Visuelle Informationsdarstellung in Smart Environments

Dissertation
zur
Erlangung des akademischen Grades
Doktor-Ingenieur (Dr.-Ing.)
der Fakultät für Informatik und Elektrotechnik
der Universität Rostock

vorgelegt von
Conrad, Thiede, geb. am 04.12.1979 in Kühlungsborn
aus Rostock

Rostock, 18. Januar 2011

urn:nbn:de:gbv:28-diss2011-0017-6

Gutachter

Prof. Dr. Heidrun Schumann, Universität Rostock
Prof. Dr. Peter Forbrig, Universität Rostock
Prof. Dr. Pavel Slavík, Technische Universität Prag

Verteidigung

Die Dissertation wurde am 08.10.2010 verteidigt.

Zusammenfassung

Bei der Analyse von Daten sind *visuelle Repräsentationen* der Informationen ein etabliertes Hilfsmittel. Bisher wurde diese in der Regel für eine bestimmte Ausgabeumgebung erzeugt. Heute verwendet man aber zunehmend Multi-Display-Umgebungen wie *Smart-ad-hoc-Environments* zur Ausgabe der visuellen Repräsentation. Diese zeichnen sich durch selbstständige Aktionen und eine Dynamik der verwendeten Geräte aus. Diese Eigenschaften stellen eine wesentliche Herausforderung dar, da die visuelle Ausgabe *On-the-Fly* an die Geräte angepasst werden muss.
Ziel dieser Dissertation ist daher die Entwicklung von Problemlösungen für eine adaptive Informationsvisualisierung in *Smart-ad-hoc-Environments* unter besondere Berücksichtigung der Gerätedynamik.

Dafür sind in erster Linie verschiedene *Adaptionsmechanismen* erforderlich, die den Erzeugungsprozess der visuellen Repräsentation an unterschiedliche Ausgabegeräte anpasst. Die entwickelten *Mechanismen* basieren auf einem Operator Pipeline Modell, welches zur Generierung der visuellen Repräsentationen genutzt wird. Zur Auswahl geeigneter *Adaptionsmechanismen* aus dem Datenraum, Darstellungsraum oder dem Mapping – entsprechend der Stufen des Pipeline Modells – wurde ein Konzept auf der Basis eines Entscheidungsbaums entworfen.

Wird das Operator Pipeline Modell verwendet, kann einer Adaption auf der gesamten visuellen Repräsentation erfolgen oder nur auf einem lokal begrenzten Bereich. Letzteres wird durch das Konzept der *Intelligente Linsen* realisiert. *Intelligente Linsen* können als Filter aufgefasst werden, die sich auf allen Stufen der Pipeline einsetzen lassen und eine lokale Modifikation des Generierungsprozesses erlauben. Dabei können prinzipiell komplexe Linsen als Kombinationen von einfachen Linsen realisiert werden. Somit stellen *intelligente Linsen* ein hoch flexibles Werkzeug zur lokalen Adaption dar.
Weiterhin kann zwischen räumlichen und zeitlichen Adaptionen unterschieden werden. Als ein Repräsentant einer zeitlichen Adaption wurde die progressive Informationsdarstellung untersucht. Dabei werden bei der Adaption der visuellen Repräsentation an die unterschiedlichen Ausgabegeräte von diesen nur so viele Daten verarbeitet und ausgegeben, wie sie auch darstellen können.
Zur Realisierung der Konzepte wurde ein Framework auf der Basis einer serviceorientierten Architektur entwickelt, welches das ad-hoc Verhalten der Geräte berücksichtigt und dabei die visuelle Repräsentation an unterschiedliche Geräte *On-the-Fly* anpasst.

Abstract

The visual representation of information has become a well-established tool for data analysis. So far, such representations have generally been created for a specific output device. Nowadays, multi-display environments like smart ad-hoc environments see increasing use for data analysis activities. They are characterised by autonomous actions and a dynamic set of used devices. These properties present a major challenge since visual output must be adapted to different devices on-the-fly.
This dissertation's objective is therefore to develop solutions for adaptive information visualisation in smart ad-hoc environments with particular regard to the device set's dynamic.

This primarily requires several adaption mechanisms to adapt the generating process of a visual representation according to the requirements of different output devices. The mechanisms proposed in this thesis are based on an operator pipeline model of the generation process. To select appropriate adaption mechanisms for data space, visual mapping or view space – corresponding to subsequent pipeline stages –, a concept for a suitable decision tree has been developed.
Using the operator pipeline model, adaption can be performed on the entire visual representation or only within a locally constrained region. The latter is enabled by the concept of Smart Lenses. Smart Lenses can be seen as pluggable filters that allow to locally modify the representation generation process on all pipeline stages, whereas complex modifications can be attained by combination of several simple lenses. Thus smart lenses are a highly flexible tool for local adaption.
Furthermore, spatial and temporal adaption can be distinguished. As a representative for temporal adaption, progressive information visualisation was examined. Thereby, different output devices process and show only as much data as they can display effectively.

To substantiate the practical applicability of the proposed concepts, a framework based on service-oriented architecture was developed. This framework accounts for the devices' ad-hoc properties, thus facilitating on-the-fly adaptation of visual representations to various devices in a smart ad-hoc environment.

Key-Words: Information Visualization, Smart ad hoc Environments, Service-Oriented Architecture, Device Adaptation, Multi-Display-Environment

Danksagung

Diese Dissertation wurde im Rahmen des Graduiertenkollegs MuSAMA (***Mu****ltimodal* ***S****mart* ***A****ppliance Ensembles for* ***M****obile* ***A****pplications*) durch die Deutsche Forschungsgemeinschaft (DFG) gefördert. Durch diese finanzielle Unterstützung konnte diese Arbeit erst entstehen.
Bedanken möchte ich mich vor allem bei meiner Betreuerin Frau Prof. Heidrun Schumann, die mich während des Studiums und der Zeit als Promotionsstudent kritisch beraten, unterstützt und motiviert hat. Ausdrücklich möchte ich mich auch bei den beiden Gutachtern Herrn Prof. Peter Forbrig und Herrn Prof. Pavel Slavík für ihre Bereitschaft bedanken, diese Arbeit zu begutachten.
Außerdem möchte ich mich bei Georg Fuchs für die vielen konstruktiven Gespräche und Vorschläge bedanken sowie bei allen anderen Kollegen am Lehrstuhl für Computergraphik, die mich in dieser Zeit bei meiner Arbeit unterstützten. Gleichzeitig bedanke ich mich bei allen Studenten, die mit ihren Studien- und Diplomarbeiten zum Gelingen dieser Arbeit beitrugen.
Ein herzlicher Dank richtet sich auch an meine beiden Korrekturleser, die zahlreiche Stunden für diese Arbeit geopfert haben, an Stephan Auner für das Überarbeiten der Grafiken, an meinen Eltern, die mich stets unterstützt und in meinem Vorhaben bestärkt haben und natürlich an meine Frau, Anne Thiede, die mir in den letzten Jahren immer einen Rückhalt gab und Unterstützung leistet und viel Verständnis für meine Arbeit aufbrachte.

Inhaltsverzeichnis

Kapitel 1

Einleitung und Motivation

1.1 Motivation und Zielstellung

In unserer heutigen Informationsgesellschaft nimmt die Größe der Datensätze ständig zu. So sendet beispielsweise der Earth-Orbiting-Satellite täglich mehrere Terabyte an Daten zur Erde [SML04]. An der New Yorker Börse wurden bereits 1995 täglich durchschnittlich 300 Millionen Transaktionen abgewickelt [New95]. Aus den akquirierten Daten müssen die relevanten Informationen extrahiert werden. Dies ist durch eine numerische Darstellung der Daten nur schwer zu realisieren. Es sind also neue Strategien erforderlich, um eine effiziente Verarbeitung dieser immensen Datenmengen zu gewährleisten.
Im umrissenen Problemfeld, der Analyse großer Datenmengen, sind *visuelle Repräsentationen* der Informationen ein etabliertes Hilfsmittel. Die visuelle Repräsentation wird schrittweise aus Rohdaten erzeugt. Hierbei werden die für den Benutzer relevanten Informationen visuell veranschaulicht und sind dadurch intuitiv wahrnehmbar.
Bisher wurden *visuelle Repräsentationen* von Daten in der Regel für einen Nutzer und eine bestimmte Ausgabeumgebung erzeugt. Dieses Szenario kann als *Single-Display-* und *Single-User-Umgebung* beschrieben werden. Heute verwendet man zunehmend mehrere visuelle Ausgabegeräte gleichzeitig und verknüpft diese miteinander [AE06, WLSS09]. In aktuellen Untersuchungen der Informationsvisualisierung werden deshalb neben der *Single-Display/Single-User-Umgebung* zunehmend auch die *Multi-Display/Multi-User-Umgebungen* berücksichtigt [WLSS09].
Smart-ad-hoc-Environments [CD07] erweitern die *Multi-Display-Umgebungen*, um zwei wichtige Aspekte *„Smart“* und *„ad-hoc“*. Der Aspekt *Smart* beschreibt in diesem Zusammenhang die Fähigkeit, selbständig Daten zu sammeln und anzuwenden [CD07] und stellt die vorrangige Herausforde-

rung in Smart Environments dar. Dafür verfügen *Smart Environments* neben den klassischen Bestandteilen einer *Multi-Display-Umgebung* zusätzlich über Sensoren. Die Sensordaten und geeignete Analyse- und Vorhersagemethoden ermöglichen einer *Smart-Environment* auf die Situation angepasst zu reagieren.
Der Begriff *ad-hoc* impliziert eine gewisse Dynamik im System. In dieser Arbeit wird die Dynamik der Geräte betrachtet. Dynamische Geräte sind dadurch charakterisiert, dass sie jederzeit zu den bereits in der Umgebung vorhandenen Geräten hinzukommen können oder diese wieder verlassen. Das stellt eine weitere Herausforderung für die Informationsdarstellung in *Smart-ad-hoc-Environments* dar, da die visuelle Ausgabe *On-the-Fly* an aktuelle Geräte angepasst werden muss.
Die besonderen Eigenschaften der *Smart-ad-hoc-Environments* erfordern neue Konzepte für die Informationsvisualisierung. Dieses Problem ist bisher in der gängigen Literatur kaum thematisiert worden.
Das Ziel dieser Dissertation ist deshalb die Entwicklung von Problemlösungen für eine adaptive Informationsvisualisierung in *Smart-ad-hoc-Environments* unter besonderer Berücksichtigung des ad-hoc Charakters.

Aus der formulierten Zielstellung ergeben sich für diese Arbeit zwei Schwerpunkte:
Entwicklung neuer Adaptionsmechanismen: In erster Linie sind *Adaptionsmechanismen* notwendig, die den Erzeugungsprozess der visuellen Repräsentationen an unterschiedliche Ausgabegeräte anpassen.
Stehen unterschiedliche *Adaptionsmechanismen* zur Auswahl, muss entschieden werden, welche von diesen in der gegebenen Situation am besten geeignet sind. Es ist ein Konzept zu entwickeln, wie automatisch adäquate Anpassungen durchgeführt werden können.
Diese *Adaptionsmechanismen* können sich sowohl auf die Anpassung der gesamten visuellen Repräsentation als auch auf die Veränderung lokal begrenzter Bereiche beziehen. Außerdem lässt sich die Adaption sowohl durch räumliche Anordnung als auch durch zeitliche Anordnung, insbesondere durch eine progressive Darstellung erreichen.
Konzipierung und Umsetzung eines Frameworks zur dynamischen Informationsdarstellung in Multi-Display-Umgebung: Zur Umsetzung der entwickelten Konzepte sind der Entwurf und die Implementierung eines **Frameworks** erforderlich, das als Software-technische Basis zur Informationsdarstellung in Smart-ad-hoc-Environments genutzt werden kann und die entwickelten *Adaptionsmechanismen* bereitstellt. Flexibilität, Erweiterbarkeit und Robustheit sind wesentliche Eigenschaften des **Frameworks**. Insbesondere muss dabei die in *Smart-ad-hoc-Environments* gegebene Dyna-

mik der Geräte berücksichtigt werden.

Die vorliegende Dissertation ist in das Umfeld des GRK MuSAMA (***Mu**ltimodal **S**mart **A**ppliance Ensembles for **M**obile **A**pplications*) [MuS09] eingebettet. MuSAMA untersucht grundlegende Forschungsfragen, die in Smart-ad-hoc-Environments und hierbei speziell in *Smart-Meeting-Rooms* [BRH+08] eine Rolle spielen. *Smart-Meeting-Rooms* haben die Aufgabe, Meeting von mehreren Personen zu unterstützen. Bisher hierfür beschriebene Szenarien sehen Vorträge, aber auch wissenschaftliche Diskussionen als Anwendungen vor [EK05]. Im Rahmen der hier vorliegenden Arbeit wird das Problem der Informationsdarstellung in *Smart-Meeting-Rooms* behandelt, wobei sich zwangsläufig auch Querverbindungen zu den Themenstellungen anderer Stipendiaten ergeben. Darauf wird im Verlauf der Arbeit noch genauer eingegangen.

1.2 Ergebnisse und Struktur

Zu den in Abschnitt 1.1 genannten Zielstellungen wurden in dieser Dissertation folgende Lösungsansätze erarbeitet: Das Hauptziel ist die **automatische Adaption** von Informationsdarstellungen an heterogene, dynamische Ausgabegeräte.
Dafür wurden unterschiedliche *Adaptionsmechanismen* erarbeitet. Um zu entscheiden, wann welcher *Adaptionsmechanismus* eingesetzt wird, ist ein Konzept auf der Basis eines Entscheidungsbaums entwickelt worden. Dadurch können die einzelnen *Adaptionsmechanismen* angepasst an die aktuelle Situation ausgewählt werden (vgl. Abbildung 1.1). Allgemein betrachtet ergibt sich folgender Entscheidungsablauf: Zu Beginn wird ermittelt, ob es sich um eine effektive Visualisierung handelt. Ist das nicht der Fall, so ist eine Adaption erforderlich. Je nach Größe der Displayfläche und insbesondere im Bezug dazu, in welchem Maße sie von der Displayfläche des bisherigen Ausgabegerätes abweicht, werden unterschiedliche *Adaptionsmechanismen* ausgewählt.
Die einzelnen Zweige des Entscheidungsbaums erfordern unterschiedliche Vorgehensweisen für die Adaption, die im Rahmen der vorliegenden Arbeit exemplarisch umgesetzt wurden.
So wurde ein Konzept zur Adaption im Datenraum (abstrakte Daten) erarbeitet, das auf einem hierarchischen Clustern (vgl. [ED06b]) basiert. So können im Erzeugungsprozess die zu verarbeitenden Daten frühzeitig reduziert werden. Dies ist besonders dann ein Vorteil, wenn das Ausgabegerät nicht in der Lage ist, alle Informationen darzustellen oder es zu leistungs-

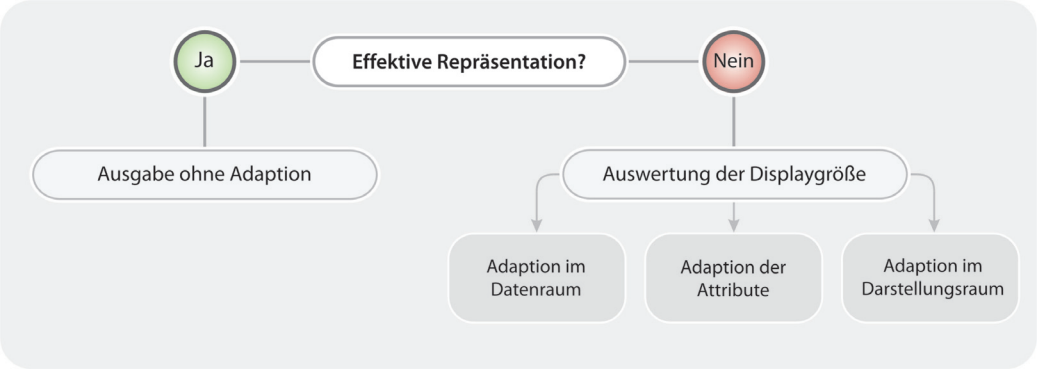

Abbildung 1.1: Allgemeiner Entscheidungsbaum zur Auswahl von adäquaten Adaptionsmechanismen in Abhängigkeit des aktuellen Ausgabegerätes.

schwach ist, um alle Daten zu verarbeiten.
Um statt der Daten die Darstellung selbst zu adaptieren, wurde das *Binning* (vgl. [NH06]) untersucht und für die Anwendung auf Scatterplots erweitert. Dadurch wird eine Vereinfachung der Darstellung vorgenommen und ein überladenes Display (zu viele Daten auf zu kleiner Displayfläche) wird vermieden. Die Ergebnisse zur automatischen Adaption wurden auf dem ersten IEEE CoVIS Workshop auf der VisWeek 2009 veröffentlicht [FTSS09].

Die obigen *Adaptionsmechanismen* beeinflussen die gesamte visuelle Repräsentation. Dies ist je nach Kontext nicht immer notwendig oder sinnvoll. Daher wurde ein allgemeines Konzept einer automatischen, lokal begrenzten Adaption des Erzeugungsprozesses der Informationsdarstellung erarbeitet. Das Konzept basiert auf der Metapher der Linsen. Bei Linsen handelt es sich um lokal begrenzte Filterfunktionen [BSP+93, KR97, ED06b, GFS05]. Die Filterfunktion verändert nur die Darstellung im Linsenbereich. Der Bereich außerhalb der Linse bleibt unverändert.
Je nach Situation kann die Filterfunktion verändert werden. Der Vorteil besteht darin, dass die Berechnung der Adaption aufgrund der geringeren zu bearbeitenden Datenmengen extrem schnell durchgeführt werden kann. Dieser Vorteil wiegt bei leistungsschwachen Geräten besonders schwer. Die grundlegend neue Herangehensweise in dieser Arbeit ist die Verallgemeinerung dieses Konzeptes, so dass Linsen auf allen Stufen des Erzeugungsprozesses der visuellen Repräsentation definiert und beliebig miteinander kombiniert werden können. Dadurch lässt sich ein hoch flexibles Werkzeug zur lokalen Adaption einer visuellen Repräsentation realisieren. Die Ergebnisse zu diesem Teilaspekt wurden auf der Smart Graphics 2008 [TFS08a] sowie als interaktives Poster auf der IEEE InfoVis 2007 [FTS07] veröffentlicht.

Neben der Adaption ist die Progression ein weiteres wichtiges Konzept, das aber in der Informationsvisualisierung bisher kaum angewendet wird. Progression bedeutet, eine visuelle Repräsentation schrittweise zu übertragen, angefangen mit einem groben Überblicksbild bis hin zum Original mit allen Details.
Die Idee besteht nun darin, für jedes Ausgabegerät jeweils nur so viele Daten zu verarbeiten und auszugeben, wie das Gerät auch darstellen kann. Dazu wird für jedes Gerät ein individueller Datenstrom aus einmal vorberechneten Daten zusammengestellt. Dadurch wird der Benutzer solange mit Vorschauen, die schrittweise verfeinert werden, versorgt, bis die höchste Detailstufe übertragen ist. So bekommt der Benutzer bereits zu einem sehr frühen Zeitpunkt einen groben Überblick über die Daten. Dieses Konzept wurde auf der IMC 2009 veröffentlicht [TSR09].

Die gesamten *Adaptionsmechanismen* wurden in einem allgemeinen und erweiterbaren **Framework** zur Erzeugung von visuellen Repräsentationen in *Smart-ad-hoc-Environments* zusammengeführt.
Um das ad-hoc-Verhalten der Geräte zu berücksichtigen, verwendet das Framework eine serviceorientierte Architektur. Es wurde ein Konzept entwickelt, das es erlaubt, die am Visualisierungsprozess beteiligten Services hinzuzufügen, zu entfernen und auszutauschen. Das Framework arbeitet auf der Basis des Data-State-Reference-Models (DSRM) (vgl. [CR98, Chi00]). Hierdurch ist es möglich, den Visualisierungsprozess an unterschiedliche Geräte und die aktuelle Situation anzupassen. Das Konzept wurde auf der IEEE Information Visualization, IV'09, veröffentlicht [TTS09].

Die vorliegende Arbeit ist in sechs Kapitel gegliedert. Zunächst werden in **Kapitel 2** die notwendigen Grundlagen beschrieben. Das Kapitel beginnt mit der Definition der grundlegenden Begriffe in *Smart-ad-hoc-Environments*. Im Anschluss daran werden deren Eigenschaften und aktuelle Anwendungen vorgestellt. Es folgt eine kurze Einführung in die Informationsvisualisierung. Besonders wird hierbei auf das Data-State-Reference-Model eingegangen. Anschließend werden die verschiedenen Einflussfaktoren auf den Prozess der Informationsvisualisierung untersucht und eingeordnet. Nachfolgend wird die gängige Literatur zu verteilter Visualisierung beleuchtet. Einen Schwerpunkt bilden dabei die *Service-orientierten* und *Agenten-basierten* Ansätze.
Kapitel 3 beinhaltet die Problem- und Anforderungsanalyse. Zu Beginn werden die offenen Probleme aufgezeigt. Daraus werden die Anforderungen von Informationsvisualisierungen in *Smart-ad-hoc-Environments* abgeleitet. Dabei stehen die automatische und dynamische Adaption der Informations-

visualisierung an unterschiedliche Ausgabegeräte im Vordergrund.
In **Kapitel 4** werden die neuen Konzepte zur adaptiven Informationsdarstellung in *Smart-ad-hoc-Environments* eingeführt. Hierzu gehören die verschiedenen *Adaptionsmechanismen* sowie das neue Konzepte der Progression für die Informationsanzeige in *Smart-Meeting-Rooms*. Außerdem wird ein Konzept vorgestellt, welches die verteilten Geräte in einer *Smart-ad-hoc-Environment* berücksichtigt.
Kapitel 5 behandelt das allgemeine *Visualisierungsframework*, das als Basis für die rechentechnische Umsetzung der in Kapitel 4 erarbeiteten Konzepte dient. Wesentliche Eigenschaften des *Frameworks* sind Adaptivität, Flexibilität, Erweiterbarkeit und die Unterstützung eines dynamischen Geräteensembles mit unterschiedlichen Ausgabedisplays.
Den Abschluss bildet **Kapitel 6** mit einer Zusammenfassung und einem Ausblick auf weiterführende Arbeiten.

