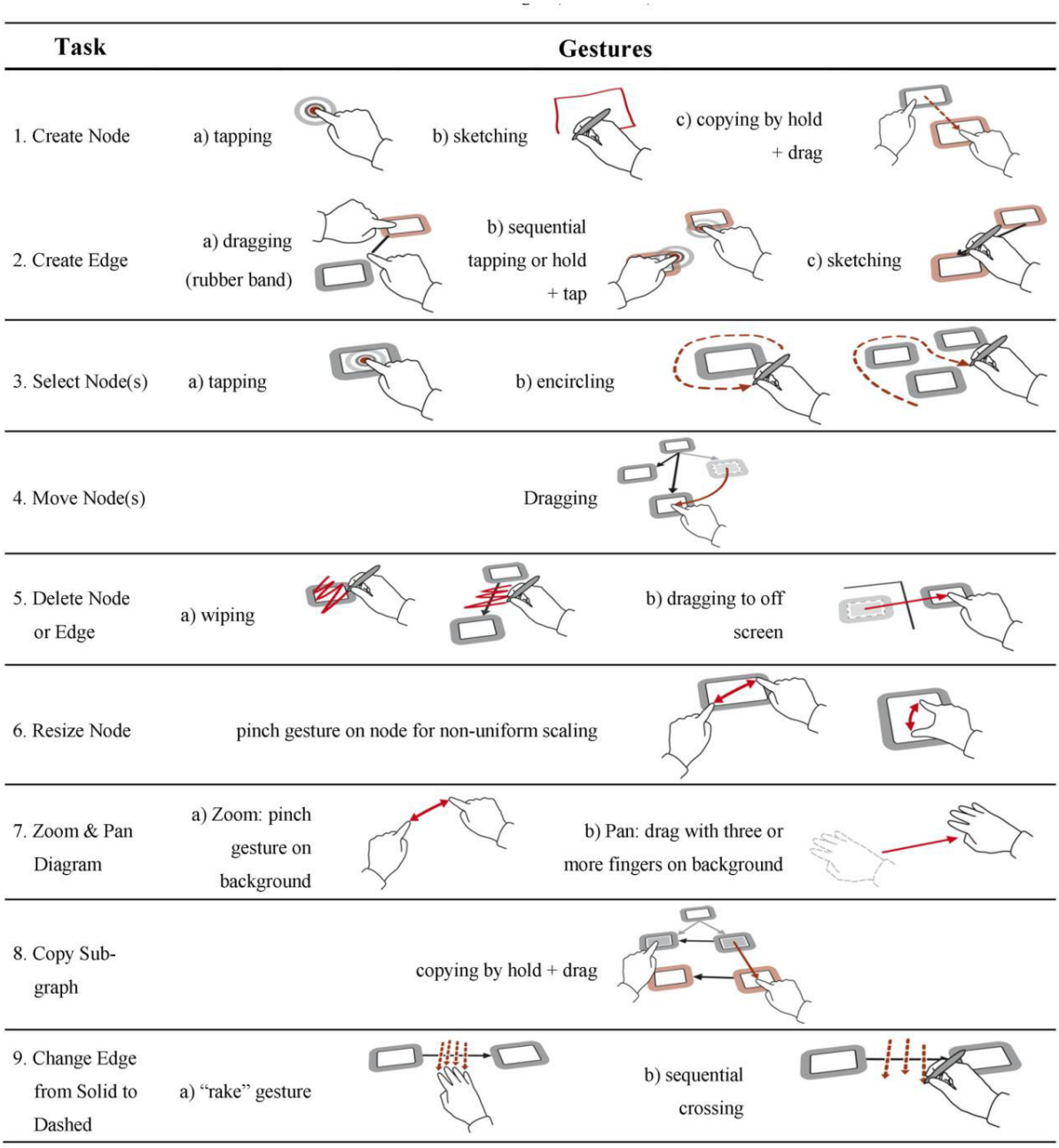


Abbildung 7.3 - Gesture Set Sketch-based diagram creation[65]

[65] Quelle: (Frisch et al. 2009), S. 8