Kapitel 2

Grundlagen und Stand der Forschung

2.1 Smart-ad-hoc-Environments

2.1.1 Begriffsklärung und Eigenschaften

Seit Jahrzehnten besteht der Wunsch, Smart Environments zu schaffen [CD05]. Smart Environments sind physikalische Räume, welche auf die Aktivitäten der Benutzer reagieren und daher die Benutzer in ihrer aktuellen Situation unterstützen [AE06].
Dazu sammeln sie Informationen über die Benutzer und die Umgebung [CD07, YHHC05]. Diese Informationen werden durch die Smart Environments verarbeitet, um die von den Benutzern angestrebte Situation zu ermitteln und anschließend eine Adaption des aktuellen Zustands der Umgebung hin zum angestrebten Zustand durchzuführen [CD07, YHHC05, RMSF09]. Smart Environment ist der Oberbegriff für eine ganze Reihe von intelligenten Umgebungen mit unterschiedlichen Aufgaben. So gibt es beispielsweise *Smart-Meeting-Rooms* [EK05, HK05a, AE06], Smart Offices [GMLC01, AHF+02, RMSF09], Smart Conference Rooms [HK05a] oder Smart Class Rooms [Abo99, FFH00, SXX+03]. Aufgrund des durch das GRK MuSAMA [MuS09] vorgegebenen Leitszenarios eines *Smart-Meeting-Rooms* wird sich diese Arbeit darauf konzentrieren.

Smart Environments im Allgemeinen und *Smart-Meeting-Rooms* im Speziellen bestehen aus einer Vielzahl von einzelnen verteilten Geräten, welche sich selbstständig zu einem zusammenhängend agierenden *Geräteensemble* zusammenfinden [HK05b, BMK+00]. Roard und Jones stellen die Behauptung auf, dass bei verteilten Visualisierungssystemen *heterogene Hardware*

berücksichtigt werden muss; sie schließt die Geräte für die Berechnung und die Geräte für die Ausgabe ein [RJ06]. Auch Sousa et al. [SG02] gehen davon aus, dass die Benutzer mehr und mehr von heterogener Technologie umgeben sind. Auch im Umfeld der Visualisierung wird diese These bestätigt. Brodlie et al. setzen voraus, dass Benutzer, die zusammenarbeiten, auch ihre jeweiligen Geräte benutzen sollten, mit denen sie vertraut sind [BDG+04a]. Daher soll auch an dieser Stelle von einem heterogenen Geräteensemble ausgegangen werden. Zur Abgrenzung des Begriffs *heterogen* soll Tantawi und Towsley [TT85] herangezogen werden. Sie sprechen von heterogenen Geräten, wenn diese über unterschiedliche Eigenschaften verfügen.
Die Gerätestruktur in einem *Smart-Meeting-Room* ist von zentraler Bedeutung. Bauer et al. [BBR02] zufolge wird zwischen einer gegebenen Gerätestruktur und einer Gerätestruktur differenziert, die sich *ad-hoc* bildet. Zudem gibt es auch die Kombination aus beiden Varianten, bei der ein Teil der Geräte bereits gegeben ist und andere spontan hinzukommen. In den weiteren Betrachtungen soll von einer Gerätestruktur ausgegangen werden mit sowohl in der Umgebung bereits vorhandenen Geräten, aber auch solchen, die *ad-hoc* zum Geräteensemble hinzukommen. Diese Grundannahme ist in der Literatur nicht unüblich. So sprechen Heider et al. in ihrem Szenario von einem *Smart-Meeting-Room*, der auf einem *ad-hoc-Ensemble* basiert [HK05a]. Auch Aarts und Encarnação betrachten ein ad-hoc Meeting, bei dem sich einige Personen in einem durchschnittlichen Raum treffen und alle ihre Laptops und wenigstens einer einen Projektor mitbringt (vgl. [AE06] S. 327). Brumitt et al. betonen, dass mobile Geräte häufig ihre physikalische Position verändern, wenn sie sich mit den Benutzern bewegen. Diese Geräte müssen entsprechend zum Geräteensemble hinzugefügt oder aus diesem entfernt werden [BMK+00].
In der Regel sind in *Smart-Meeting-Rooms* mehrere Ausgabegeräte vorhanden, was sie zu *Multi-Display-Umgebungen* macht. Diese Displays werden im Allgemeinen von mehreren Benutzern verwendet, was sie zu *Multi-User-Umgebungen macht* (vgl. [BRH+08, WLSS09, HK05a, HNC+08]).
Eine weitere Eigenschaft von Smart Environments ist die *proaktive* also vorausschauende Unterstützung der Benutzer (vgl. [AHF+02, AE06, BCL+04, CD05]). Dies wird durch Interaktion und Beobachtung des Benutzers sowie durch Auswertung des aktuellen Kontextes erreicht [AE06, HW05, RMSF09, SPL06]. Da Smart Environments in der Regel proaktiv sind, müssen diese sich an den aktuell gegebenen Kontext und die Benutzersituation *adaptieren* bzw. anpassen [RMSF09, HNC+08]. So hebt beispielsweise Giersich hervor, dass bei Ressourcen, die sich über die Zeit verändern, eine Adaption an die unterschiedlichen Ressourcen erforderlich ist (vgl. [GFF+07]).

Die Arbeitsweise von Smart Environments wird folglich [HW05] [FFH00] in drei grundlegende Schritte unterteilt:

1. Die Umgebung bzw. deren Geräteensemble muss die aktuelle Situation des Benutzers, seine Interaktionen mit der Umgebung und ihren eigenen Zustand ermitteln.
   Brumitt et al. [BMK+00] führen für diese Aufgabe die *Inputgeräte* ein. Dey et al. beschreiben die aktuelle Situation über die durch die Geräte wahrgenommenen Informationen wie Identität, Position, Aktivitäten der Benutzer, der Benutzergruppen bzw. Objekte. In [JW03, OW05] wird diese Beschreibung erweitert und ein Kontext-sensitives Anwendungsmodell vorgestellt. Die aktuelle Situation bezogen auf den Benutzer wird durch sechs Fragen ermittelt: Was? (das Objekt), Wann? (der Zeitpunkt), Wo? (der Ort), Wer? (die Person), Warum? (die Benutzerintention) und Wie? (durch Gesten des Benutzers). Die angegebene Kontextbeschreibung fokussiert demnach auf die Benutzer.

2. Anschließend leitet die Umgebung daraus die Intension des Benutzers und dementsprechende mögliche Reaktionen ab, welche eine kooperative, proaktive Unterstützung der Benutzer erlauben. Es ist folglich ein *Adaptionsmechanismus* zu bestimmen, welcher den aktuellen Zustand der Smart Environment in den vom Benutzer gewünschten Zustand überführt. Für diese Aufgabe werden die *verarbeitenden Geräte* eingesetzt [BMK+00].

3. Im letzen Schritt wird der ermittelte Adaptionsmechanismus ausgeführt. Dadurch wird die aktuelle Situation an die Wünsche des Benutzers angepasst. Die letztendliche Manipulation der Umgebung vom aktuellen Zustand hin zum angestrebten Zustand fällt in der Regel den *Outputgeräten* zu [BMK+00].

2.1.2 Anwendungsszenario

Die Untersuchungen in dieser Arbeit beziehen sich auf *Smart-Meeting-Rooms*. Zum besseren Verständnis sollen Aufbau und Bestandteile eines *Smart-Meeting-Rooms* kurz vorgestellt werden. Die Bestandteile eines *Smart-Meeting-Rooms* lassen sich analog zur Schrittfolge aus [BMK+00] drei Geräteklassen zuordnen.

Inputgeräte Zu diesen zählen Kameras, Trackingsysteme, Wandschalter, aber auch mit Sensoren ausgestattete Bodenplatten [BMK+00], Interaktive Berührungsdisplays und Tablet-PCs für Interaktionen [WLSS09] sowie Mäuse und Zeigegeräte.

Verarbeitende Geräte Auf diesen Geräten können Berechnungen durchgeführt werden [BMK+00]: Dazu zählen beispielsweise Desktop Computer, Laptops, PDAs, Smart-Phones [TTS09].

Outputgeräte Hierzu gehören fest installierte Displays, Lautsprecher oder Beleuchtungselemente [BMK+00], aber auch mobile Ausgabegeräte wie Table-PCs [WLSS09] und darüber hinaus PDA's, Smart-Phones und mobile Beamer [TTS09].

Software Heider und Kirste stellen fest, dass Geräte, die zusammen ein Geräteensemble bilden, eine Software benötigen, welche die Gerätekooperation und die Unterstützung der Benutzer realisiert [HK05a]. Eine geeignete Software-Basis ist somit ebenfalls Bestandteil eines *Smart-Meeting-Rooms*.

2.2 Informationsvisualisierung

2.2.1 Erzeugung von visuellen Repräsentationen

Definition und Begriffsbestimmung

Ziel der Informationsvisualisierung ist es, Daten in eine visuelle Repräsentation zu überführen, um damit die visuelle Analyse der Daten sowie die Kommunikation der zugrunde liegenden Eigenschaften zu unterstützen. Um den Prozess der Informationsvisualisierung zu beschreiben, wurden in der Vergangenheit verschiedene Modelle entwickelt.
Ein leistungsfähiges Modell ist das Data-State-Reference-Model (DSRM) von Chi [Chi00, CR98]. Es ist flexibel, fein granular und wird in zahlreichen Visualisierungssystemen eingesetzt (vgl. Advizor [Eic00], Improvise [Wea04], Polaris [STH02, STH03], Prefuse [HCL05], and SAS/JMP [Che04], History Mechanism [KNS04]). Das DSRM soll daher auch in dieser Arbeit als Basis dienen. Es beschreibt die schrittweise Überführung der darzustellenden *Daten* in eine visuelle Repräsentation. Hierbei werden zunächst die gegebenen *Daten* durch das *Filtering* in die Datenstufe der *analytischen Abstraktionen* überführt. Das *Mapping* überführt wiederum die *analytischen Abstraktionen* in *visuelle Abstraktionen*. Diese werden durch das *Rendering* letztendlich in die *Bilddaten* überführt (siehe hierzu Abbildung 2.1).

Die vier Datenstufen des DSRM im Einzelnen:

Gegebene Daten sind die Basisdaten, die zum Erstellen der visuellen Repräsentation verwendet werden.

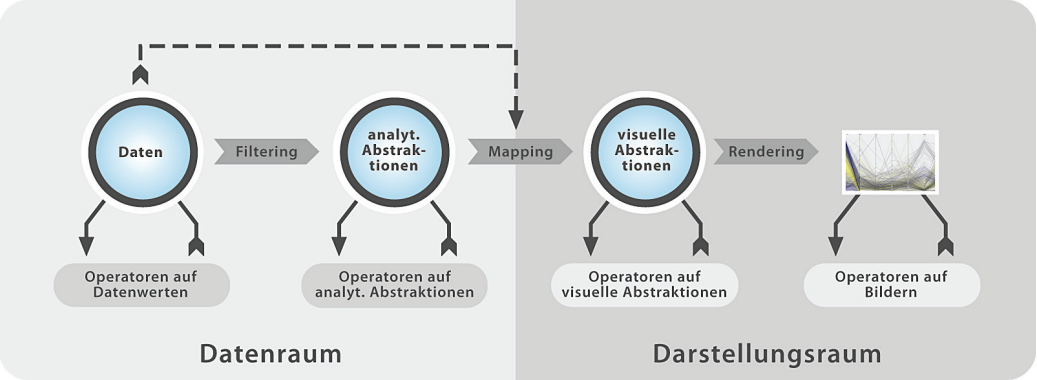

Abbildung 2.1: Das Data-State-Reference-Model in Anlehnung an [Chi00]

Analytische Abstraktionen werden durch das Filtering aus den gegebenen Daten gewonnen. Bei den Daten dieser Stufe handelt es sich um Metadaten, Informationen oder berechnete Eigenschaften zu den zu visualisierenden Daten. Es kann sich dabei beispielsweise um Clustereigenschaften oder statistische Kennwerte handeln.

Visuelle Abstraktionen werden durch das Mapping aus den beiden vorhergehenden Stufen erzeugt. Es handelt sich dabei um geometrische Daten. Allgemein betrachtet liegen die Daten in einer visuellen Abstraktion in einer definierten grafischen Beschreibungsform vor. Dabei werden sowohl die Geometrie wie Form, Richtung und Größe beschrieben, wie auch zusätzliche Attribute wie Farbe und Textur [Ber82].

Bilddaten werden durch den Renderingschritt aus den visuellen Abstraktionen berechnet. Bei den Bilddaten handelt es sich entweder um Pixel (bei einem 2D Ausgabegerät) oder Voxel (im Fall eines 3D Ausgabegerätes). Bilddaten sind also die visuelle Repräsentation der gegebenen Daten.

Die Stufen der Pipeline (DSRM) lassen sich dem *Datenraum* und dem *Darstellungsraum* zuordnen. Die Daten und analytischen Abstraktionen beschreiben abstrakte Daten und werden deshalb dem *Datenraum* zugeordnet. Hingegen werden die visuellen Abstraktionen und die Bilddaten, welche grafische Daten enthalten, dem *Darstellungsraum* zugeordnet. Um zu beschreiben, wie Daten verarbeitet und die Datenstufen ineinander überführt werden, verwendet das DSRM Operatoren.
Das Data-State-Reference-Model unterscheidet zwei Klassen von Operatoren:

Stufen-interne-Operatoren modifizieren die Daten innerhalb einer Stufe der Pipeline. Sie beeinflussen also ausschließlich die Daten in der zu-

gehörigen Stufe. Dies sind also Operatoren auf den gegebenen Daten, den Operatoren auf den analytischen Abstraktionen, den Operatoren auf den visuellen Abstraktionen und den Operatoren auf den Bilddaten (vgl. Abbildung 2.1).

Transformationsoperatoren überführen die Daten von einer in die nächste Stufe der Pipeline. Dies sind also die Operatoren Filtering, Mapping und Rendering (vgl. Abbildung 2.1).

Wichtig für die weiteren Betrachtungen ist, dass durch die Operatoren der gesamte Erzeugungsprozess der visuellen Repräsentation in kleine funktionale Einheiten unterteilt wird, die sich leicht entfernen, austauschen oder hinzufügen lassen [FTS07]. Der Bildaufbau lässt sich also gezielt durch den Einsatz bestimmter Operatoren steuern.

2.2.2 Einflussfaktoren zur Steuerung des Visualisierungsprozesses

Die aktuelle Situation bzw. der Kontext ist durch die vier "W" **W**as? (die Daten), **W**er? (der Benutzer), **W**ie? (die Aufgabe), **W**o? (das Ausgabegerät) bestimmt. Die Steuerung des Visualisierungsprozesses muss unter deren Berücksichtigung erfolgen (vgl. [TS07]). Die Ansätze in der Literatur fokussieren in der Regel ein oder zwei dieser Bereiche und vernachlässigen andere. Bevor existierende Ansätze untersucht werden, soll der Kontext weiter ausgeführt werden.

Beschreibung der Daten

Der Gegenstand einer Visualisierung wird durch die darzustellenden Daten vorgegeben. Zur Spezifikation der Eigenschaften dieser Daten eignen sich Metadaten. Prinzipiell können folgende Metadaten unterschieden werden (vgl. [Noc07, RH97]):

Beschreibende Metadaten Beschreiben grundlegende Eigenschaften sowie die Zugriffs- und Speicherstruktur der Daten. Dazu gehören z. B.:

- der Variablenname
- Wertebereichseigenschaften der Variablen und
- semantische Informationen zur Beschreibung der Variablen. Diese enthalten z. B. Informationen, ob es sich bei den Datenwerten um numerische Werte, Zeichenketten oder Referenzen handelt.

Abgeleitete Metadaten Sind Metadaten, die aus den Daten durch automatische Berechnung gewonnen werden. Welche Eigenschaften berechnet werden, ist im Allgemeinen von den speziellen Anwendungen abhängig. Am häufigsten werden statistische Verfahren eingesetzt, z. B. zur Berechnung von Extremwerten und Trends sowie Clustereigenschaften und Hauptkomponenten.

Historische Metadaten Beinhalten Informationen über die Erhebung der Daten, z. B. Ort, Zeit und Qualität der Daten.

Beschreibung der Aufgabe

Im Nutzerinterfacedesign (HCI[1]) werden die zu lösenden Aufgaben über komplexe Aufgabenmodelle beschrieben. Beim Visualisierungsdesign verwendet man dagegen einfachere Beschreibungen. Beide Ansätze sollen im Folgenden kurz vorgestellt werden.

Taskmodelle in der HCI *Aufgabenmodelle* wurden in der HCI-Community entwickelt, um den Workflow zu beschreiben, für den ein Software-System eingesetzt werden soll. Aufgabenmodelle werden als hierarchische Struktur gespeichert.
Die Knoten der Struktur repräsentieren die einzelnen Aufgaben. Die Kindknoten eines Vaterknotens beschreiben Teilaufgaben, die zur Lösung der Aufgabe des Vaterknotens durchgeführt werden müssen. Zwischen den Kindknoten können kausale und zeitliche Abhängigkeiten definiert werden. Diese Abhängigkeiten sowie die Strukturierung komplexer Aufgaben in eine Folge von Teilaufgaben werden über die Kanten der Hierarchie beschrieben. Ein Pfad in dieser Struktur beschreibt die Aufgabenabfolge zur Lösung eines konkreten Problems.
Die Aufgaben eines Aufgabenmodells können wie folgt unterteilt werden [WPF04]:

Nutzeraufgaben (user tasks) Diese werden hauptsächlich vom Benutzer selbst ausgeführt, z. B. das Lesen eines Flugplans und das Auswählen eines Fluges.

Abstrakte Aufgaben (abstract tasks) Dies sind Aufgaben, die komplexe Aktionen erfordern. Eine abstrakte Aufgabe besteht aus einer Reihe von Teilaufgaben.

Interaktionsaufgaben (interaction tasks) Hierunter versteht man Benutzerinteraktionen mit dem System.

[1] Human–Computer-Interaction

Anwendungsaufgaben (application tasks) Das sind Aufgaben, die das System ausführt. Sie können z. B. den Benutzer mit Informationen versorgen.

Weitere Informationen zum Aufbau und zur Spezifikation von Aufgabenmodellen können in [MPS02, PS02, Pat00, PMM97, BBP+00] nachgelesen werden.
Aufgabenmodelle werden für einen konkreten Anwendungshintergrund erstellt. In [FRSF06] wird für ein Wartungsszenario die Steuerung der visuellen Ausgabe anhand eines Aufgabenmodells beschrieben. Im Allgemeinen finden Aufgabenmodelle im Bereich der Visualisierung bisher aber keine Anwendung. Hier werden andere Beschreibungsformen genutzt.

Aufgabenbeschreibung in der Visualisierung In der Visualisierung werden Aufgaben im Allgemeinen durch eine Auflistung verbaler Ziele beschrieben. Dies ist ein sehr einfacher Ansatz, der es nicht erlaubt, komplexere Zielstellungen zu formulieren. Deswegen wird in [And06] ein formales Modell vorgestellt, welches es ermöglicht, je nach Betrachtungsweise Ziele durch *elementare Ziele* oder *zusammengesetzte Ziele* zu beschreiben.
Dabei werden zwei Klassen von Visualisierungszielen unterschieden. *Elementare Ziele* (elementary tasks) und *zusammengesetzte Ziele* (synoptic tasks). Die Klassen werden anhand der Daten unterschieden, für die sie gelten.
Elementare Ziele arbeiten nur mit einzelnen Elementen der Daten. Bei den Elementen handelt es sich um konkrete Beobachtungspunkte[2] oder Ausprägungen[3]. Alle Ziele, die Teilmengen der Elemente gleichzeitig betrachten, werden der Klasse der zusammengesetzten Ziele zugeordnet.
Zu den *elementaren Visualisierungszielen* gehören:

Suchziele (Betrachtung) Dabei werden die Daten gefiltert und anschließend dargestellt.

- **direkte Suchziele** (Identifikation). Dabei ist ein Beobachtungspunkt gegeben und es ist eine Ausprägung gesucht.
- **inverse Suchziele** (Lokalisation). Bei diesen ist eine konkrete Ausprägung gegeben und die zugehörigen Beobachtungspunkte sind gesucht.

[2]Die zu visualisierenden Daten seien an Beobachtungspunkten eines Beobachtungsraumes gegeben.

[3]Die im Beobachtungsraum erhobenen Daten (Merkmale) werden als *abhängige Variable* bezeichnet. Ein konkreter Wert einer abhängigen Variablen wird als *Ausprägung* bezeichnet.

Relationale Ziele (Vergleiche) behandeln Beziehungen zwischen Elementen oder Teilmengen von Elementen. Relationale Ziele lassen sich in *Vergleichsziele* und in *Beziehung suchende Ziele* unterteilen.

- **Vergleichsziele** Gegeben sind Elemente oder Teilmengen. Gesucht ist die Beziehung zwischen den Elementen bzw. Teilmengen. Vergleichsziele beinhalten eine Suchfunktion. Je nach Art der Suchfunktion können die Vergleichsziele in *direkte* und *inverse Vergleichsziele* unterteilt werden.
- **Beziehung suchende Ziele** Gegeben sind konkrete Beziehungen. Gesucht sind Ausprägungen oder Beobachtungspunkte, die in den gegebenen Beziehungen zueinander stehen.

Die *zusammengesetzten Ziele* beschreiben das Verhalten und suchen nach Mustern. Verhalten drückt die Relationen zwischen Merkmalsausprägungen in verschiedenen Bereichen des Beobachtungsraumes aus. Ein Muster beschreibt charakteristische Eigenschaften eines Verhaltens, z. B. in Form konkreter Werte. Zusammengesetzte Ziele lassen sich in *Suche nach Zusammenhängen* und *Beschreibende Ziele* unterteilen.

Suche nach Zusammenhängen hat zum Ziel, in den Daten Zusammenhänge aufzudecken.

Beschreibende Ziele:

- **Betrachtung von Mustern:**
  - **Identifikation** Hierbei ist eine Menge von Beobachtungspunkten gegeben. Gesucht ist ein Muster, das das Verhalten der Beobachtungspunkte beschreibt.
  - **Lokalisation** Hierbei ist ein Muster gegeben. Gesucht ist eine Menge von Beobachtungspunkten, deren Ausprägungen durch das Muster möglichst gut angenähert werden.
- **Vergleich von Mustern** Es wird von zwei Mustern untersucht, wie sie bezüglich Gemeinsamkeiten und Unterschieden miteinander in Beziehung stehen.
- **Suche nach Ähnlichkeiten** Suchen nach bestimmten Beziehungen zwischen Merkmalsausprägungen und Bestimmung der zugehörige Menge von Beobachtungspunkten.

In Abbildung 2.2 werden die einzelnen Ziele in einer Grafik dargestellt.

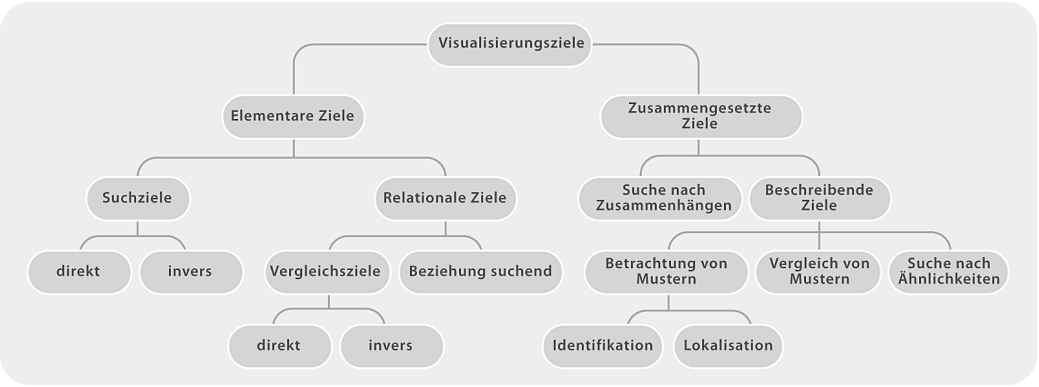

Abbildung 2.2: Schema der Visualisierungsziele nach [And06]

Die in [And06] eingeführten Ziele lassen sich beliebig kombinieren und damit gut einsetzen, um eine Adaptation der visuellen Repräsentationen zu steuern. Zum Beispiel wurden in [Sch06] diese Visualisierungsziele eingesetzt, um geeignete Farbskalen auszuwählen und anzupassen. Dazu wird für die einzelnen Ziele spezifiziert, welche Eigenschaften die Farbskalen haben müssen, um dieses Ziel ausdrücken zu können. Im Detail werden folgende Aufgaben unterschieden:

- Betrachtung: Werte sollen möglichst genau abgelesen werden.
- Vergleich: Es wird eine globale Farbskala verwendet, um einen Vergleich von Werten zu ermöglichen.
- Identifikation: Die Ausprägung eines Merkmals soll möglichst genau abgelesen werden.
- Lokalisierung: Die Abstände von Ausprägungen eines Merkmals in Relation zu einem gegebenen Wert sollen verdeutlicht werden.
- kontinuierlich: Die Abstände der Ausprägung eines Merkmals zu einem gegebenen Wert werden dargestellt.
- segmentiert: Die Ausprägung eines Merkmals soll hervorgehoben werden.

Abbildung 2.3 veranschaulicht dieses Vorgehen.
Das Beispiel macht deutlich, welchen Einfluss unterschiedliche Farbskalen auf die Ausgestaltung der visuellen Repräsentation und damit, welchen Einfluss die Ziele haben.

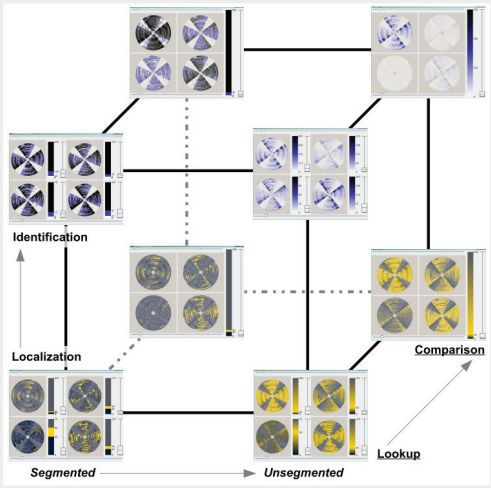

Abbildung 2.3: Adaption der Farbskala einer visuellen Repräsentation anhand von Visualisierungszielen aus [TFS08b]

Beschreibung der Ressourcen

Während es für die Spezifikation von Daten und Zielen Beschreibungen gibt, die auf den Visualisierungsprozess zugeschnitten sind, gibt es für die Eigenschaften der Ressourcen derzeit keine allgemein gültige und allgemein anerkannte Beschreibung, die sich zur Steuerung des Visualisierungsprozesses einsetzen lässt. Im Kontext von *Smart-Meeting-Rooms* spielen aber für eine Visualisierungsanwendung gerade die Eigenschaften des *Ausgabegerätes*, die *Rechenleistung* und der *Kommunikationskanal* eine wichtige Rolle. Erst durch das Berücksichtigen dieser Faktoren ist es möglich, die visuellen Repräsentationen an die gegebenen Randbedingungen anzupassen.

Eigenschaften des Ausgabegerätes In *Smart Metting Rooms* und speziell in Multi-Display-Umgebungen können grafische Ausgaben auf den unterschiedlichsten Ausgabegeräten erfolgen. Diese Geräte unterscheiden sich in Pixelauflösung und Farbfähigkeit, was für die Visualisierung von größter Bedeutung ist.
Für die Ausgabegeräte sollen deshalb folgende Eigenschaften spezifiziert werden:

- Die **Auflösung der Anzeige** in X- und Y-Richtung. Die Auflösung gehört zu den wichtigsten Kenngrößen. Sie beschreibt, wie viele Bildpunkte auf einem Display dargestellt werden können.

- Die **Abmessung der Anzeige** in X- und Y-Richtung. Die Größe muss ebenfalls betrachtet werden, da es ein Unterschied ist, ob auf einem 19" Monitor 1600 x 1280 Bildpunkte dargestellt werden oder auf einem 3" Monitor.

- Die **Farbfähigkeit** bestimmt, welche Farbtiefe das Display hat und ob es generell in der Lage ist, Farben darzustellen. Eine grobe Unterteilung der Farbfähigkeit eines Displays ist sinnvoll, um zu bestimmen, inwieweit Farbe als eine visuelle Variable [Ber82] verwendet werden kann. Die Farbfähigkeit soll wie folgt eingeteilt werden:

  - **Keine:** Es können keine Farben dargestellt werden. Das trifft z. B. für einige Verfahren des E-paper zu.
  - **Normal:** Es können alle im RGB-System definierten Farben wiedergegeben werden. Dieser Stufe werden gebräuchliche Ausgabegeräte wie Desktop PCs, aber auch Smart-Phones oder PDAs zugeordnet.

- Für die **Anzeigequalität** soll die Eigenschaft der Leuchtdichte verwendet werden. Die Leuchtdichte spielt z. B. dann eine Rolle, wenn Umgebungslicht die Ausgabe beeinflusst. Streulichteffekte mindern die Anzeigequalität. Weiterhin soll das Umgebungslicht in etwa der Leuchtdichte des Ausgabegerätes entsprechen [EZ98]. Hierbei lassen sich folgende Stufen unterscheiden:

  - **Gering:** Die Bildausgabe des Gerätes ist bereits bei stärkerem diffusen Lichteinfall nicht mehr deutlich zu erkennen. Hierzu zählen z. B. Laptops, die die Helligkeit ihrer Anzeige stark reduzieren, wenn sie im Batteriemodus betrieben werden oder Beamer mit nur geringer Leuchtkraft. Die Leuchtdichte von Geräten dieser Stufe liegt unter 100 cd/m^2.
  - **Normal:** Die Anzeigequalität wird als Normal eingestuft, wenn bei leichtem direkten Licht oder starkem diffusen Licht die Ausgabe noch gut zu erkennen ist. Diese Stufe entspricht einer normalen Anzeige wie z. B. einem Röhrenmonitor oder hochwertigen Flachbildschirmen. Die Leuchtdichte der Geräte sollte ca. 250 cd/m^2 betragen.

Die Eigenschaften der Ausgabegeräte können auf verschiedene Weise bei der Adaption einer Visualisierungstechnik eingesetzt werden. Die Anzeigequalität und Farbfähigkeit beeinflussen den Mappingschritt, d. h. die Abbildung von Informationen auf visuelle Attribute. Die Auflösung und Abmessungen

parametrisieren den Renderschritt, d. h. die Darstellung der visuellen Attribute. Weiterhin können die Parameter auch verwendet werden, um ein *Visual Clutter* vorherzusagen und gegebenenfalls ein *Information-Hiding* anzustoßen. Darüber hinaus spielt die Anzeigequalität bei der Wahl der Kontrasteinstellungen eine Rolle.

Die Berechnungskapazität Jede Visualisierungstechnik benötigt eine Ressource, die für die Ausführung des Visualisierungsprozesses verantwortlich ist. Die Berechnungskapazität kann von Gerät zu Gerät variieren. Gleichfalls variiert auch die Rechenkapazität, die von verschiedenen Visualisierungstechniken benötigt wird. An dieser Stelle soll eine Einteilung der Rechenleistung vorgestellt werden, die die Leistung der Geräte im Allgemeinen widerspiegelt und zunächst keine parallele Berechnung berücksichtigt. Damit wird ausgedrückt, ob das Kriterium der Angemessenheit einer visuellen Repräsentation (vgl. [SM00]) überhaupt erfüllbar ist, d. h. ob Kosten und Nutzen dem Ziel der Visualisierung entsprechen. Die Einteilung entspricht den heute gängigen Geräteklassen.

1. **Stufe** Geräte dieser Stufe entsprechen in etwa der Leistung eines heutigen WAP-Handys.

2. **Stufe** Diese Stufe wird Geräten mit der Leistung eines Smartphones oder eines PDAs zugeordnet.

3. **Stufe** Rechenressourcen, die über die Leistung von heutigen Laptops oder Desktop PCs verfügen, sollen dieser Stufe zugeordnet werden.

4. **Stufe** Diese Stufe wird Geräten oder Systemen mit einer hohen Leistungsfähigkeit zugewiesen. In diese Kategorie fallen z. B. Mehrprozessor-Systeme oder Rechencluster.

Steht nicht genug Rechenleistung zur Verfügung, müssen einfache Visualisierungstechniken zum Einsatz kommen oder die Datenmenge, die visualisiert werden soll, ist stärker einzuschränken. Somit hat auch die Rechenkapazität einen direkten Einfluss auf die Ausgestaltung einer Visualisierung.

Der Kommunikationskanal In *Smart-Meeting-Rooms* wird von vernetzten Ressourcen ausgegangen. Der Kommunikationskanal zwischen einzelnen Ressourcen kann dabei unterschiedlich leistungsfähig sein. Daraus folgt, dass für jede Ressource die Parameter des Kanals angegeben werden müssen. Dabei ist zu beachten, dass ein Kanal nicht exklusiv einer Anwendung zur Verfügung stehen muss.

Nachfolgend sollen die Parameter aufgeführt werden, die für die Bewertung des Kommunikationskanals für Visualisierungszwecke von Bedeutung sind. Dabei sollen die Parameter in zwei Gruppen eingeteilt werden.
Die *primären Parameter* müssen für jeden Kommunikationskanal angegeben werden. Die *sekundären Parameter* spielen nur in speziellen Situationen eine Rolle und sollen deswegen optional sein. Nach Mao [MBNP03] spielen bei der Kommunikation im multimedialen Umfeld Parameter wie Bandbreite, Latenz, durchschnittliche Verlustrate[4] und Verlustmuster[5] eine Rolle. Während im Umfeld der Informationsvisualisierung die Parameter Verlustrate und Verlustmuster nur eine untergeordnete Rolle spielen, sind die Parameter Bandbreite und Latenz von vorrangigem Interesse (vgl. [TSR09]). Diese Einflussfaktoren sollen für jeden Kommunikationskanal geeignet erfasst werden.

Zu der Gruppe der *primären Parameter* zählen damit also:

Bandbreite Die Bandbreite eines Kommunikationskanals gibt dessen Transferleistung in KB/sec an. Die Bandbreite soll im Allgemeinen als durchschnittliche Bandbreite aufgefasst werden.

minimale Bandbreite Die minimale Bandbreite kann im Rahmen einer „Quality of Service-Anforderung“ eine Rolle spielen. Die minimale Bandbreite wird genauso wie die Bandbreite in KB/s angegeben.

Latenz Die Latenz eines Kommunikationskanals ist von Bedeutung, wenn die Antwortzeit eine Rolle spielt. Wird ein Inputsignal, wie z. B. die Mausbewegung, auf einen anderen Rechner übertragen, führt eine hohe Latenz zu starken Qualitätseinbußen bei der Interaktion.

Zu der Gruppe der *sekundären Parameter* gehören:

Datensicherheit Diese Eigenschaft soll angeben, ob ein Kommunikationskanal über Protokolle verfügt, die eine Sicherung des Datenstroms vornehmen, z. B. Paketverlust oder Paketreihenfolge usw.

Verschlüsselung Handelt es sich bei den Daten um sicherheitssensible Daten, so ist eine Verschlüsselung dieser Daten sinnvoll. Sensible Daten dürfen nicht über Kanäle versendet werden, die von Dritten mitgelesen werden können und die nicht verschlüsselt sind.

[4] Verlust von Netzwerkpaketen
[5] wiederkehrende Paketverluste im Bezug zur Trägerfrequenz oder im zeitlichen Verlauf

Die genannten Einflussfaktoren sind für die Adaption einer Visualisierungstechnik von Bedeutung. Es ist möglich, eine visuelle Repräsentation in mehreren Schritten zu berechnen.
Die einzelnen Schritte müssen nicht zwangsläufig auf derselben Ressource berechnet werden. Wird nun eine Visualisierungstechnik aufgeteilt, müssen die Ergebnisse der Berechnungen eines Schrittes auf die Ressource transferiert werden, die die Daten als Nächstes benötigt. Auch wenn man nicht von einer Verteilung der Berechnung ausgeht, so können trotzdem die zu visualisierenden Daten auf einer anderen Ressource liegen, als für die Visualisierungsberechnung zur Verfügung steht. Ebenso können die Berechnung und Ausgabe der visuellen Repräsentation durchaus auf unterschiedlichen Ressourcen erfolgen.
Es ist also in der Regel ein Datentransfer zwischen den unterschiedlichen Ressourcen notwendig. Die genannten Einflussparameter aus dem Bereich Kommunikation bestimmen nun, ob es möglich ist und gegebenenfalls wie es möglich ist, eine Visualisierung auf unterschiedliche Ressourcen zu verteilen. Im Fall von sehr großen gegebenen Datenmengen und nur geringen Bandbreiten ist es sinnvoll, die Berechnung auf derselben Ressource durchzuführen, auf der sich die Daten befinden und nur das Ergebnisbild zu übertragen. Da Visualisierungstechniken per Definition interaktiv sind, spielt auch die Latenz eine wichtige Rolle. Es muss also vermieden werden, dass eine zu hohe Latenz interaktive Antwortzeiten verhindert.

Beschreibung des Benutzers

Auch die spezifischen Eigenschaften eines Nutzers müssen bei der Visualisierung berücksichtigt werden. Diese lassen sich in objektive und subjektive Eigenschaften unterteilen. Lange [LNS06] führt unter Berücksichtigung dieser Einteilung die folgenden Eigenschaften auf:
Zu den subjektiven Eigenschaften werden z. B. die perzeptiven und kognitiven Fähigkeiten, Erfahrungen, Präferenzen und Eigenheiten des Nutzers gerechnet. Zu den objektiven Eigenschaften werden z. B. die grundlegenden Eigenschaften der menschlichen visuellen Wahrnehmung gezählt.
Außerdem müssen, um einen Benutzer identifizieren zu können, weitere Eigenschaften, wie z. B. Vorname, Name und Anschrift erfasst werden. In einigen Szenarien, insbesondere in *Smart-Meeting-Rooms*, ist zudem auch die Position des Nutzers interessant[6]. Siehe auch [KEM07].

[6]Dieser Aspekt wird verstärkt in der zweiten Phase des GRK-MuSAMA untersucht.

Ein allgemeines Schema zur Beschreibung der Einflussfaktoren

Eine Visualisierungsanwendung wird durch eine Vielzahl von unterschiedlichen Faktoren beeinflusst. Es muss möglich sein, diese Einflussfaktoren systematisch zu verarbeiten, um adäquate Informationsdarstellungen zu erzeugen. Daraus folgt, dass hierfür geeignete Beschreibungen zur Verfügung stehen müssen. In bisherigen Anwendungen wurde hauptsächlich das *Was* und das *Warum* betrachtet. In heterogenen Multi-Display-Umgebungen ist insbesondere aber noch das *Wo* zu berücksichtigen. Um die Eigenschaften und Intentionen eines Benutzers berücksichtigen zu können, ist zusätzlich auch noch das „*für Wen*“ zu erfassen. Die Beschreibung des Kontextes, der für die Informationsvisualisierung relevant ist, ist somit konform zu [JW03].
Auf der Basis dieser vier Fragestellungen lassen sich die Einflussfaktoren in die Bereiche *Daten*, *Aufgaben*, *Geräte* und *Nutzer* einteilen und entsprechend spezifizieren.
Dazu wurden in den vorhergehenden Abschnitten mögliche Beschreibungen vorgestellt (vgl. Abschnitt 2.2.2). Daraus ergibt sich das in Abbildung 2.4 aufgeführte Schema, das je nach Bedarf beliebig verfeinert werden kann (vgl. [TS07]).

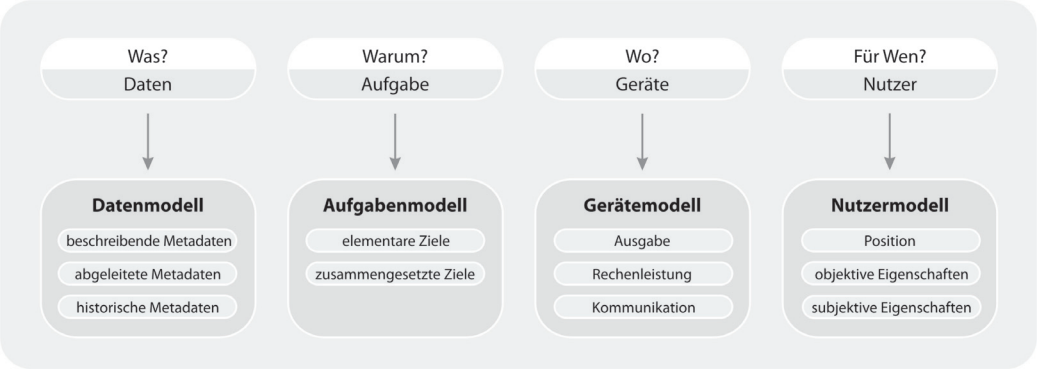

Abbildung 2.4: Schematische Darstellung der Einflussfaktoren zur Adaption visueller Repräsentationen aus Thiede et al. [TS07].

2.2.3 Adaption von visuellen Repräsentationen

Nachdem die Rahmenbedingungen vorgestellt wurden, die bei der Erzeugung visueller Repräsentationen zu berücksichtigen sind, soll nun auf Ansätze in der Literatur eingegangen werden, die bei der Adaption der visuellen Repräsentation einen oder mehrere dieser Faktoren nutzen.

Berücksichtigung der Daten

Die Eigenschaften der Daten als Gegenstand der Visualisierung werden von jeher als wichtigster Faktor im Visualisierungsdesign berücksichtigt.
Bereits 1986 wurden dazu erste Ansätze von Mackinley [Mac86] veröffentlicht. Der Autor geht auf ein Konzept für ein Präsentationstool ein, welches automatisch effektive grafische Repräsentationen für eine große Bandbreite von unterschiedlichen Klassen von Daten erstellt.
Das Konzept sieht vor, grafische Repräsentation durch grafische Sprachen zu beschreiben und auf dieser Basis Kriterien für eine Algebra aufzustellen, welche die Komposition von effektiven und expressiven Repräsentationen erlaubt. Auf dieser Basis kann ein Programm automatisch visuelle Repräsentationen erstellen.
Der Autor räumt ein, dass auch mit dem von ihm vorgestellten Tool, noch nicht alle Probleme im Zusammenhang mit der automatischen Generierung von grafischen Repräsentationen gelöst sind. [Mac86]

Carmo et al. [CCC02] stellen 2002 ein generisches Visualisierungssystem vor, welches automatisch die visuelle Repräsentation zu gegebenen abstrakten Daten erstellt. Das Interface wird dabei automatisch an die Daten angepasst. Ihr Ansatz setzt unterschiedliche Filtermethoden ein:

- Ausblendung von ein oder mehr Informationsgruppen
- Anpassung der angezeigten Attribute
- Bereiche von Interesse, um die Menge der angezeigten Informationen zu reduzieren und um eine geeignete Repräsentation auszuwählen

Das von den Autoren vorgestellte System ist auf Daten, die tabellarisch organisiert sind, beschränkt. Bei der Adaption an die Daten wird in erster Linie das Benutzerinterface entsprechend angepasst.

Auch aktuelle Arbeiten, wie die von Gilson et al. von 2008 [GSGC08] befassen sich mit dem automatischen Visualisierungsdesign auf der Basis der gegeben Daten.
Ihr Ansatz hat zum Ziel, für Domain spezifischen Daten automatisch eine Visualisierung zu generieren. Dazu verwenden die Autoren eine generelle Pipeline, welche ein Ontologie Mapping und wahrscheinlichkeitstheoretisch Ansätze vereinigt.
Dabei wird beim automatischen Generieren der Visualisierung in einem ersten Schritt eine Webseite auf eine *Domain Ontologie* (DO) abgebildet. In einem zweiten Schritt wird die DO auf eine oder mehrere *Ontologien zur*

visuellen Repräsentation (VRO) abgebildet. Unter Verwendung der *Semantik Bridging Ontologie* (SBO), welche das Mapping von Datenwerte auf die visuellen Artefakte enthält, wird abschließend die VRO auf eine visuelle Repräsentation abgebildet und so die visuelle Repräsentation erzeugt.
Siehe auch Abbildung 2.5.

Neben den genannten finden sich zahlreiche weitere Ansätze in der Literatur,

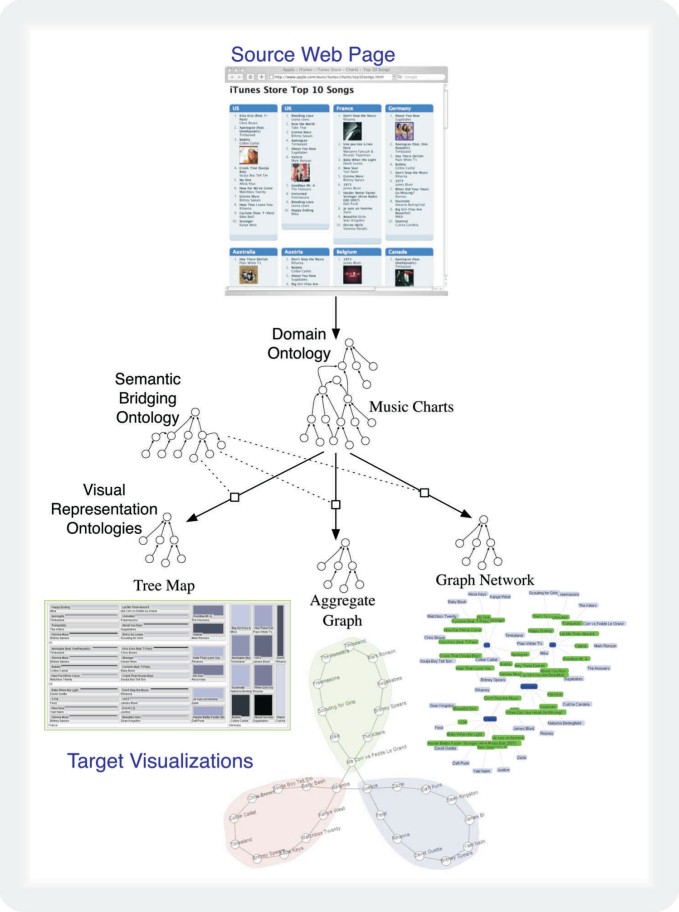

Abbildung 2.5: Visualisierungsdesign auf der Basis von Ontologien, aus [GSGC08].

die sich mit dem Visualisierungsdesign unter Berücksichtigung der Daten befassen (vgl. [RM90, Rob91, GRKM94, SI94, RLS+96, ZF96, ZCF02, NS02, TSB04, Noc07, GSGC08]).

Obwohl die Erzeugung visueller Repräsentation unter Berücksichtigung der Daten bereits so lange thematisiert wurden, gibt es bis heute keinen Ansatz, der für beliebige Daten, wie beispielsweise Vektordaten oder Strömungsda-

ten, automatisch und *On-the-Fly* Techniken auswählen und parametrisieren.

Berücksichtigung der Aufgabe

In diesem Abschnitt soll auf die Ansätze in der Literatur eingegangen werden, welche die visuelle Repräsentation an die Ziele der Benutzer anpassen.

In der HCI ist das Ziel der Benutzer bereits seit längerem ein Forschungsthema. In erster Linie werden dabei die Ziele der Benutzer ausgewertet, um auf dieser Basis die Modellierung der *WIMP*-Schnittstelle (*W*indow, *I*con, *M*enu, *P*ointing device) und geeigneten Interaktionen auf dieser anzupassen [Fuc10]. In der Regel werden dazu geeignete Aufgabenmodelle eingesetzt, um die Ziele der Benutzer zu beschreiben.
Hingegen existieren auf diesem Gebiet für die Visualisierung nur wenige Ansätze. Die vorhandenen Ansätze arbeiten dabei im Allgemeinen mit verbalen Auflistungen von Zielen.

Es gibt einige Arbeiten im Bereich der Visualisierung, welche sich mit der Auswahl von „guten“ Visualisierungen für eine gegebene Aufgabe befassen, beginnend mit den ersten Arbeiten dazu von Casner [Cas91] über [FTIN97, FFIT00, AA02] bis hin zu neueren Ansätzen im Bereich Aufgaben getriebene Visualisierung, wie [And06, MSK+06, SSK07, TFS08b].
So stellen beispielsweise Fredj und Duce [FD06] eine durch Semantik gesteuerte Erstellung von intelligenten Organisationsdiagrammen vor.
Mikovec [Mík07] beschreibt, wie kontextsensitive Methoden für die Präsentation von komplexen räumlichen Informationen verwendet werden können. Sein Ansatz basiert auf einer vereinheitlichten, angereicherten Datenbeschreibung, welche durch Prioritätswerte ergänzt wird, um die Präsentation an die Aufgabe anzupassen.
Eine weitere interessante Arbeit stellt Winckler [WPF04] vor. Er verwendet den Aufgabenmodellierungsformalismus CTT [Pat00, MPS02] nicht zum Steuern des Erzeugungsprozesses, sondern zum strukturellen Testen und Evaluieren von interaktiven Visualisierungstechniken. Dabei werden 11 grundlegende Aufgaben berücksichtigt [WL90], wie beispielsweise Identifizieren, Lokalisieren, Differenzieren oder Gruppieren.
In [FS09] wird ein Ansatz behandelt, der zum Ziel hat, die visuelle Ausgabe von gegebenen Modelldaten an die gegebenen Aufgaben anzupassen.
Obwohl Ansätze in der Literatur existieren, welche die Visualisierung an die Ziele der Benutzer anpassen, verwenden aktuell nur sehr wenige eine systematische Beschreibungen der Aufgaben wie beispielsweise die CTT und nutzen diese, um die Visualisierung umfassend an die Ziele der Benutzer anzupas-

sen.
Wie im Abschnitt 2.2.2 gezeigt wurde, handelt es sich bei diesem Bereich um ein eigenständiges Thema. Aktuell wird dieser Themenkomplex in der Dissertation von Georg Fuchs [Fuc10] bearbeitet.

Berücksichtigung des Benutzers

Die Benutzer in der Visualisierung sind zunehmend Gegenstand der aktuellen Forschung [TM04].
Besonders das Forschungsgebiet der *visual Analytics* hat zum Ziel, die Visualisierung, den Benutzer und die Datenanalyse stärker miteinander zu verbinden.
Auch andere Abhandlungen halten fest, dass zukünftige Forschung im Bereich der Visualisierung den Benutzer stärker in den Mittelpunkt rücken müssen [KEM07], da aktuell nur wenige Ansätze dazu in der Literatur vorhanden sind.
Als ein Vertreter stellen Schädlich und Mukasa [SM04] einen benutzerorientierten Ansatz zur Anpassung einer Informationsvisualisierung für den Bereich der Produktionsautomatisierung vor. Der Ansatz berücksichtigt dabei die Benutzeraufgaben, den Benutzerkontext und die Benutzerprofile. Die verwendeten Benutzerprofile geben dabei Auskunft über die Erfahrung und mögliche Einschränkungen des Benutzers.
Nach Aussage der Autoren beeinflusst die Erfahrung des Benutzers direkt, wie komplex eine Visualisierung sein darf. Sind Einschränkungen der Benutzer wie eine Farbenblindheit bekannt, können diese ebenfalls berücksichtigt werden.
Als Lösungsansatz verwenden die Autoren einen agentenbasierten Ansatz, bei dem die einzelnen Aufgaben der Benutzer in Teilaufgaben zerlegt werden und bei der Bearbeitung durch die Agenten jeweils das Benutzerprofil berücksichtigt wird.

Auf dem Gebiet der Benutzer-orientierten Anpassungen von Visualisierung existieren nur wenige Ansätze. Aus diesem Grund sehen Forschungsgebiete wie die *visual Analytics* in diesem Bereich einen Handlungsbedarf.

Berücksichtigung der Geräte

Die Verwendung von mehreren unterschiedlichen Geräten in der Informationsvisualisierung stand bisher nicht im Fokus (vgl. [SKKM+05]), anders in anderen Bereichen der Visualisierung wie beispielsweise der Volumenvisualisierung oder auch der Strömungsvisualisierung. Hier wurden aufgrund des

hohen Ressourcenbedarfs bereits sehr früh mehrere Rechner verwendet, um eine visuelle Repräsentation zu erstellen.
Einen Ansatz aus dem Bereich der Volumenvisualisierung stellen Bavoil et al. [BCC+05] vor. Das Besondere hierbei ist, dass die Autoren Beschreibung und Ausführung der Pipeline trennen. Das bietet den Vorteil, dass die Pipelinebeschreibung wieder verwendet werden kann. Zudem bietet der Ansatz ein großes Potential bei der Optimierung, da die Beschreibung der Pipeline optimiert werden kann, bevor diese ausgeführt wird.
Der Ansatz von Brodlie et al. [BDG+04b] hat zum Ziel, eine Visualisierung wie eine Strömungsvisualisierung auf mehrere Rechner in einem Grid zu verteilen.
Auch der Ansatz von Grimstead et al. [GAW04], welcher den Erzeugungsprozess von visuellen Repräsentationen von 3D-Modellen auf mehre heterogene Ressourcen in einem Netzwerk verteilt.
Ein Ansatz im Bereich der gerätebasierten Adaption findet sich bei [KJDG+06]. Dabei werden die Displaygröße und der zur Verfügung stehenden Kommunikationskanal bei der Adaption berücksichtigt (vgl. [KJDG+06]). Allerdings fokussieren auch hier die Autoren auf die Darstellung von 3D-Modellen.

Auch im Bereich der gerätebasierten Adaption von Videos und Bildern existieren Ansätze. Beispielsweise stellen Lu et al.[LED06] ein Framework vor, welches es erlaubt, Bilder und Videos an heterogene Ausgabegeräte anzupassen. Dazu werden die Medien mit zusätzlichen Metadaten versehen. Bei der anschließenden Ausgabe werden diese Metadaten verwendet, um eine adaptierte Variante zu erstellen.

In jüngerer Vergangenheit rückten zudem mobile Ausgabegeräte mehr und mehr ins Blickfeld für die Ausgabe von visuellen Repräsentationen. Daher finden sich auch in diesem Bereich einige Ansätze, die sich mit den Anpassungen von graphischen Inhalten an die unterschiedlichen mobilen Ausgabegeräte befassen (vgl. [EWE09]).

Auch der Trend Visualisierungssysteme in das World Wide Web zu verlagern und so einer breiteren Masse den Zugang dazu zu erleichtern [PBB+08], erfordert eine Anpassung der visuellen Repräsentation an unterschiedliche Ausgabegeräte. So stellt [ZHHM07] eine Visualisierung über eine Reihe von Web-Services bereit.
Andere Autoren verwenden direkt den Webbrowser, um GIS zu realisieren, indem Kartendarstellungen beispielsweise von Google geladen werden und anschließend mit weiteren Informationen, die im SVG Format vorliegen, anzureichern [EEF+07].

Auch gibt es Trends, über das Internet einzelnen Informationsvisualisierungen einer breiten Masse von Benutzern zugänglich zu machen. Ein Vertreter aus diesem Bereich ist *Many Eyes* [VWvH+07].
Es existieren außerdem Ansätze, welche im Internet eingesetzt werden. Dabei wird die Visualisierung an ein leistungsschwaches Ausgabegerät angepasst, indem Multi-Resolution-Modelle verwendet werden [FBBB03] oder auch indem vom Ausgabegerät eine Visualisierung von einem Server anfordert und diese dann als VRML-Modell durch den Server bereitgestellt wird [WBW96]. Weitere Ansätze für verteilte kooperative Visualisierung finden sich in Brodlie et al. [BDG+04a].

Geht man einen Schritt weiter und sucht in anderen Bereichen, so finden sich die meisten Ansätze zur automatischen Adaption an die Ausgabegeräte unter Berücksichtigung der Displaygröße auf dem Gebiet der HCI (vgl. [MPS04, KF02, CCT01, EVP01, MVL06]). Aber auch im Bereichen der Dokumentenanpassung sind Ansätze zu verzeichnen (vgl. [AHFS08, CG06]).
Zudem gibt es eine Reihe von Systemen, die allgemein zum Ziel haben, die grafischen Daten an den aktuellen Kontext anzupassen. Aber auch hier werden einige Rahmenbedingungen stärker berücksichtigt als andere(vgl. [JS05, CDM+00]).

Hingegen sind konkrete Ansätze im Bereich der Informationsvisualisierung, die eine visuellen Repräsentation an unterschiedliche Ausgabegeräte anpassen, in der Literatur stark unterrepräsentiert (vgl. [SKKM+05, ARL+06]).
Ein System in diesem Bereich stellen Skorin-Kapov et al. [SKKM+05] vor. Dieses passt gegebene Visualisierungstechniken auf unterschiedliche Ausgabegeräte an. Für die Adaption an ein spezifischen Ausgabegerät werden so genannte Plattformtreiber verwendet. Diese sind für die angepasste Ausgabe und Interaktion auf dem jeweiligen Gerät zuständig.

Zusammenfassung

Werden die Rahmenbedingungen 2.2.2 zugrundegelegt, so gibt es in allen Bereichen Arbeiten in der Literatur. Besonders viele Ansätze lassen sich zu der datenbasierten Adaption finden, hingegen finden sich nur wenige Ansätze in der Literatur, die aufgaben-, benutzer- und auch die gerätebasierte Adaption von Informationsvisualisierung behandeln.
Werden allerdings multiple, dynamische, heterogene Ausgabegeräte zugrundegelegt, die eine *On-the-Fly* Adaption der visuellen Repräsentation erforderlich machen, betrachten die vorgestellten Ansätze zwar immer einige dieser

Eigenschaften wie sie in *Smart-Meeting-Room* auftreten, doch es gibt in der Literatur keine Ansätze, die systematisch auf alle diese Eigenschaften eingehen.

2.2.4 Visual Clutter

Eine Adaption der visuellen Repräsentation an unterschiedliche Ausgabegeräte ist notwendig, weil *Visual Clutter* vermieden werden muss. Es gibt verschiedenen Formen von *Visual Clutter*:

- **Räumliches Clutter** *Visual Clutter* bezeichnet den Zustand, wenn zu viele Informationen auf zu wenig Platz dargestellt werden, so dass es zu Verdeckungen und Informationsverlusten kommt. Es tritt auch *Visual Clutter* mit räumlicher Verdeckung auf, wenn zu wenig Informationen auf zu viel Platz dargestellt werden und dabei der Zusammenhang zwischen den Informationen verloren geht (vgl.[FTSS09]).

- **Farbliches Clutter** Hierbei tritt *Visual Clutter* als Verdeckungen bezüglich der Farben auf. Einzelne Farben lassen sich also nicht mehr eindeutig erkennen (vgl.[Rad08, BB04, FWR99]).

Zum Bestimmen von *Visual Clutter* werden Metriken verwendet, um die gegebene visuelle Repräsentationen zu bewerten.
Bertini [BS04] setzt sich mit dieser Problematik auseinander. Er betrachtet dabei die Effektivität einer gegebenen visuellen Repräsentation. Er stellt fest, dass Metriken und Evaluierungsmethoden, welche bestimmen, wie effektiv eine gegebene visuelle Repräsentation ist, immer noch ein offener Forschungsgegenstand sind.
Nachfolgend soll untersucht werden, wie sich *Visual Clutter* bestimmen lässt. Der Fokus liegt dabei auf *Visual Clutter* durch räumliche Verdeckung. Die ersten Arbeite hierzu gehen auf Tufte zurück [Tuf06].
Er stellte bereits sehr früh die Metriken *Data-Ink Ratio* und *Data Density* auf, durch die Verletzungen der Effektivität von gedruckten visuellen Repräsentationen, wie Karten und Diagrammen ermittelt werden konnten.
In der Literatur finden sich neben den Metriken von Tufte auch Ansätze von anderen Autoren, so beispielsweise von Ellis et al. [ED06b] die Metriken, *Overplotting*, *Overcrowded* und *Hidden*, um Rückschlüsse auf *Visual Clutter* in visuellen Repräsentationen ziehen zu können. Neben diesen gibt es auch spezialisierte Metriken, wie *Class Consitency* [SNLH09] oder die *Visual Density* [FTSS09], die vornehmlich mit Scatterplots arbeiten.

Data-Ink Ratio

$$Data\ Ink\ Ratio = \frac{data\ ink}{total\ ink}$$

Beim *Data-Ink Ratio* kann *data-ink* als Anzahl der Pixel aufgefasst werden, die Datenwerte abbilden. Die *total ink* kann als Anzahl der belegten Pixel in der visuellen Repräsentation interpretiert werden. Diese Interpretationen der Metriken von Tufte sind erforderlich, da sich seine Ausführungen auf gedruckte visuelle Repräsentationen beziehen [Tuf06].

Data Density

$$Data\ Density = \frac{number\ of\ entries\ in\ data\ matrix}{area\ of\ data\ graphics}$$

Diese Metrik setzt die Anzahl der zu visualisierenden Datenwerte ins Verhältnis zur der Fläche, die eine visuelle Repräsentation einnimmt [Tuf06].

Overplotting Overplotting ist eine Prozentangabe über Pixel, die mit mehr als einem Datenwert belegt sind. Der Wertebereich geht von 0 (Alle Pixel sind nur einmal belegt.) bis 100 (Alle Positionen sind mehrfach belegt.) [ED06b].

Overcrowded Die Anzahl der Datenwerte in Prozent, welche sich gemeinsame Pixel mit anderen Datenwerten teilen. Der Wertebereich geht von 0 (Es gibt keine Überschneidung von Datenwerten.) bis 100 (Alle Datenwerte überschneiden sich.) [ED06b].

Hidden Prozentuale Anzahl von Datenwerten, welche nicht sichtbar sind, weil sie überzeichnet werden. Es sind Werte von 0 bis etwas weniger als 100 möglich [ED06b].

Class Consistency Diese bestimmt die Qualität einer visuellen Repräsentation von verschiedenen Clustern in einem 2D-Plot, indem die Separierbarkeit der Cluster und der Abstand der Datenpunkte zum Clusterzentrum bestimmt werden (vgl. [SNLH09]).

Visual Density Für einen Scatterplot ist sie definiert als der Durchschnitt der Verhältnisse von Anzahl der Datenwerte im Cluster zur Fläche im Darstellungsraum, die ein Cluster einnimmt [FTSS09].

Auch der Abstand der Betrachter spielt eine Rolle bei der Bewertung von effektiven visuellen Repräsentationen. Mit diesem Problem setzen sich Bennett und Quigley [BQ08] auseinander.

Zur Vermeidung von *Visual Clutter* können verschiedene Methoden herangezogen werden, einen systematischen Überblick dazu liefert Ellis et al. [ED07]. Die Autoren unterscheiden zwischen Methoden, welche das Erscheinungsbild anpassen (Reduzierung der Datendichte), solchen die eine räumliche Verzerrung vornehmen und denjenigen, die eine zeitliche Verzerrung vornehmen:

Reduzierung Datendichte In diese Kategorie fallen Filtermethoden, bei denen die dargestellten Daten nach bestimmten Kriterien vorausgewählt werden. Zudem können die Größe von Repräsentationen einzelner Datenwerte oder deren Transparenz angepasst werden. Als letzte Variante in diesem Bereich geben die Autoren das Clustern von Datenwerten an. Als konkrete Methode kann das Sampling [BS05b, ED06b] genannt werden. Bei diesem werden nicht alle Datenwerte dargestellt, sondern nur bestimmte Datenwerte zur Darstellung ausgewählt. Bertini [BS04] schlägt weiter vor, die *Data Density* zu reduzieren, indem alle Datenwerte, die zusammen in einem Cluster liegen, durch einen einzigen Repräsentanten ersetzt werden. Ein ähnliches Verfahren, bei dem die Datenwerte aber erst im Darstellungsraum zusammengefasst werden, stellen Novotny et al. [NH06] vor. Ihr Ansatz arbeitet mit *Parallelen Koordinaten* [Ins85, ID90]. Hierbei werden einzelne Datenwerte durch ein geeignetes Binning zusammengefasst, und so wird die Lesbarkeit der visuellen Repräsentation verbessert.

räumliche Verdeckung Hierzu gehören Punkt-/Linien-Verschiebung, topologische Verzerrung, raumfüllende Techniken, Pixel-basierte Methoden und Umordnung der dargestellten Daten.
Bertini [BS04] schlägt für diesen Bereich raumfüllende Techniken (space filling techniques) vor, da bei diesen jeder Datenwert seinen eigenen Platz hat und es nicht zu Verdeckungen zwischen den Datenwerten kommt.
Ein Beispiel hierfür sind die Treemaps [Shn92]. Als konkrete Beispiele in diesem Bereich können auch die Arbeiten von Keahey et al. [KR97, Kea98] angegeben werden. Sie befassen sich mit nichtlinearen Verzerrungen in visuellen Repräsentationen sowie Fokus- und Kontext-Techniken.

zeitliche Verzerrung Diesem Bereich ordnen die Autoren die Animation zu.

Ein konkreter Ansatz in diesem Bereich sind *Progressive Informationsdarstellungen*. Hierbei werden in der visuellen Repräsentation schrittweise mehr Details angezeigt. Je nach gegebenem Grad des *Visual Clutter* werden dann mehr bzw. weniger Details in der visuellen Repräsentation ausgegeben (vgl. [TTS09, RS07]).

Zusammenfassend kann gesagt werden, dass in der Literatur Ansätze vorhanden sind, um zu entscheiden, ob *Visual Clutter* vorliegt. Es können aber daraus keine Rückschlüsse gezogen werden, ob es sich dann auch um eine effektive visuelle Repräsentation handelt. Auch ist ein Vergleich des auftretenden *Visual Clutter* bei visuellen Repräsentationen schwierig, welche durch unterschiedliche Techniken erzeugt wurden.
Zu den allgemeinen Strategien, mit denen *Visual Clutter* entgegengewirkt werden kann, ist mit der Arbeit von Ellis et al. [ED07] ein sehr guter systematischer Überblick vorhanden. Konkrete Umsetzungen sind hingegen sehr schwierig und bisher nur in Einzelfällen gelöst, z. B. bei Sips et al. [SNLH09, FTSS09] für Scatterplots und Clusterseparierung.

2.3 Informationsdarstellungen in Smart-Meeting-Rooms

In den meisten Szenarien für *Smart-Meeting-Rooms* werden die Ausgabegeräte für Vorträge vor einem Auditorium oder eine Diskussion zwischen mehreren Teilnehmern eingesetzt [BRH+08, CLB+03].
So lässt sich beispielsweise das System von Chiu et al. [CLB+03] im Bereich der heterogenen Multi-Display-, Multi-User-Umgebung ansiedeln. Es befasst sich mit der Manipulation von Vortragsfolien. Hierzu müssen die Vortragsfolien auf den unterschiedlichen Ausgabegeräten dargestellt werden. Im Vordergrund stehen dabei die Interaktion und die Frage, wie sich mobile Geräte einbinden lassen, um Folien zu annotieren.
Eine Anpassung der visuellen Repräsentation an die unterschiedlichen Ausgabegeräte ist nicht Teil dieses Ansatzes - dieses ist gegenwärtig eine aktuelle Forschungsfrage. Bisher existieren nur erste Ansätze für die Lösung spezifischer Teilprobleme.

Ein Ziel von aktuellen Forschungsarbeiten ist die Anpassung von bestehenden Anwendungen, sodass diese die Möglichkeiten von Multi-User-, Multi-Display-Umgebungen genutzt werden können.
Ein erster Ansatz wurde z. B. von Forlines und Lilien [FL08] veröffentlicht.

Sie stellen ein System vor, mit dessen Hilfe bestehende Anwendungen angepasst werden können. Ihr System ist in der Lage, visuelle Repräsentationen von Molekülen auf einem Table-Display darzustellen und zudem auf mobilen Ausgabegeräten, über die eine präzise Interaktion auf den Molekülen realisiert wird.
Der Fokus dieses Systems liegt allerdings nicht in der automatischen gerätebasierten Anpassung der visuellen Repräsentation.

Die Autoren in Ali et al. [AHFS08] stellen fest, dass bestehende Wissensquellen wie Bücher oder Dokumente an unterschiedliche Ausgabegeräte angepasst werden müssen, wenn diese weiterhin genutzt werden sollen. Für Texte existieren hierfür Lösungen. Sie setzen daher ihren Fokus auf die Anordnung der Abbildungen. Hierfür verwenden sie so genannte Templates. Durch die Templates können unterschiedliche Aufgaben und unterschiedliche Ausgabegeräte berücksichtigt werden. Wird ein entsprechendes Template ausgewählt, laufen alle weiteren Anpassungen automatisch ab.

Ein Ansatz, der Informationsvisualisierungen für mehrere Benutzer in Multi-Display-Umgebungen betrachtet, wird in [WLSS09] vorgestellt.
Die Autoren betrachten sowohl eine Anpassung der visuellen Repräsentation an die Benutzer als auch an die gegebenen Ausgabegeräte. Von den beiden genannten Bereichen fokussieren sie auf die Adaption an die unterschiedlichen Benutzer, indem die visuellen Repräsentationen an die vorliegende Expertise der Benutzer angepasst werden.
Zudem gestattet das System kooperatives Arbeiten zwischen den Benutzern mit mehreren Ausgabegeräten. Eine Adaption an die Ausgabegeräte ist nur exemplarisch vorhanden. So halten sie fest, dass bei einer *Parallelen-Koordinaten-Darstellung* der Daten auf größeren Displays mehr Achsen dargestellt werden können als auf kleineren Displays. Eine systematische Behandlung der gerätebasierten Adaption fehlt aber.

Ein System, das mehr auf den ad-hoc Charakter von einzelnen mobilen Ausgabegeräten fokussiert ist, wird in [JWFS08] vorgestellt.
Durch das System können mobile Ausgabegeräte *On-the-Fly*, also ad-hoc, in eine bestehende Multi-User-, Multi-Display-Umgebung eingebunden und für die Ausgabe von visuellen Repräsentationen eingesetzt werden. Eine Anpassung der visuellen Repräsentationen an die unterschiedlichen Eigenschaften der Ausgabegeräte steht aber auch in dieser Arbeit nicht im Fokus.
Es ist offensichtlich, dass im Bereich der automatischen gerätebasierten Adaption von visuellen Repräsentation in dynamischen Multi-Display-Umgebungen noch ein großer Handlungsbedarf besteht.

2.3.1 Visualisierung in verteilten Geräteensembles

In einem *Smart-Meeting-Room* stehen verschiedene Input-, Output- und Berechnungsgeräte zur Verfügung. Diese sind durch die Softwarebasis zu berücksichtigen.
Die Softwarebasis in einem *Smart-Meeting-Room* muss demnach eine verteilte heterogene Gerätestruktur berücksichtigen. In diesem Abschnitt sollen verfügbare Ansätze in der Literatur betrachtet werden, die sich mit der Visualisierung in einer verteilten Gerätestruktur auseinandersetzen.

Ein erster Ansatz wurde von Brodlie et al. [BDG+04a] für die webbasierte Visualisierung vorgestellt. Sie schlagen eine Client-Server-Architektur vor. Bei dieser Architektur wird die Visualisierungspipeline typischerweise nur einmal getrennt. Der eine Teil wird durch den Server, der andere durch den Client bearbeitet. Die Trennung kann dabei je nach Leistung des Clients an unterschiedlichen Punkten in der Visualisierungspipeline vorgenommen werden.
In jedem Fall erfolgt eine Verteilung immer nur auf zwei Geräte (den Client und den Server). Visualisierungssysteme, die dieses Prinzip im Web umsetzen, finden sich in [BKDE00, ZHHM07].
Auch das System von Skorin-Kapov et al. [SKKM+05] arbeitet webbasiert. Die Trennung erfolgt hier vor dem Rendering. Die letztendliche Ausgabe auf den heterogenen Ausgabegeräten wird durch speziell auf die unterschiedlichen Geräte angepasste Module realisiert.

Dagegen wird beim Data-State-Reference-Model von Chi [CR98] die Visualisierung aus einer Reihe von Modulen oder Komponenten zusammengefügt, so dass sich prinzipiell unterschiedliche Module auf unterschiedlichen Geräten ausführen lassen.
Wenn allerdings auch kooperatives Arbeiten der Benutzer berücksichtigt werden muss, so reicht eine einfache Verteilung von Modulen auf unterschiedliche Geräte nicht aus. Daher erweitern Wood et al. [WWB95, WWB97] die bestehenden Modelle.
Im ersten Schritt duplizieren sie das User-Interface so, dass auf allen betreffenden Ausgabegeräten eine Benutzerschnittstelle vorhanden ist, über welche die Visualisierung gesteuert werden kann.
Aber gerade, wenn mehrere Ausgabegeräte verwendet werden und der Erzeugungsprozess an die unterschiedlichen Ausgabegeräte angepasst wird oder wenn Benutzer die visuelle Repräsentation auf ihren Ausgabegeräten anpassen wollen, ist auch diese Erweiterung noch nicht ausreichend.
Daher sieht ihr Modell vor, dass die gesamte Pipeline oder Teile der Pipeline

dupliziert und verknüpft werden können. In Abbildung 2.6 ist das allgemeine Modell einer kooperativen verteilten Visualisierung dargestellt. Das Modell

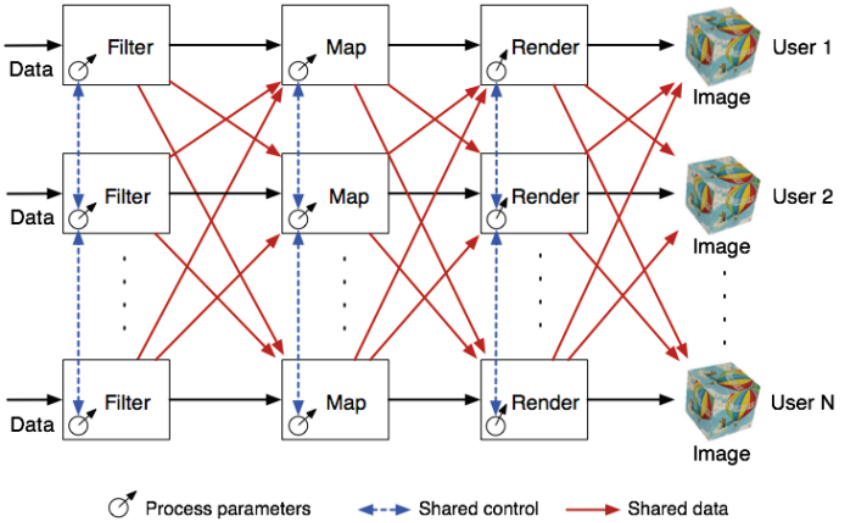

Abbildung 2.6: Generelles Modell einer kooperativen Visualisierung aus [KEM07] (vgl. [WWB95, WWB97])

verfügt über folgende Eigenschaften [KEM07]:

- Daten können an beliebigen Punkten der Pipeline eingespeist und ausgelesen werden.
- Auf jeder Stufe können die Parameter entweder lokal durch den Benutzer oder extern durch einen kooperierenden Benutzer gesetzt werden.
- Module und sogar ganze Pipelines können auf beliebigen Ressourcen durch die Benutzer geladen werden.

Betrachtet man aber den Software-technischen Ansatz, so lassen sich die Ansätze der verteilten Visualisierung entweder den serviceorientierten oder den agentenbasierten Architekturen zuordnen. Die Eigenschaften der Architekturen sowie einige Beispielsysteme werden in den beiden folgenden Abschnitten diskutiert.

2.3.2 Agentenbasierte Visualisierung

Bevor auf Ansätze zur agentenbasierten Visualisierung eingegangen wird, soll der Begriff *Agent* näher definiert werden.
In der Literatur findet sich dazu keine einheitliche Definition. So beschreiben

Ding et al. einen Agenten als ein Computersystem, welches automatisch in einer Umgebung agiert, um zuvor definierte Ziele zu erreichen [DMKS01].
Hagen et al. beschreiben einen Agenten als ein Stück Software, welches Veränderungen in einer Umgebung erkennt und automatisch und selbständig darauf reagiert, um vorher festgelegte Ziele zu erreichen [HBE+00]. Die Definition von Roard und Jones deckt sich mit der bereits angegebenen Definition. Sie verstehen unter einem Agenten ein Stück selbstständige Software, welches mit seiner Umgebung interagieren kann [RJ06]. Tsoi und Gröller erweitern die bereits angegebenen Eigenschaften eines Agenten um dessen Lernfähigkeit [TG00].
Allgemein kann ein Agent definiert werden als:
eine Einheit, die automatisch auf Einflüsse aus der Umwelt reagiert und dabei eine definierte Zielstellung verfolgt.

Agentenbasierte Systeme werden in der Computergrafik bisher noch selten und in der Visualisierung fast gar nicht eingesetzt. Die Dissertation von Germer [Ger09] gibt hierzu einen sehr guten Überblick. Um das prinzipielle Vorgehen zu beschreiben, sollen an dieser Stelle einige allgemeine Beispiele angegeben werden.

Hagen et al. [HBE+00] stellen ein agentenbasiertes Visualisierungssystem vor. Dabei teilen sie die Softwarearchitektur allgemein in drei Schichten ein: der *Kernel Layer*: umfasst grundlegende Visualisierungsaspekte wie Geometrie und Beleuchtung. Im *Extension Layer* sind Hilfsfunktionen abgelegt. Dabei handelt es sich um Methoden für den Datenimport und -export, Methoden zur Geometrieerstellung und unterschiedliche Visualisierungsmethoden. Der *Hardware-Layer* kapselt hardwarespezifische Renderroutinen.
In ihrer agentenbasierten Architektur werden drei unterschiedliche Typen von Agenten eingesetzt:

reactive agents Sie dienen zum Lösen von einfachen elementaren Aufgaben, wie z. B. der Berechnung eines Level-of-Detail oder dem Rendering.

deliberative agents Diese lösen komplexe Aufgaben mit der Unterstützung der *reactive agents*. Sie bilden hierzu eine Hierarchie von Agenten. Dabei realisieren mehrere *reactive agents* die einzelnen Abschnitte der Pipeline; diese werden jeweils von *deliberative agents* gesteuert. Die *deliberative agents* der einzelnen Abschnitte werden ihrerseits wieder durch *deliberative agents* gesteuert, bis zuletzt die gesamte Steuerung in einem einzelnen Agenten zusammenläuft.

performance agents Diese überwachen die *reactive agents* mit Hilfe der Hierarchie von *deliberative agents*.

Dieser von den Autoren vorgestellte agentenbasierte Architektur liegt also eine Hierarchie von Agenten zugrunde. Die angegebene allgemeine Einteilung der Softwarearchitektur lässt sich in der Form nicht ohne weiteres auf die Informationsvisualisierung übertragen.
Dennoch gibt sie Impulse für ein Schichtenmodell für Softwarearchitekturen in der Informationsvisualisierung, die in einer verteilten Gerätestruktur betrieben werden. Im Abschnitt 5.2.2 wird dieser Gedanke aufgegriffen und für ein Frameworkkonzept eingesetzt.

Ein weiteres agentenbasiertes Softwaresystem stellen Roard und Jones [RJ06] vor. Auch dieses System fokussiert auf die Darstellung von 3D-Geometrie (vgl. Abbildung 2.7).
Die Architektur des Systems umfasst drei Bereiche:

Agenten Die Agenten bilden die Basis des Systems und haben jeweils eine fest definierte Aufgabe, wie z. B. das Laden von Daten, das Rendering oder das Bereitstellen von Umgebungsparametern.

Factorys Die Factorys haben die Aufgabe, bei Bedarf neue Agenten zu starten. Das ist z. B. dann der Fall, wenn keine Agenten für eine gegebene Aufgabe vorhanden sind oder wenn ein einzelner Agent überlastet ist und die Berechnung deswegen auf mehrere Agenten verteilt wird.

Steuerbereich Der Steuerbereich umfasst den Factory Manager, den Remote Manager, den Creation Manager und den Name-Server. Die einzelnen Manager haben die Aufgaben, Anfragen zu verarbeiten, neue Factorys zu starten und gestartete zu überwachen und die eingebundenen Ressourcen gleichmäßig auszulasten (Load Balancing Problem [MCEF94]). Dem Name Server fällt die Aufgabe zu, die Agenten zu registrieren.

Dieser Ansatz ist im Rahmen dieser Arbeit besonders interessant, weil mehrere unterschiedliche Ressourcen eingesetzt werden können, um ein gegebenes Problem zu bearbeiten. Bei Bedarf werden neue Agenten ausgeführt, so können andere Ressourcen entlastet werden. Die Agenten selbst werden mehr durch die Komponenten im Steuerbereich gesteuert und verwaltet, als dass eine Initiative von ihnen ausgeht.

Ein agentenbasierter Ansatz im Bereich der Informationsvisualisierung findet sich bei Tsoi und Gröller [TG00]. Siehe hierzu auch Abbildung 2.8.

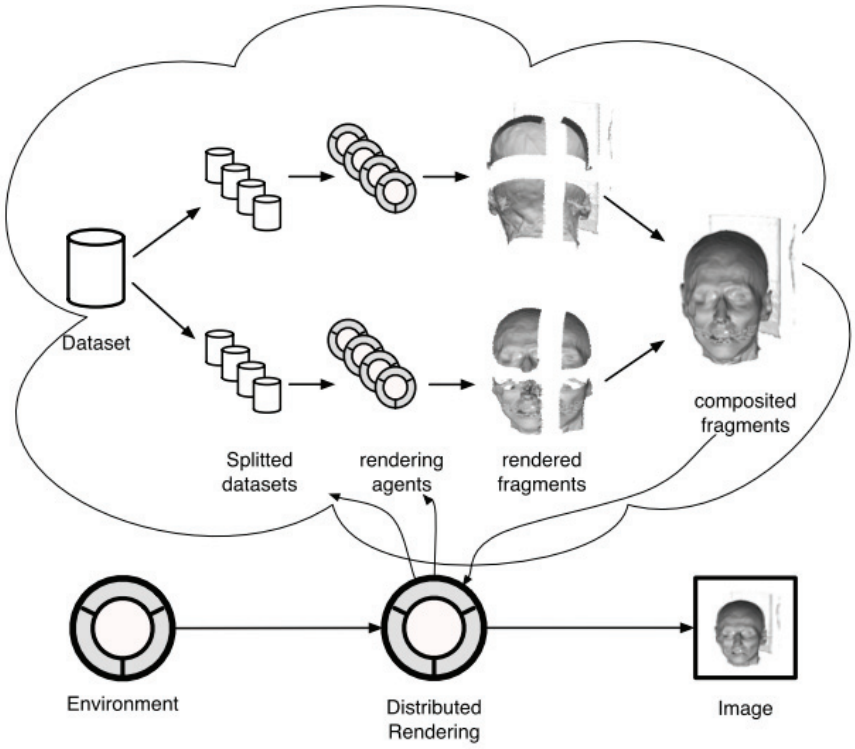

Abbildung 2.7: Einsatz von mehreren Render-Agenten zur Generierung eines Ausgabebildes, aus [RJ06].

Die Autoren haben sich für ein agentenbasiertes System entschieden, da sie der Meinung sind, dass sich ein solches System leichter an die veränderlichen und heterogenen Ressourcen, wie sie im Internet geben sind, anpassen kann. Das System (AVAM) teilt sich in die drei Bereiche Sensoren, Arbitrator und Visualisierungsagenten.
Durch die Sensoren werden dem System Informationen über die Umgebung, die Ressourcen und auch Informationen über die Daten bereitgestellt.
Der Arbitrator entscheidet darüber, wo ein Agent ausgeführt wird, sein Aufgabenbereich ist somit das Load-Balancing [MCEF94].
Die Agenten realisieren unabhängige Visualisierungsmodule in der Pipeline. Zudem setzen sie die Entscheidung des Arbitrators um.
Der vorgestellte Ansatz umfasst zudem ein Entscheidungsfindungssystem, welches mit einer Fuzzy-Logik und einem neuronalen Netz arbeitet. Das System ist somit lernfähig.

Einen weiteren interessanten Ansatz stellen Schädlich und Mukasa [SM04]

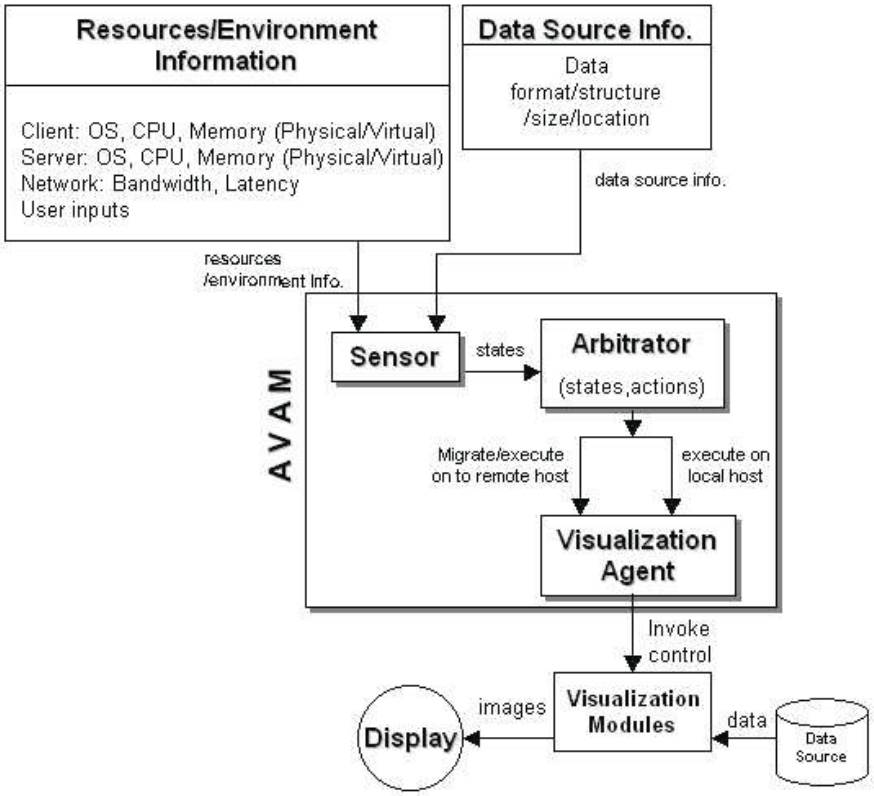

Abbildung 2.8: Schematische Übersicht über AVAM, aus [TG00].

vor. Sie setzen ein adaptives agentenbasiertes Softwaresystem im Bereich der Produktionsautomatisierung ein. Sie verwenden dazu zwei unterschiedliche Typen von Agenten:

reactive agents Diese verfügen nur über ein eingeschränktes Kontextwissen. Sie sind dabei jeweils Modulen oder Modulgruppen zugeordnet, die sie parameterisieren. Sie realisieren in der Visualisierungspipeline das Filtering und Rendering.

deliberative agents Diese lösen hingegen komplexe Probleme. Sie verfügen dazu über ein breiteres Kontextwissen. Von diesem Typ ist üblicherweise nur ein Agent im System vorhanden. Zum Lösen des gegebenem Problems kontrolliert dieser Agent mehre *reactive agents*. Der *deliberative agent* übernimmt in der Visualisierungspipeline das Mapping.

Die Auswahl einer geeigneten Visualisierungtechnik wird in der vorgestellten Architektur durch ein Regelsystem gesteuert. Die Grundarchitektur weist

Parallelen zu dem Ansatz von Hagen et al. [HBE+00] auf.

Ein weiteres dezentrales agentenbasiertes Softwaresystem beschreibt Moere [Moe07]. Dieses System gehört in den Bereich der selbstorganisierenden Systeme.
Während die meisten agentenbasierten Systeme häufig ganze Stufen der Pipeline als Agenten realisieren, bearbeiten Moeres Agenten jeweils nur ein einzelnes Datum, ein Datentupel oder eine Datenspalte einer Datenbank. Der Agent ist dafür zuständig, diese Daten auf ein visuelles Primitiv zu mappen. Dabei berücksichtigt jeder Agent die visuellen Variablen [Ber82, Mac86]. Zudem kommunizieren die Agenten untereinander, um sich aneinander anzupassen.
Durch den Ansatz von Moere ergeben sich ganz neue visuelle Repräsentation (vgl. Abbildung 2.9). Der Autor räumt allerdings ein, dass dieser Ansatz noch einige technische und konzeptionelle Hürden nehmen muss, bevor er praxistauglich wird.
Bei der vorliegenden Systemarchitektur wird prinzipiell immer derselbe Typ von Agent verwendet. Es gibt demzufolge keine übergeordneten Agenten, die andere Agenten steuern. Die Steuerung ist also dementsprechend in jeden einzelnen Agenten integriert [Moe07].

Neben dem Ansatz von Moere findet sich in der Literatur auch der von

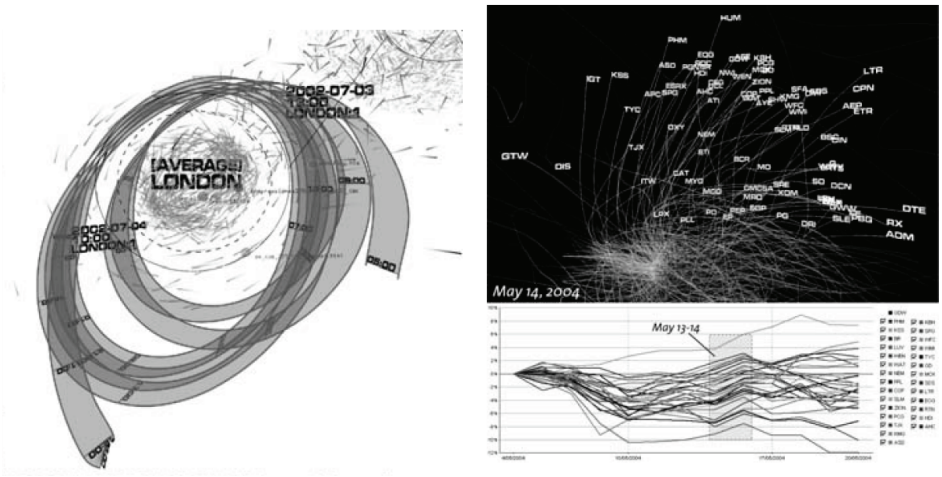

Abbildung 2.9: Visuelle Repräsentationen, die mit dem Agenten-basierten System von Moere erzeugt wurden, aus [Moe07].

Germer [Ger09], der ebenfalls auf selbstorganisierenden Agenten basiert.

Es soll ein weiteres agentenbasiertes System vorgestellt werden, welches aber außerhalb der Visualisierung eingeordnet werden muss. Das System RAJA von Ding et al. [DMKS01] arbeitet mit zwei Typen von Agenten:

Basisagenten Diese führen Domain-spezifische Aufgaben aus.

Metaagenten Diese überwachen und verwalten die verfügbaren Ressourcen im System. Sie kümmern sich um die gleichmäßige Auslastung der Ressourcen und starten sowie steuern die Basisagenten.

Auch in diesem System wird eine Einteilung in zwei Schichten vorgenommen: eine ausführende Schicht und eine steuernde Schicht.

Bei agentenbasierten Systemen wird die Steuerlogik entweder durch extra Agenten realisiert, die dann ihrerseits andere Agenten steuern, oder sie ist wie im Fall der selbstorganisierenden Systeme Teil eines jeden Agenten.
Agentenorientierte Systeme sind in der Regel robust, da sie Buttom-Up-Ansätze verfolgen [Moe07]. Zudem können sie sich dynamisch an eine veränderte Umgebung anpassen [Ger09].
Neben diesen Vorteilen haben agentenbasierte Systeme auch Nachteile. Sie stellen hohe Leistungsansprüche, da die Verhaltensregeln jedes Agenten in jedem Entwicklungsschritt neu angewendet werden müssen. Zudem kann nicht immer gesagt werden, wann ein selbstorganisierendes System seinen „Gleichgewichtszustand“ gefunden hat (vgl. [Moe07]). Dabei ist gerade der verstärkte Rechenaufwand auf leistungsschwachen Ausgabegeräten nicht von Vorteil.

2.3.3 Service-orientierte Visualisierung

Servicebasierte Architekturen werden vor allem für Visualisierungen im Word Wide Web eingesetzt. Das Prinzip besteht darin, die einzelnen Arbeitsschritte zum Erzeugen einer visuellen Repräsentation in Services zu kapseln.
Allgemein sind Services Softwareeinheiten, die eine gewisse Funktionalität kapseln und diese über ein wohl definiertes Interface bereitstellen [HEHB08, DAB05]. Services sind in der Regel in serviceorientierte Architekturen (SOA) eingebunden. Diese sind charakterisiert durch zwei grundlegende Eigenschaften [HEHB08]:

1. Die gesamte Architektur ist auf eine fein granulare Unterteilung ausgerichtet.

2. Die einzelnen Services sind nur lose gekoppelt.

Eine SOA wird in drei Hauptbestandteile unterteilt. In [HS05] werden die drei Komponenten als *Provider*, *Consumer* und *Registry* bezeichnet[7]. Dabei meldet der Service *Provider* den Service bei der *Registry* an. Dort sucht der *Consumer* nach einem benötigten Service und führt diesen aus.

Bei einer SOA sind zwei unterschiedliche Kommunikationsstrukturen üblich [DAB05]:

Direkte Kommunikation Hierbei kommunizieren die Services direkt miteinander

Indirekte Kommunikation Hierbei wird die Kommunikation über einen *Service Broker* abgewickelt

Ein weiterer wichtiger Punkt ist die Komposition von Services. Yang und Papazoglou [YP02] geben in ihrer Veröffentlichung drei Strategien für mögliche Kompositionen von Services an:

Explorative Komposition Die Servicekomposition wird zur Laufzeit durch die Anfrage eines Benutzers bestimmt. Der Benutzer beschreibt dabei, was er benötigt, das System ermittelt, was verfügbar ist und stellt mögliche Kompositionspläne auf. Durch ein Ranking wird einer der Pläne ausgewählt. Dabei können unter anderem Kosten, Performance und Verfügbarkeit eine Rolle spielen.

Semi-statische Komposition Einige der Serviceaufrufe werden zur Laufzeit bestimmt. Andere Aufrufe sind fest definiert und unterliegen somit keiner Dynamik. Dadurch wird die Flexibilität des Systems eingeschränkt, da fest zugeordnete Services in jedem Fall verfügbar sein müssen.

Statische Komposition Die Servicekomposition ist hierbei vordefiniert. Diese Variante ist von den drei angegebenen die mit der geringsten Flexibilität. Ein Vorteil ist, dass der aufwendige Bewertungsschritt von Kompositionsplänen entfällt.

[7] In [SG08] werden die Komponenten als *Service Provider*, *Service Consumer* und *Service Broker* bezeichnet.

Bisher gibt es in der Visualisierung nur sehr wenige Ansätze, die eine serviceorientierte Architektur verwenden.
Ein Vertreter der serviceorientierten Softwaresysteme in der Visualisierung, welches mit Web-Services arbeitet, wird in [PBB+08] vorgestellt. Das System hat zum Ziel, visuelle Analyseverfahren einer breiteren Masse zugänglich zu machen. Das System unterstützt die schnelle Portierung von Client-basierten Analyseanwendungen auf unterschiedlichen Plattformen.

Zudilova-Seinstra und Yang [ZSY05] stellen einen serviceorientierten Ansatz für interaktive Visualisierungen vor. Der Ansatz verwendet die übliche Visualisierungspipeline. Zusätzlich wird dieser noch eine weitere Präsentationsstufe nachgeschaltet.
Alle relevanten Berechnungen werden durch Services realisiert. So gibt es beispielsweise Datentransfer-Services, Filter-Services, Mapping-Services, Rendering-Service und Präsentationsservices. Die gesamten Berechnungen in der Visualisierungspipeline sind demzufolge durch Services gekapselt [ZSY05]. Das System sieht in der gegenwärtigen Form keine Adaption an heterogene Ausgabegeräte vor.

2.4 Zusammenfassung

Viele Ansätze adressieren spezifische Fragestellungen, wie die Adaption der Visualisierung bezüglich der Daten oder der Aufgaben. Vereinzelt finden sich auch Ansätze, welche Aspekte wie Multi-Display oder auch die Verteilung der Erzeugung auf mehrere Geräte betrachten. Es finden sich aber keine Ansätze, welche die Erfordernisse in einem *Smart-Meeting-Room* vorliegen, geschlossen betrachten und für Informationsrepräsentationen anwenden. Insbesondere werden ad-hoc-Ausgabegeräte nicht von den vorhandenen Ansätzen berücksichtigt. Darum ist ein wichtiges Anliegen dieser Dissertation ein Konzept zu entwickel, welches eine Adaption an multiple, heterogene und dynamische Ausgabegeräte erlaubt.

Kapitel 3

Anforderungsanalyse und Problemdiskussion

Zunächst soll die *Ausgangssituation* klar umrissen und mit Beispielen veranschaulicht werden, um daraus Anforderungen und Probleme abzuleiten.
Smart-Meeting-Rooms sind im Allgemeinen *Multi-Display-Umgebungen*. Das bedeutet, dass im Geräteensemble des *Smart-Meeting-Rooms* mehrere Ausgabegeräte mit unterschiedlichen Eigenschaften für die Ausgabe von visuellen Repräsentationen zur Verfügung stehen. Ein solches Geräteensemble kann sowohl statische Geräte enthalten als auch dynamische, die von den Benutzern mit in den *Smart-Meeting-Room* gebracht werden.
In der Regel arbeiten in einem *Smart-Meeting-Room* mehrere Benutzer kooperativ zusammen. Dies macht ihn zu einer *Multi-User-Umgebung*. Dabei wird davon ausgegangen, dass mehrere der zur Verfügung stehenden Ausgabegeräte von den Benutzern gemeinsam verwendet werden.
Für das Geräteensemble eines *Smart-Meeting-Rooms* leiten sich daraus die folgenden Eigenschaften ab:

Die Anzahl der Geräte im Ensemble ist nicht konstant Das bedeutet, dass Geräte jederzeit zum Ensemble hinzukommen oder dieses wieder verlassen können. Beispielsweise kommen Geräte hinzu, wenn weitere Teilnehmer einer Diskussionsrunde ihre mobilen Geräte mitbringen. Andererseits fallen Geräte weg, wenn Teilnehmer mit ihren mobilen Geräten ein Meeting verlassen. Die dynamischen Veränderungen des Geräteensembles finden also *On-the-Fly* statt, das heißt während eines Vortrages oder während einer Diskussion.

Die Geräte haben unterschiedliche Eigenschaften Verschiedene Geräte ergeben sich deshalb, weil jeder Benutzer mit dem ihm vertrauten Gerät arbeiten will [BDG+04a]. Heterogene Geräte zusammen mit der

Dynamik der Geräte implizieren, dass keine feste Gerätestruktur im *Smart-Meeting-Room* vorausgesetzt werden kann. Die Eigenschaften des Geräteensembles können sich also *On-the-Fly* verändern.

Nicht-exklusive Nutzung der Geräte durch Prozesse[1] Durch Prozesse können sich die Eigenschaften der Geräte im Ensemble *On-the-Fly* verändern, da nicht immer ein Prozess ein Gerät exklusiv nutzen kann. Das heißt, dass auf einem einzelnen Gerät mehrere Prozesse parallel ausgeführt werden und sie somit um die Ressourcen auf dem Geräte konkurrieren. Fällt ein laufender Prozess auf einem Gerät weg oder kommt ein Neuer hinzu, so ändern sich aus Sicht der einzelnen Prozesse die Ressourcen und damit möglicherweise auch die Eigenschaften des betreffenden Gerätes.

Ein *Smart-Meeting-Room* ist also durch ein *dynamisches heterogenes Geräteensemble* charakterisiert. Zudem muss eine proaktive und automatische Unterstützung der Benutzer realisiert werden (vgl. 2.1).

3.1 Anforderungen

Entsprechend der beschriebenen Ausgangssituation sollen nun an dieser Stelle die Anforderungen an Informationsvisualisierungen in einem *Smart-Meeting-Room* betrachtet werden. Hierzu gehören:

1. **Auftretende Veränderungen des Geräteensembles müssen automatisch erkannt werden**. Das schließt das Erkennen von hinzukommenden und wegfallenden Geräten ein, ebenso das Erkennen von sich ändernden Eigenschaften für laufende Prozesse.

2. **Die Informationsdarstellung muss an das aktuelle Geräteensemble angepasst werden**. Das schließt auch ein, die Visualisierungspipeline zum Erzeugen der visuellen Repräsentation unter Berücksichtigung der Eigenschaften des aktuellen Geräteensembles *On-the-Fly* zu konfigurieren. Das heißt, bei jeder Veränderung des Geräteensembles im *Smart-Meeting-Room* muss automatisch überprüft werden, ob die aktuelle visuelle Repräsentation und damit die sie erzeugende aktuelle Visualisierungspipeline gültig bleibt (vgl. 2.2.2) oder ob gegebenenfalls Anpassungen erforderlich sind.

3. **Ein und dieselbe visuelle Repräsentation muss auf mehreren Ausgabegeräten gleichzeitig angezeigt werden können.** Dabei

muss gewährleistet sein, dass dieselben Informationen bzw. dieselben visuellen Repräsentationen auf vergleichbaren Geräten auch vergleichbar repräsentiert werden. Es sind also für den Benutzer unnötige Wechsel bzgl. seiner „mental map“ zu vermeiden. Auch **verschiedene visuelle Repräsentationen sollen auf heterogenen Ausgabegeräten dargestellt werden** können.

3.2 Problemdiskussion

Im Folgenden sollen entsprechend des Abschnitts 3.1 die dabei auftretenden Probleme diskutiert werden.

3.2.1 Erkennen von Veränderungen des Geräteensembles

Bisher wurden im Umfeld der Informationsvisualisierung visuelle Repräsentationen in der Regel für ein festes Ausgabegerät erzeugt. In *Smart-Meeting-Rooms* dagegen muss auf Grund der dynamischen Änderungen des Geräteensembles die aktuell vorliegende Gerätesituation erkannt werden.
Für jedes Ausgabegerät müssen dessen Eigenschaften bekannt sein, um die visuelle Repräsentation entsprechend anpassen zu können. Verändern sich die Eigenschaften eines Ausgabegerätes, muss dies ebenfalls geeignet kommuniziert und bei der Adaption der visuellen Repräsentation berücksichtigt werden.

Bisher stand eine Dynamik der Ausgabegeräte nicht im Vordergrund der Forschungen im Bereich der Informationsvisualisierung (vgl. 2.2.3). Somit fehlt es in diesem Bereich in der Literatur an geeigneten und etablierten Beschreibungen von Geräteeigenschaften (vgl. 2.2.2), die für die visuelle Ausgabe von Bedeutung sind. Aber gerade diese Beschreibungen sind bei der Ausgabe von visuellen Repräsentationen in *Smart-Meeting-Rooms* notwendig, um visuelle Repräsentationen adäquat an die Ausgabegeräte anpassen zu können.
Z. B. muss die Frage beantwortet werden, ob es sich beim Wechsel eines Ausgabegerätes beim neuen Gerät um ein größeres, kleineres oder ein Ausgabegerät mit gleicher Displayfläche handelt. Diese Fragestellung ist wichtig, um im Voraus abschätzen zu können, ob die Ausgabe der visuellen Repräsentation auf dem neuen Ausgabegerät Anpassungen erfordert. Das Ermitteln von vorliegenden Veränderungen der Geräteeigenschaften liegt im Aufgabenbereich des Software-Framework. Dieses muss die veränderten Eigenschaften

an die entsprechende *Adaptionsstrategie* weiterleiten.

3.2.2 Problem der Anpassung der visuellen Repräsentation

Die Dynamik des Geräteensembles erfordert *On-the-Fly* Anpassungen der visuellen Repräsentation. Es ist eine geeignete Basis erforderlich, auf der entschieden werden kann, ob eine Adaption erforderlich ist. Zu diesem Zweck werden Metriken eingesetzt. Auf diese wurde bereits im Abschnitt 2.2.4 im Grundlagenkapitel eingegangen. In dieser Arbeit wird bei der Adaption der visuellen Repräsentation auf die Eigenschaften Displayfläche und Auflösung fokussiert, und dabei werden drei Fälle unterschieden:

1. **Anpassungen bei gleicher Displayfläche:** Verändert sich die Displayfläche nicht, kann davon ausgegangen werden, dass die Gültigkeit der visuellen Repräsentation erhalten bleibt. Gegebenenfalls müssen Anpassungen in der Parametrisierung vorgenommen werden, z. B. Anpassungen von Farbskalen. Hinzu kommt, dass unter Verwendung spezieller Konzepte, wie z. B. einer progressiven Informationsanzeige (vgl. 4.5), weitere Eigenschaften des Ausgabegerätes, wie die zur Verfügung stehende Bandbreite, einzubeziehen sind.

2. **Anpassungen bei kleinerer Displayfläche:** Ist das aktuelle Display kleiner als das bisher zur Verfügung stehende und wird dieselbe visuelle Repräsentation verwendet, müssten ohne Anpassungen zwangsläufig mehr Informationen auf weniger Raum dargestellt werden. Es ergeben sich folgende Probleme:
   Im Allgemeinen tritt *Visual Clutter* auf (vgl. 2.2.4). Um dieses zu reduzieren gibt es verschiedene Methoden. Die Herausforderung besteht darin zu entscheiden, wann welche Methode eingesetzt wird und wie stark dabei die Informationsreduktion sein muss.

3. **Anpassungen bei größerer Displayfläche:** Die Wahl des neuen Ausgabegerätes kann auch auf ein Gerät mit größerer Displayfläche fallen. Hierbei behält die bisherige visuelle Repräsentation nicht automatisch ihre Gültigkeit. So kann sich die Wahrnehmung der Verteilung der Datenwerte sowie von lokalen Bereichen mit erhöhter Dichte verändern (vgl. [FTSS09, BS04]). Dieses Problem wurde in der Literatur bisher kaum behandelt. Auch hier besteht die Herausforderung darin, *Adaptionsmechanismen* zu identifizieren und einzuordnen, welche diese Be-

sonderheiten berücksichtigen und die visuelle Repräsentation an das neue Ausgabegerät anzupassen.

Allgemein betrachtet kann eine Adaption einer visuellen Repräsentation entweder durch einen Eingriff in die Visualisierungspipeline erfolgen oder auf Ebene der visuellen Repräsentation.

1. Wird die Adaption unter Verwendung der gesamten Visualisierungspipeline durchgeführt, bieten sich mehrere Ansatzpunkte dafür an. Es gibt die Möglichkeit, Anpassungen im *Datenraum* vorzunehmen oder Anpassungen im *Darstellungsraum* durchzuführen. Die Herausforderung besteht darin, für beide Varianten geeignete *Adaptionsmechanismen* bereitzustellen. Hierbei muss eine geschickte Adaption vorgenommen werden, um den Verlust von für den Benutzer relevanten Informationen zu vermeiden und das aktuelle Ausgabegerät zu berücksichtigen (vgl. [ED07, ED06b, NH06, FTSS09]).

2. Eine Adaption, die ausschließlich auf der Ebene der visuellen Repräsentation arbeitet, bietet den Vorteil, dass im Grunde die gleiche Darstellung ausgegeben wird. Der Benutzer hat es einfacher, sich in die adaptierte Variante hineinzufinden. Nachteilig wirkt sich aus, dass bei dieser Strategie die Auswahl an *Adaptionsmechanismen* stark eingeschränkt ist, da nur *Mechanismen* verwendet werden können, die auf den Bilddaten arbeiten. Die Herausforderung besteht darin, geeignete *Mechanismen* bereitzustellen und für diese die notwendige Parametrisierungen zu finden.

Stehen unterschiedliche *Adaptionsmechanismen* bereit, liegt eine weite Herausforderung darin, einen adäquate *Adaptionsmechanismus* auszuwählen. Hierfür muss eine geeignete *Adaptionsstrategie* bereitgestellt werden, welche automatisch die Auswahl eines *Adaptionsmechanismus* unter Berücksichtigung des aktuellen Ausgabegerätes vornimmt.

3.2.3 Gleichzeitige Informationsdarstellung auf mehreren Ausgabegeräten

Wird die gleiche visuelle Repräsentation gleichzeitig auf mehreren unterschiedlichen Ausgabegeräten dargestellt und sind daher unterschiedliche Anpassungen der visuelle Repräsentation erforderlich, so ergeben sich in diesem Zusammenhang weitere Probleme, die zu lösen sind.
Es muss gewährleistet werden, dass die Unterschiede zwischen den einzelnen

Repräsentationen so gering wie möglich sind, um die „mental map“ der Benutzer zu erhalten. Stattdessen ist es auch möglich, den Benutzer langsam an die adaptierte visuelle Repräsentation heranzuführen, so dass er nachvollziehen kann, wie er die aktuelle Repräsentation im Bezug zur bisherigen Repräsentation zu interpretieren hat.

3.2.4 Verschiedene visuelle Repräsentationen verteilt auf heterogene Ausgabegeräte in Smart Meeting Rooms

Durch die gleichzeitige Verwendung von mehreren Ausgabegeräten in *Smart-Meeting-Rooms* (vgl. 2.1) ergeben sich neue Herausforderungen. Dabei können grundlegend drei Situationen (siehe Abbildung 3.1) bei der Ausgabe von verschiedenen visuellen Repräsentationen differenziert werden:

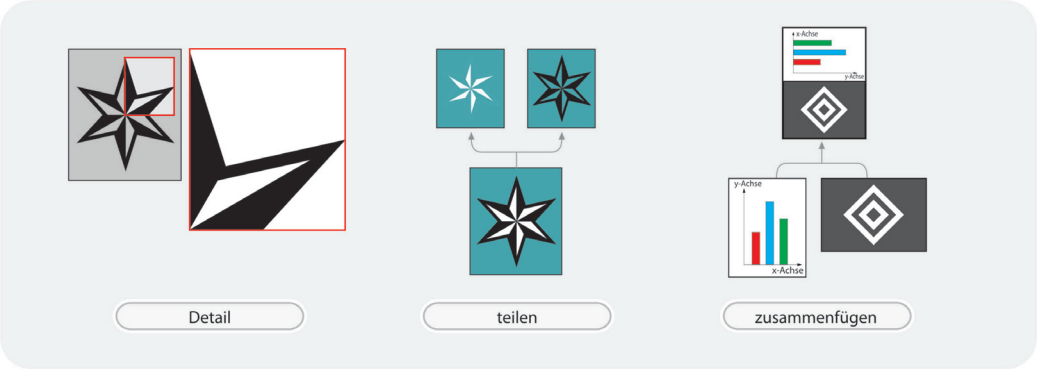

Abbildung 3.1: Drei unterschiedliche Varianten bei der Darstellung von mehreren visuelle Repräsentationen in einer Multi-Display-Umgebung.

1. Auf einem Ausgabegerät werden Detailinformationen zu einer bereits dargestellten visuellen Repräsentation angezeigt. Die Herausforderung besteht darin, die Visualisierungspipeline an einer geeigneten Stelle zu verzweigen und eine Darstellung der Details auf einem weiteren Ausgabegerät zu erlauben.

2. Eine gegebene visuelle Repräsentation wird geteilt[2]. Auch hier steht die Herausforderung im Vordergrund, dass die Teile der visuellen Repräsentationen auf mehr als einem Ausgabegeräte dargestellt werden müssen. Das schließt eine adäquate Adaption der visuellen Repräsentation ein.

[2]Beim Teilen wird eine visuelle Repräsentation in mehrere Teile zerlegt. Die einzelnen Teile haben dabei dasselbe Abstraktionslevel.

3. Stehen weniger Ausgabegeräte zur Verfügung als visuelle Repräsentationen angezeigt werden sollen, so können entweder nicht alle Repräsentationen angezeigt werden, oder es müssen mehrere visuelle Repräsentationen zusammen auf einem Ausgabegerät angezeigt werden. Hierdurch ergibt sich das Problem, dass sich aus Sicht der einzelnen Prozesse die Eigenschaften des Ausgabegerätes verändern.

4. Zudem können auch Mischformen der zuvor genannten Situationen auftreten.

Um diese Probleme lösen zu können, ist ein geeignetes Modell einer Visualisierungspipeline notwendig. Bei diesem muss beachtet werden, dass die Pipeline grundsätzlich auf mehreren Geräten ausgeführt wird (vgl. 2.1.1) und zusätzlich die visuelle Repräsentation gleichzeitig für mehrere Ausgabegeräte angepasst werden muss. Darüber hinaus muss diese robust genug sein, um auch bei wegfallenden Geräten weiterhin visuelle Repräsentationen erzeugen zu können. Diese Herausforderungen sind vorrangig bei der Konzeption des Software-Framework zu berücksichtigen.
Untersuchungen zu visuellen Repräsentationen in Multi-Display-Umgebungen finden sich nur vereinzelt in der Literatur (vgl. 2.3.1), da in der Vergangenheit zumeist auf Single-Display-Umgebungen gesetzt wurde. Aber gerade die Optionen, die sich durch mehrere Displays ergeben, die kooperativ verwendet werden, erschließen neue Anwendungen und Möglichkeiten in *Smart-Meeting-Rooms* gegenüber herkömmlichen Single-Display-Umgebungen. Die in dieser Problemdiskussion aufgeworfenen Fragen sind in der gängigen Literatur noch gar nicht oder nur unter einem anderen Blickwinkel betrachtet worden. Ziel dieser Dissertation ist die Lösung der hier aufgeworfenen Fragen.

3.3 Lösungsansätze

3.3.1 Erkennen von Veränderungen des Geräteensembles

Zur Überwachung der Veränderungen des Geräteensembles wird in dieser Arbeit eine serviceorientierte Architektur (vgl. 2.3.3) verwendet. Die einzelnen Geräte werden dabei von den Services überwacht, die auf diesen ausgeführt werden. Dadurch ist bekannt, welche Geräte im Geräteensemble vorhanden sind und auch wie deren Eigenschaften aussehen. Eigenschaften, die nicht automatisch von den Services auf den einzelnen Geräten bestimmt werden können, machen eine Benutzerabfrage erforderlich. Verändern sich die Geräteeigenschaften, so liegt es im Aufgabenbereich der Steuerschicht der

Services, angemessen auf Änderungen der Geräteeigenschaften zu reagieren (vgl. [TTS09]). Die Lösung der in diesem Bereich aufgeworfenen Probleme wird durch das Konzept des Framework in Kapitel 5 bereitgestellt.

3.3.2 Anpassung der visuellen Repräsentation an das zur Verfügung stehende Geräteensemble

Die Adaption der visuellen Repräsentation auf unterschiedliche Ausgabegeräte erfordert verschiedene *Adaptionsmechanismen*. Diese werden in Kapitel 4 bereitgestellt.
Zu Beginn wird eine Klassifikation von *Adaptionsmechanismen* vorgestellt. Für jede hierbei identifizierte Klasse wird im Rahmen der vorliegenden Arbeit exemplarisch eine Methode entworfen und umgesetzt. Zunächst wird hierbei zwischen Adaption auf Bild- und Prozessebene unterschieden. Als konkrete Lösung für die Bildebene wurde die *inhaltsbasierte Skalierung* auf Informationsvisualisierungen übertragen (vgl. Abschnitt 4.2).
Weiterhin wurden *Adaptionsmechanismen* identifiziert, die auf der Prozessebene arbeiten. Hierbei werden Adaptionen im Datenraum, Darstellungsraum und für das Mapping vorgestellt. Als *Adaptionsmechanismus* im Datenraum findet ein hierarchisches Clusterverfahren Anwendung. Zur Adaption im Darstellungsraum wird ein erweitertes Binning bereitgestellt, welches auf dem Binning von Novotny et al. [NH06] aufbaut. Als eine Adaption, die das Mapping in der Informationsvisualisierung beeinflusst, wird konkret eine Helligkeitsadaption verwendet.
Die Ergebnisse für Adaptionen im Bereich der Prozessebene wurden auf dem CoVis Workshop auf der VisWeek 2009 veröffentlicht [FTSS09].
Weiterhin wurden globale und lokale *Adaptionsmechanismen* gegenübergestellt. Als Lösung für lokale Anpassungen wurden an dieser Stelle die *intelligenten Linsen* eingeführt, welche eine lokale Adaption der visuellen Repräsentation auf allen Stufen der Visualisierungspipeline erlauben. Die Ergebnisse hierzu wurden in gemeinsamen Veröffentlichungen als interaktives Poster auf der InfoVis 2007 und als Artikel auf der Smart Graphics 2008 publiziert [FTS07, TFS08a].
Als eine weitere Klasse von *Adaptionsmechanismen* konnte die progressive Informationsdarstellung identifiziert werden. Besonders hervorzuheben ist die Eigenschaft der progressiven Informationsdarstellung, die Daten nur einmal zu kodieren, aber mehrfach für Adaptionen auf unterschiedliche Ausgabegeräte zu verwenden. Die Ergebnisse hierzu wurden auf der IMC 2009 veröffentlicht [TSR09].

In der Problemanalyse wurde herausgestellt, dass entschieden werden muss, durch welche der zur Verfügung stehenden *Adaptionsmechanismen* die vorliegende visuelle Repräsentation an ein Ausgabegerät adaptiert wird. Als Ergebnis wurde für die *Adaptionsmechanismen*, die auf der Prozessebene arbeiten, eine *Adaptionsstrategie* in Form eines Entscheidungsbaums bereitgestellt. Dieser entscheidet anhand von geeigneten Metriken, ob die Adaption im Datenraum, Darstellungsraum oder für das Mapping durchgeführt wird. Der Entscheidungsbaum wurde auf dem CoVis Workshop auf der VisWeek 2008 veröffentlicht [FTSS09].

3.3.3 Gleichzeitige Informationsdarstellung auf mehreren Ausgabegeräten

Um visuelle Repräsentationen auf verteilten, heterogenen Ausgabegeräten darstellen zu können, wurde ein angepasstes Modell einer Visualisierungspipeline entwickelt. Um einen Operator abstrakt beschreiben zu können, wurden die Operatoren in Operator-Interface und Operator-Implementierung getrennt. Ersteres beschreibt die Funktionalität, die ein Operator leistet. Die Implementierung eines Operators umfasst die konkrete Umsetzung der Funktionalität.
Unter Verwendung dieser Trennung wurde das Konzept der Pipeline-Templates entwickelt. Diese beschreiben eine Visualisierungspipeline abstrakt unter der Verwendung der Operator-Interfaces. In den Pipeline-Templates wird also weder spezifiziert wie noch wo die einzelnen Operatoren umgesetzt werden. Dieses Vorgehen erlaubt eine *Virtualisierung der Visualisierungspipeline*, da jetzt die Operatoren auf beliebigen Geräten berechnet werden können, ohne dass es für die Visualisierungspipeline eine Rolle spielt.
Als Umsetzung der Operator-Implementierung werden in dem hier vorgestellten Konzept Services verwendet. Diese Services können nun auf verschiedenen Ausgabegeräten zur Darstellung von visuellen Repräsentationen verwendet werden.
Entsprechend des vorliegendes Ausgabegerätes werden dabei die Pipeline-Templates um entsprechende Operator-Interfaces erweitert, um so eine Adaption der visuellen Repräsentation an unterschiedliche Ausgabegeräte zu erreichen. Das vorgestellte Konzept wird im Kapitel 5 erarbeitet. Der prinzipielle Framework-Entwurf wurde auf der IV 2009 veröffentlicht [TTS09].
Sollen dieselben visuellen Repräsentationen auf verschiedenen Geräten dargestellt werden, muss zudem die geforderte Vergleichbarkeit der Darstellungen gewährleistet werden. Hierfür eignen sich insbesondere die lokale und progressive Anpassung. Im ersten Fall verändert sich nur eine kleine loka-

le Umgebung, im zweiten Fall wird das Bild schrittweise aufgebaut, so dass die „mental map“ erhalten bleibt. Beide Mechanismen werden im folgenden Kapitel genauer vorgestellt.

Kapitel 4

Adaption der Informationsdarstellung

4.1 Einordnung

Es gibt die unterschiedlichsten Ansätze, um eine Adaption der visuellen Repräsentation vorzunehmen (vgl. 2.2.3). Ziel dieser Promotion ist es, prinzipielle Vorgehensweisen zu unterscheiden und exemplarisch für jede Klasse von Ansätzen eine konkrete Technik zu entwickeln und umzusetzen. Die *Adaptionsmechanismen* lassen sich bezüglich ihrer Eigenschaften wie folgt klassifizieren:

Adaption auf Bildebene vs. Adaption auf Prozessebene Es wird unterschieden, ob die visuelle Repräsentation oder der Prozess der Visualisierung adaptiert wird. Ein Beispiel für einen *Adaptionsmechanismus* auf der visuellen Repräsentation wird in Abschnitt 4.2 erläutert.

Zeitliche Adaption vs. räumliche Adaption Wahlweise kann die Adaption in einem Schritt erfolgen oder sequenziell über die Zeit eine progressive Bildanzeige realisieren. In Abschnitt 4.5 wird auf die zeitliche Adaption eingegangen.

Lokal vs. global Die *Adaptionsmechanismen* werden danach klassifiziert, ob diese global (auf allen zu visualisierenden Daten) oder lokal (auf einer ausgewählten Teilmenge der Daten) arbeiten. Mit der lokalen Adaption von *Adaptionsmechanismen* befasst sich Abschnitt 4.4.

Unterschiedliche *Adaptionsmechanismen*, welche auf der Prozessebene ansetzen, eine räumliche Adaption vornehmen und alle zu visualisierenden Daten

in die Adaption mit einbeziehen, werden in Abschnitt 4.3 behandelt.
Das in Abbildung 4.1 dargestellte Schema fasst noch einmal die Klassifikation zusammen. Hier lassen sich sowohl aus der Literatur bekannte als auch bereits existierende und neue *Adaptionsmechanismen* einordnen, so dass es eine gute Grundlage für systematische Untersuchungen bietet. Im Folgenden soll für jede hier aufgeführte Klasse eine konkrete Technik vorgestellt werden.

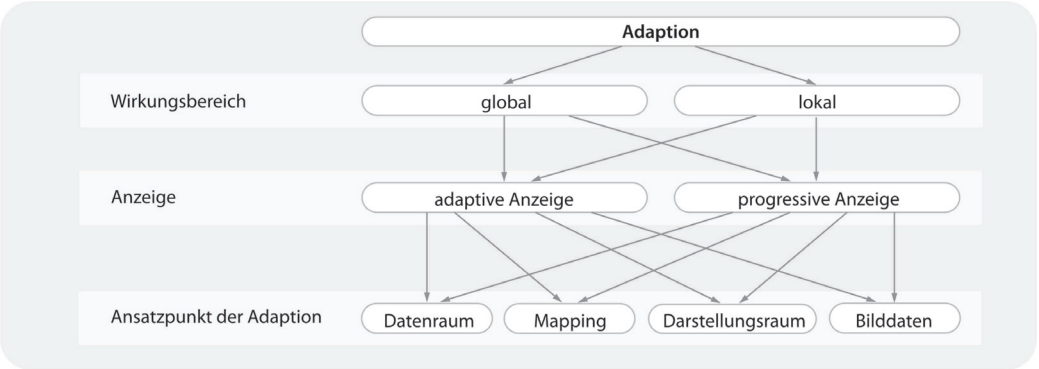

Abbildung 4.1: Schema zur Einordnung unterschiedlicher Adaptionsmechanismen. Beispielsweise kann es sich bei einem Adaptionsmechanismus um eine globale, adaptive Anzeige im Darstellungsraum handeln.

4.2 Adaption auf der Bildebene

Der Wechsel auf ein neues Ausgabegerät kann mit einer neuen Auflösung des Bildes verbunden sein. Dabei handelt es sich um ein bekanntes Problem, zu dessen Lösung viele Möglichkeiten wie *Panning und Zooming*, *Skalierung* oder auch *Fokus und Kontext-Techniken* existieren, die sich aber nicht unmittelbar auf die Zielstellung dieser Arbeit übertragen lassen.
Ziel dieser Arbeit ist die gleichzeitige Ausgabe von Informationsdarstellungen auf mehreren Displays, deren grundlegende Eigenschaften kommuniziert werden müssen. Zudem muss die "mental map" der Benutzer berücksichtigt werden. Es sollen deshalb keine Bildausschnitte (*Panning und Zooming*) verwendet werden, da dabei Informationen verloren gehen können. Auch *Fokus und Kontext* bietet sich nicht an, da man nicht davon ausgehen kann, dass ein Bereich von Interesse gegeben ist. Bei allgemeinen Skalierungen können Verzerrungen auftreten und so unter Umständen zusammenhängende Informationen verfälscht werden.
Deshalb wurde im Rahmen dieser Dissertation der Ansatz von Avidan und

Shamir [AS07] zur *inhaltsbasierten Skalierung*, die bisher nur auf Fotos angewendet wird, als Grundlage verwendet. Dieses Konzept wurde auf die Informationsvisualisierung übertragen und in einer Diplomarbeit umgesetzt (vgl. [Ros09a]).
Bei Avidan und Shamir erfolgt die Skalierung der einzelnen Bildbereiche in Abhängigkeit davon, wie wichtig der entsprechende Bereich ist. Das heißt, unwichtige Informationen werden aus dem Bild entfernt und schaffen so Platz für wichtige Informationen, bei denen deshalb keine Skalierung mehr erforderlich ist. Es handelt sich demzufolge um eine *inhaltsbasierte Skalierung*. Abbildung 4.2 veranschaulicht an einem Beispiel das grundlegende Vorgehen von Avidan und Shamir.
Die Autoren nehmen im ersten Schritt eine Analyse der Bilddaten mit Hilfe

Abbildung 4.2: Inhaltsbasierte Skalierung von fotorealistischen Bilddaten, aus [AS07]

einer Energiefunktion vor. Die Funktion liefert Aussagen über die Wichtigkeit jedes Bildpunktes. Anschließend werden so genannte *Seams*[1] innerhalb

[1]Ein *Seam* ist ein zusammenhängender Pfad von niederenergetischen Bildpunkten. Vertikale *Seams* beginnen am oberen Rand des Bildes und enden am unteren Bildrand. Pro

des Bildes berechnet. Durch das sukzessive Entfernen von *Seams* können die Ausmaße des Bildes verändert werden. Die für die Aussage des Bildes wichtigen Bildbereiche bleiben dabei erhalten.
Bisher wurde dieser *Adaptionsmechanismus* nur auf Fotos angewendet. Für die Informationsdarstellung in einem *Smart-Meeting-Room* muss aber auch untersucht werden, ob und wie sich dieser Ansatz auf andere visuelle Repräsentationen übertragen lässt. Als Ergebnis der Untersuchungen konnte festgestellt werden, dass die *inhaltsbasierte Skalierung* zwar nicht auf alle Informationsdarstellungen sinnvoll angewendet werden kann, aber doch für viele Techniken eine gute Lösung darstellt. Prinzipiell darf es sich bei den zu adaptierenden Bilddaten nicht um raumfüllende visuelle Repräsentationen handeln. Das heißt, es muss zwischen den einzelnen Bildbereichen, die wichtige Informationen repräsentieren, Bereiche geben, die weniger wichtige Informationen repräsentieren und damit ausgeblendet werden können.
Betrachtet man den Einfluss der *inhaltsbasierten Skalierung* auf die einzelnen *visuellen Variablen* (vgl. [Ber82]), so gilt:
Werden Informationen auf die visuellen Variablen Größe, Farbe, Helligkeit, Textur, Orientierung und Form abgebildet, so kann eine *inhaltsbasierte Skalierung* angewendet werden unter der Voraussetzung, dass die Energiefunktion so gewählt wird, dass keine *Seams* durch *grafische Primitive* führen, welche die *visuellen Variablen* repräsentieren.
Werden hingegen Informationen auf die Position abgebildet, wie es bei einer Scatterplot-Darstellung der Fall ist, so darf die *inhaltsbasierte Skalierung* nicht angewendet werden. Durch das Entfernen von Bildzeilen und -spalten kommt es zu Verzerrungen, so dass die Positionen der Punkte im Bild falsch interpretiert werden können. Siehe hierzu Abbildung 4.3.
Die größte Herausforderung besteht nun darin, eine Energiefunktion bereitzustellen, welche alle wichtigen *grafischen Primitive* davor schützt, dass ein *Seam* durch sie hindurchführt. Dabei ist es unerheblich, ob diese *grafischen Primitive* Texturen, Formen oder andere *visuelle Variablen* repräsentieren. Solange keine Informationen in Positionen kodiert sind, lassen sich entsprechende Energiefunktionen aufstellen und die *inhaltsbasierte Skalierung* anwenden.

Exemplarisch wurde deshalb in der bearbeitenden Diplomarbeit (vgl. [Ros09a]) die *inhaltsbasierte Skalierung* auf Schaltpläne angewendet. Schaltpläne enthalten neben den Bauelementen noch die Verbindungen zwischen den Bauelementen und weisen damit eine Analogie zu Graphdarstellungen auf, wie

Zeile belegt der *Seam* genau ein Pixel. Horizontale *Seams* beginnen am linken Rand des Bildes und enden am rechten Bildrand. Pro Spalte belegt der *Seam* genau ein Pixel.

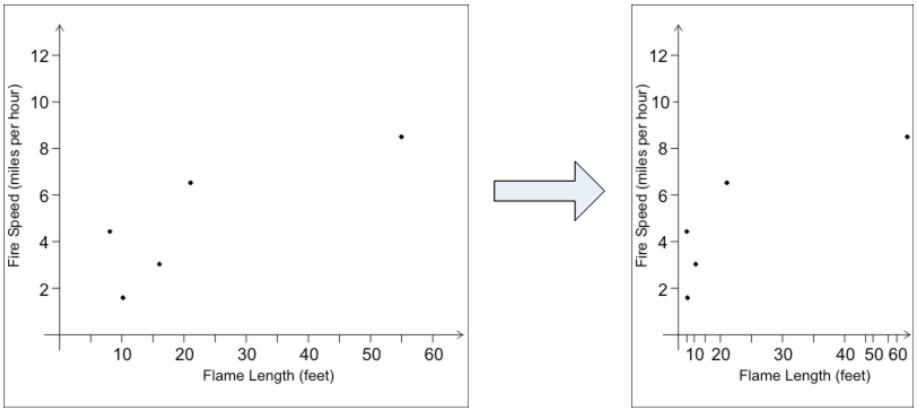

Abbildung 4.3: Anwendung der inhaltsbasierten Skalierung auf einen Scatterplot, aus [Ros09a]

sie in der Informationsvisualisierung üblich sind. Als Energiefunktion wurde die Entropie ausgewählt.
Bei der Berechnung von vertikalen und horizontalen *Seams* werden getrennte Energiebilder verwendet. In den Energiebildern für die vertikalen *Seams* wurde die Energie von vertikalen Linien erhöht, wodurch verhindert wird, dass *Seams* vertikale Linien schneiden. Bei den Energiebildern, welche zur Berechnung der horizontalen *Seams* verwendet werden, wurde analog zu den horizontalen Linien verfahren. Dadurch kann das Auftreten von Artefakten beim Schneiden von Linien durch die *Seams* vermieden werden. Ein anschauliches Beispiel, bei dem Artefakte[2] beim Entfernen von *Seams* auftreten, ist in Abbildung 4.4 dargestellt. Während bei der herkömmlichen Skalierung die Schrift und einzelne Bauteile nur schwer oder gar nicht zu erkennen sind, sind diese bei einer visuellen Repräsentation, die durch eine *inhaltsbasierten Skalierung* entstanden ist, immer noch zu kennen.
Die Resultate der *inhaltsbasierten Skalierung* und der herkömmlichen Skalierung für Schaltpläne sind in Abbildung 4.5 gegenübergestellt. Deutlich sind die hiermit erreichbaren Verbesserungen zu erkennen.

4.3 Adaption auf der Prozessebene

Bei einer Adaption auf der Prozessebene erfolgt ein Eingriff in die Visualisierungspipeline. Dabei muss der Standardablauf, welche durch die Operatoren realisiert wird (vgl. 2.2.1), durch entsprechende Operatoren ergänzt werden.

[2] Artefakte in diesem Kontext sind Treppeneffekte und nicht mehr durchgängige Linien.

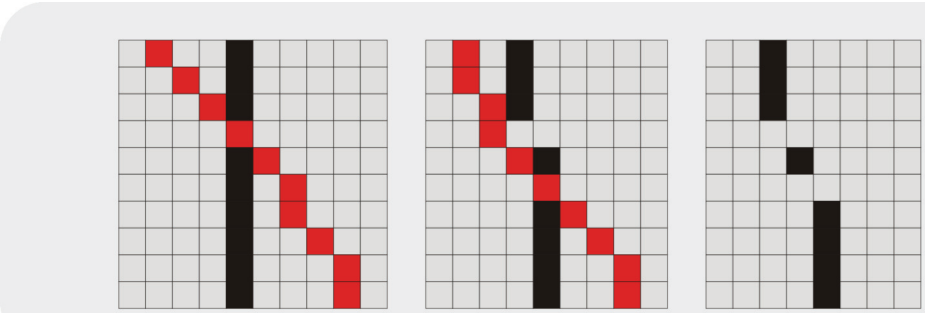

Abbildung 4.4: Veranschaulichung der Bildung von Artefakten beim Schneiden von vertikalen Linien durch vertikale *Seams*. Dabei wurden die Linie schwarz dargestellt und die beiden *Seams* in rot.

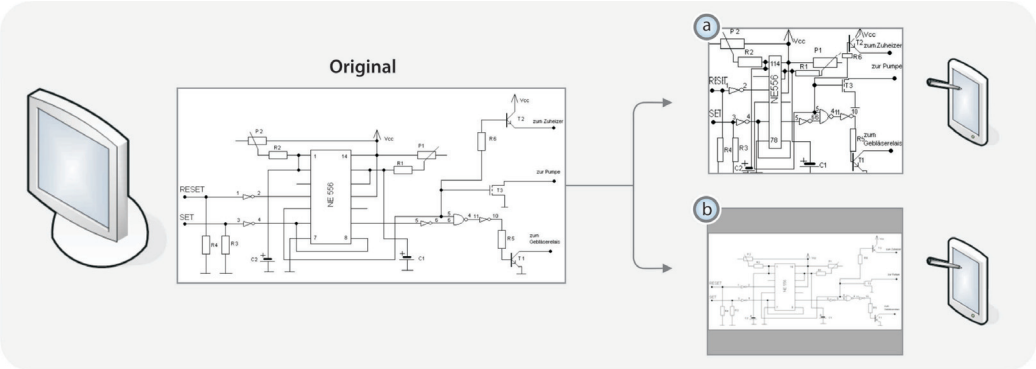

Abbildung 4.5: Vergleich der Resultate von a) inhaltsbasierter Skalierung und b) herkömmlicher Skalierung, aus [Ros09a]

Die Herausforderung dabei: Welche Operatoren können verwendet werden, um die *Adaptionsmechanismen* zu realisieren? Prinzipiell können diese im *Datenraum*, *Darstellungsraum* oder als *Mapping-Operatoren* definiert sein. Für jeden Bereich soll ein Beispiel angegeben werden.
Eine weitere Herausforderung ist die Fragestellung, wie die *Adaptionsmechanismen* in die Visualisierungspipeline eingebunden werden. Eine in dieser Dissertation entwickelte Lösung basiert auf einem *Entscheidungsbaum*, dessen Funktionsweise im Abschnitt 4.3.4 näher beschrieben wird.
Im Rahmen der vorliegenden Dissertation wurden hauptsächlich *Adaptionsmechanismen* zur Anzeige auf kleineren Displays entwickelt. Die Fragestellung der Anzeige auf größeren Displays wird in der zweiten Phase des GRK's MuSAMA behandelt[3].

[3]Große Displays werden erst mit dem Eintritt von Prof. Staadt in der zweiten Phase des GRK MuSAMA involviert.

Bei der Adaption einer Informationsvisualisierung auf ein kleineres Display muss entweder die Komplexität der Daten oder die Komplexität der Darstellung reduziert werden. Beide Fälle werden im Folgenden genauer untersucht.

4.3.1 Adaptionsmechanismen im Datenraum

Bei einer Reduzierung der Datenkomplexität können prinzipiell *Datenwerte ausgeblendet* oder *zusammengefasst* werden. Das bedeutet, dass anschließend durch die weitere Visualisierungspipeline nur eine reduzierte Datenmenge verarbeitet werden muss. Daraus ergeben sich einerseits Vorteile hinsichtlich der Verarbeitungsgeschwindigkeit und andererseits Vorteile durch eine daraus resultierende reduzierte Komplexität in der visuellen Repräsentation und somit eines geringeren *Visual Clutter*.

Ausblenden von Datenwerten

Beim Ausblenden von Datenwerten wird die Datenmenge der zu visualisierenden Daten reduziert. Die Auswahl der Datenwerte, die später nicht mehr visualisiert werden, muss systematisch erfolgen.
Es gibt Vorschläge, die Auswahl zufällig vorzunehmen (vgl. [DE02, ED02, ED06b]); dahinter steht der Gedanke, dass dann auch der Fehler, der durch das Entfernen der Datenwerte auftritt, gleichmäßig in der visuellen Repräsentation verteilt wird. Insbesondere verändert sich dabei nicht das Dichteverhältnis in den einzelnen Regionen der visuellen Repräsentation, wenn weiterhin eine ausreichende Anzahl von Datenwerten weiterhin dargestellt wird.

Soll aber nicht die allgemeine Verteilung der Daten aus der visuellen Repräsentation ersichtlich sein, sondern sollen beispielsweise Ausreißer sichtbar gemacht werden, so kann ein zufälliges Entfernen von Datenwerten zu Fehlinterpretationen der visuellen Repräsentation führen. Dieses Beispiel macht deutlich, dass beim Ausblenden von Datenwerten unbedingt die aktuelle Aufgabe berücksichtigt werden muss. Beim Ausblenden von Datenwerten muss demnach eine Aufgaben-basierte Adaption vorgenommen werden. Eine genauere Betrachtung der Aufgaben-basierten Datenreduktion erfolgt im Rahmen der Dissertation von Georg Fuchs[4] (vgl. [Fuc10]). Daher soll an dieser Stelle nicht weiter darauf fokussiert werden.

[4]Georg Fuchs ist ein Kollegiat des GRK MuSAMA.

Zusammenfassen von Datenwerten

Ein möglicher Lösungsansatz, bei dem eine Zusammenfassung der Datenwerte im Datenraum vorgenommen wird, ist eine hierarchische Strukturierung der Daten. Dabei wird folglich [Kre05] eine stufenweise Zusammenfassung inhaltlich ähnlicher Datenwerte vorgenommen (siehe auch Abbildung 4.6). Durch das stufenweise Zusammenfassen von Datenwerten lassen sich einer-

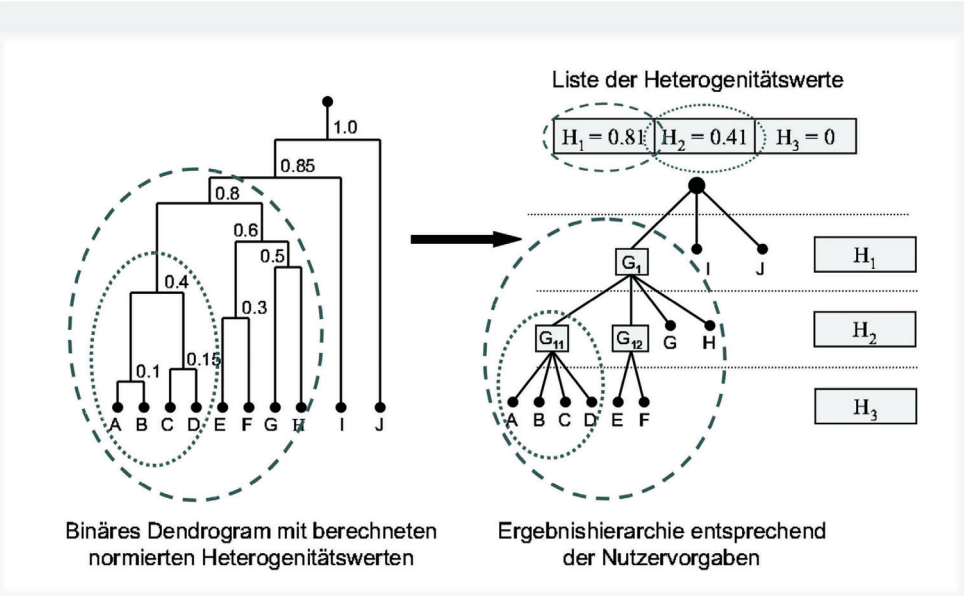

Abbildung 4.6: Konstruktion einer Datenhierarchie (Ergebnishierarchie) mit drei Ebenen aus einem zuvor berechneten Dendrogramm, aus [Kre05]

seits grobe Unterteilungen in nur wenigen Hierarchiestufen berechnen, die für einen ersten Überblick der Informationen ausreichen, andererseits besteht für detaillierte Analysen die Möglichkeit, die grobe Hierarchie zu verfeinern, um kleine Unterklassen und Details in den Daten zu identifizieren.
Kreuseler [Kre05] stellt fest, dass es sinnvoll ist, für die Hierarchisierung der Daten verschiedene Methoden bereitzustellen, um je nach Analysezielstellung ein geeignetes Verfahren auszuwählen. Er schlägt an dieser Stelle modifizierte agglomerative Clusterverfahren wie *Single Linkage*, *Complete Linkange*, *Average Linkage*, *Ward*, *Median*, *Flexibale Strategy* und *Cenroid* (vgl. [BEPW96, KR90] für weitere Details) vor.
Eine hierarchische Strukturierung der Daten hat den Vorteil, dass verschiedene Abstraktionsstufen je nach Charakter der Ausgabegeräte stufenlos auswählbar sind. Sie stellen somit eine adäquate Lösung eines *Adaptionsmechanismus* im Datenraum zur Adaption an unterschiedliche Ausgabegeräte dar und

werden daher auch im Rahmen der vorliegenden Dissertation genutzt (vgl. Kapitel 5).

4.3.2 Adaptionsmechanismen beim Mapping

Das Mapping bietet weitere Möglichkeiten, die visuelle Repräsentation an ein Ausgabegerät anzupassen. Hierzu wurden erste Untersuchungen zur Helligkeitsanpassung durchgeführt. Daneben gibt es weitere Möglichkeiten, die im Folgenden kurz skizziert werden.

Helligkeitsanpassung

Um bessere Resultate beim Einsatz des Binnings (vgl. [NH06]) auf Scatterplots zu erzielen, wurde in Kooperation mit Mike Sips und Georg Fuchs ein Ansatz entwickelt, der die verwendete Farbskala modifiziert.
Anstatt die original von Novotny et al. [NH06] vorgeschlagene lineare Farbskala zu verwenden, wurde bei Bedarf auf eine logarithmische Farbskala gewechselt. Für eine verzerrte Verteilung der Bin-Frequenzen zwischen dichten Zentren von kompakten Clustern und weniger dichten Regionen, in denen sich die einzelnen Cluster überlagern, bietet sich eine nicht lineare Farbskala an. Abbildung 4.7 veranschaulicht den Unterschied zwischen einer linearen und einer logarithmischen Farbskala. Die logarithmische Farbskala stellt mehr Grauwerte im unteren Ende des Frequenzbereiches zur Verfügung, dadurch werden die Bereiche um die Cluster-Zentren noch besser sichtbar.

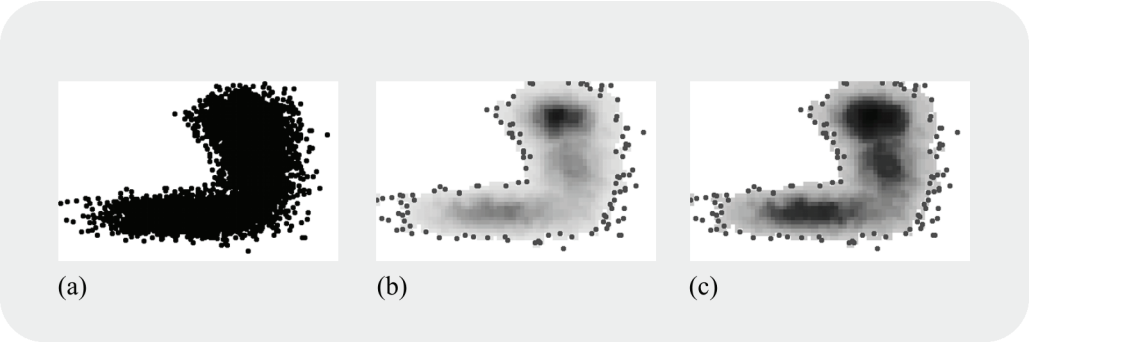

Abbildung 4.7: Dichte-Basiertes-Binning - (a) Standard Scatterplot, (b) Dichte-binning mit Ausreißer Hervorhebung und linearer Farbskala, (c) Mappen der Bin-Frequenzen auf eine logarithmische Farbskala (Es wurde ein synthetischer Datensatz zu Demonstrationszwecken verwendet).

weitere Mechanismen

Neben der hier vorgestellten Anpassung der Farbskalen sind weitere Untersuchungen zu diesem Punkt notwendig, die Gegenstand in der zweiten Phase des GRK-MuSAMA sind.
So besteht die grundlegende Idee des Stipendiaten Axel Radloff darin, zusätzliche visuelle Variablen zu nutzen und mit diesen ein redundantes Mapping vorzunehmen, um so die Effektivität der Informationsdarstellung zu erhöhen und damit auf einer größeren Bandbreite von Ausgabegeräten darzustellen (vgl. [RLS10]).
Auch die Untersuchungen zur aufgabenbasierten Informationsdarstellung thematisieren eine angepasste Farbkodierung (vgl. [Fuc10]).

4.3.3 Adaptionsmechanismen im Darstellungsraum

Während im Datenraum eine Zusammenfassung der Daten vorgenommen werden kann, um die Komplexität der Daten zu reduzieren, werden im Darstellungsraum grafische Primitive zusammengefasst, um so die Komplexität der Darstellung zu reduzieren. Eine bekannte Methode zum Reduzieren der Komplexität im Darstellungsraum ist das Binning (vgl. [NH06]). Das Binning lässt sich allgemein anwenden, produziert gute Ergebnisse und soll darum auch im Rahmen dieser Arbeit genutzt werden.

Funktionsprinzip: Beim Binning von Scatterplots werden die Achsen in b reguläre Intervalle aufgeteilt. Die resultierende Menge von $b \cdot b$ Bins ergibt die sogenannte Bin-Matrix. Diese kann als ein 2D-Histogramm des Darstellungsraums der Datenpunkt-Verteilung angesehen werden.
Während des Renderns wird jeder nicht-leere Bin durch ein Rechteck repräsentiert, bei dem die Frequenz des Bins in die Füllfarbe kodiert wird. In [NH06] wird dieses Prinzip genutzt, um eine *Parallele Koordinatendarstellung* zu verbessern, wobei besonderes Augenmerk darauf gelegt wird, dass Ausreißer erhalten bleiben. In dieser Arbeit wurde das Prinzip auf die Visualisierung klassifizierter Daten mit Scatterplots übertragen.
Um die Clusterstruktur in Scatterplots gut erkennen zu können, muss der Benutzer in der Lage sein, die Cluster-Zentren in der Repräsentation zu unterscheiden. Idealerweise sollte jeder Cluster als eine hochfrequente Region im Plot wahrgenommen werden, die sich visuell von dichten Bereichen anderer Cluster unterscheidet. Hierzu wurde folgende Erweiterung des Binning-Ansatzes vorgenommen (vgl. [FTSS09]):

- Justieren der Bin-Auflösung entlang der Scatterplot-Achsen unter Beachtung der Positionen der Cluster-Zentren

- Optionale Verwendung einer rechteckigen *Fish-Eye-Verzerrung* zur Unterteilung der Bins anhand der gegebenen Cluster-Zentren, um so die visuelle Unterscheidbarkeit der Cluster-Zentren zu verbessern

Um eine gute initiale Binning-Auflösung zu finden, werden anfangs nur die Cluster-Zentren betrachtet. Dabei wird der Bereich des Scatterplots in $b_x \cdot b_y$ Bins partitioniert, abhängig von einer anfänglich gesetzten Bin-Größe. Es wird anschließend überprüft, ob in einem Bin mehr als ein Cluster-Zentrum liegt. In diesem Fall wird die Bin-Auflösung weiter unterteilt, indem b_x bzw. b_y um 1 inkrementiert und anschließend erneut überprüft wird, ob zwei Cluster-Zentren in ein Bin fallen. Schlägt der Test immer noch fehl, wird die Unterteilung weiter vorangetrieben, indem alternierend b_y bzw. b_x inkrementiert werden, bis entweder alle Cluster-Zentren in einem separaten Bin liegen oder bis ein zuvor definierter Schwellwert für die Bin-Größe erreicht wurde.
Bei den angestellten Untersuchungen, die stattfanden, hat sich eine Bin-Größe von 5 x 5 Pixel (Startwert) bis 2 x 2 Pixel (unterer Schwellwert) als ein guter Kompromiss zur Vermeidung von *Visual Clutter* in der visuellen Repräsentation und einer guten Darstellung der Cluster auf einer kleinen Anzeige (vgl. Abbildung 4.7 (a,b)) erwiesen.
Nachdem die Bin-Größe und die resultierende Bin-Frequenz ermittelt wur-

Abbildung 4.8: Anwendung einer rechteckigen *Fish-Eye-Verzerrung* - (a) unverzerrter Plot, (b) Bins, in denen Cluster-Zentren liegen, werden vergrößert. (c) Vergrößerte Regionen werden unterteilt. Durch die verbesserte Abtastrate in (c) kommt zum Vorschein, dass die dichte obere Region in Wirklichkeit aus drei kleinen sehr dichten Regionen besteht. Weiterhin wird die Position einer weiteren dichten Region in dem Cluster mit geringer Dichte (Mitte-rechts) hervorgehoben.

den, wird wahlweise eine rechteckige *Fish-Eye-Verzerrung* über den Bins positioniert, die die Cluster-Zentren enthalten. Die Vergrößerung des verfügbaren Bildbereiches für die Regionen der Cluster-Zentren erlaubt ein weiteres Unterteilen in diesen Regionen. Der Unterteilungsfaktor ist dabei propor-

tional zum Vergrößerungsfaktor. Das bedeutet, eine Vergrößerung der fokussierten Bins um den Faktor zwei impliziert eine zweifachen Unterteilung dieser Bins. Die lokal gesteigerte Bin-Auflösung verbessert die Darstellung von Frequenzabweichungen um die Cluster-Zentren herum. Dadurch können nah beieinander liegende Cluster-Zentren besser unterschieden werden (vgl. 4.8(c)).

4.3.4 Entscheidungsbaum

Im Rahmen dieser Dissertation wurde eine zweistufige *Adaptionsstrategie* zur Geräte-basierten Adaption der visuellen Repräsentation für *Adaptionsmechanismen* auf der Prozessebene erarbeitet (vgl. Abbildung 4.9). Der Entscheidungsbaum[5] steuert, welche *Adaptionsmechanismen* eingebunden werden. Dabei werden folgende Schritte abgearbeitet:

1. Es wird die *Effektivität* der visuellen Repräsentation abgeschätzt. Ist die Informationsvisualisierung nicht effektiv, sind in einem zweiten Schritt Adaptionen erforderlich.

2. Es wird eine geeigneter *Adaptionsmechanismus* gewählt, um die charakteristischen Eigenschaften des Ausgabegerätes zu berücksichtigen. Dabei erfolgt die Auswahl eines geeigneten *Adaptionsmechanismus* entweder aus dem Datenraum, dem Darstellungsraum oder dem Mapping.

Bevor diese *Adaptionsstrategie* angewendet werden kann, müssen eine Reihe von Entscheidungen getroffen werden:
Die Effektivität einer visuellen Repräsentation auf einem anderen Ausgabegerät kann sich verändern, wenn die Repräsentation im Vergleich zum Referenzdisplay entweder auf ein kleineres Display transferiert wurde oder auf ein wesentlich größeres Display.
Im ersten Fall kann *Visual Clutter* auftreten, wenn zu viele Daten auf der kleineren Displayfläche ausgegeben werden. *Visual Clutter* wurde hinreichend untersucht und hat einen wesentlichen Einfluss auf die Effektivität einer visuellen Repräsentation (vgl. auch Abschnitt 2.2.4).
Zusätzlich gibt es erste Untersuchungen dazu, dass ein Skalieren der visuellen Repräsentation auf ein größeres Display ebenfalls zu einer Verringerung der Effektivität führen kann (vgl. [FTSS09, BS04]), da sich die wahrgenommene Verteilung der Daten und dabei insbesondere die Wahrnehmung von lokalen

[5]Der Entscheidungsbaum ist in einer gemeinsamen Kooperation mit Georg Fuchs entstanden.

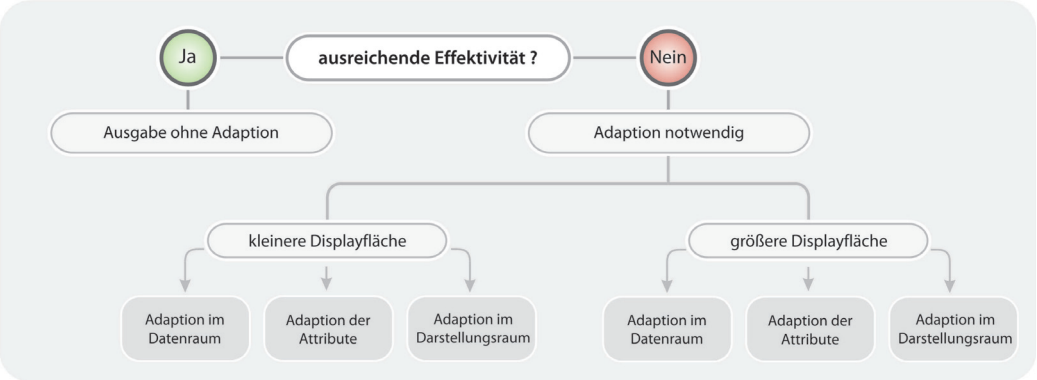

Abbildung 4.9: Schematische Darstellung des Entscheidungsbaum zur Auswahl geeigneter *Adaptionsmechanismen*.

Bereichen mit erhöhter Datendichte verändert. In dieser Arbeit liegt der Fokus auf dem ersten Fall, da sehr große Displays erst mit der zweiten Phase des GRK MuSAMA über die Teilnahme von Prof. Staadt involviert sind.

Die Definition von Metriken und Evaluierungsmethoden, welche in der Lage sind, präzise darüber Auskunft zu geben, wie effektiv eine gegebene Visualisierung die Daten präsentiert, ist immer noch eine ungelöste Forschungsfrage (vgl. [Nor06]).
Tufte [Tuf06] führt Maße ein, um die Qualität einer 2D-Repräsentation von statischen Daten abzuschätzen (siehe auch Abschnitt 2.2.4). Beispiele sind das *Data-to-Ink Ratio* oder die *Data Density*, welche die Größe der visuellen Repräsentation ins Verhältnis setzen zu der Menge an Daten, die angezeigt wird. Andere Ansätze messen das auftretende *Visual Clutter* [BS04] oder die *Consistency* [SNLH09], um zu ermitteln, wie gut eine visuelle Repräsentation die Charakteristika der Daten kommuniziert.
Zudem sind Schwellwerte erforderlich, um die Effektivität einer visuellen Repräsentation zu bestimmen und so eine automatische Adaption zu erlauben. Diese Schwellwerte sind von der konkret genutzten Visualisierungstechnik, vor allem aber von der Wahrnehmungsfähigkeit des Betrachters abhängig und werden deshalb durch Nutzerstudien festgelegt (vgl. z. B. [SNLH09]).
In den vorhergehenden Abschnitten wurden unterschiedliche *Adaptionsmechanismen* vorgestellt, die jeweils in anderen Bereichen der Visualisierungspipeline ansetzen und die exemplarisch für die Umsetzung des Entscheidungsbaums zur Verfügung stehen. Standardmäßig wird eine Adaption im Bildraum vorgenommen, um nicht die Daten zu verändern, sondern nur ihre Repräsentation (vgl. 5).

4.4 Lokale Adaption

Bisher wurde die Adaption bezogen auf die gesamte visuelle Repräsentation durchgeführt. Aber nicht immer ist dies notwendig und angebracht. Eine Alternative wäre, Anpassungen nur in lokal begrenzten Bereichen vorzunehmen. Hierfür gibt es zwei Argumente:

- Die "mental map" bleibt erhalten. Es wird für die Benutzer einfacher, die adaptierten Bereiche einzuordnen[6].

- Die Berechnung vereinfacht sich. Dieser Vorteil wiegt besonders stark im Hinblick auf leistungsschwache Ausgabegeräte wie PDAs oder Smart-Phones. Da Updates und Veränderungen sich nur auf kleine lokale Bereiche beziehen, können notwendige Berechnungen schnell durchgeführt werden.

Ein Konzept, mit dem sich lokale Anpassungen realisieren lassen, sind Linsen. Diese werden interaktiv durch den Benutzer positioniert und verändern die visuelle Repräsentation in einem definierten lokalen Bereich (vgl. [BSP+93, BSP97]).
Linsen arbeiten typischerweise im Darstellungsraum und verzerren lokale Bildbereiche. Daneben führt Bier [BSP+93] mit den *magic lenses* auch Linsen im Datenraum ein. Griethe et al. [GFS05] verallgemeinern dieses Konzept und definieren Linsen als Operatoren auf dem Data-State-Reference-Model (vgl. 2.2.1, [CR98, Chi00]). Einige Beispiele sollen die vielfältigen Optionen, die sich aus diesem Konzept „Linsen als Operatoren" ergeben, demonstrieren:

Linsen als Operatoren auf den einzelnen Stufen

- *Linsen als Operatoren auf den Daten* zur Reduzierung der zu visualisierenden Daten zur Vermeidung von *Visual Clutter* (Sampling vgl. [ED06b, ED06a, ED07, EBD05] siehe auch Abbildung 4.10), zum Entfernen von Fehlern oder zum Interpolieren von Werten im Bereich der Linse.

- *Linsen als Operatoren auf den analytischen Abstraktionen* z. B. zur Anzeige von zusätzlichen Informationen wie Clustereigenschaften der Daten im Bereich der Linse.

[6] Diese Behauptung stützt sich auf die Analogie zu den bekannten *Fokus und Kontext* Techniken, bei denen ebenfalls der Kontextbereich verwendet wird, um den Bereich auf dem der Fokus liegt, besser einordnen zu können (vgl. [Kea99, CMS99]).

- *Linsen als Operatoren auf den visuellen Abstraktionen* zur Anpassung des Detail-Grades (LoD, vgl. [FHRS08, Kea98]) oder zur Generalisierung/Vereinfachung/Zusammenfassung von *visuellen Primitiven* im Bereich der Linse (vgl. [BS05a]).
- *Linsen als Operatoren auf den Bilddaten*, zur Anpassung von Sättigung und Farbe oder zur Realisierung von Verzerrungen der Bilddaten (vgl. [Kea99]).

Abbildung 4.10: Linien innerhalb einer Linse mit 10%er Samplingrate, aus [ED06b]

Linsen zur Steuerung

- *Linsen als Filtering-Operatoren* haben die Möglichkeit, zusätzliche Werte zu den Daten zu berechnen wie beispielsweise Kennwerte zu den Knoten in einem großen Graphen. Stattdessen können auch Informationen im Bereich der Linse ausgeblendet werden (vgl. [LH94]). Oder es können im Linsenbereich nicht inzidente Kanten von Knoten entfernt werden, um so *Visual Clutter* zu reduzieren (vgl. [TAvHS06]).
- *Linsen als Mapping-Operatoren* greifen in das Mapping ein. Sie können dazu verwendet werden, um zusätzliche Aspekte der gegebenen Daten im Linsenbereich zu visualisieren. (vgl. [BSP+93, Kea98, BSP97, LBC04], siehe auch Abbildung 4.11).
- *Linsen als Rendering-Operatoren* dienen zur Realisierung von grafischen Linsen, wie z. B. Verzerrungslinsen. Die kartografische Lupe von Rase ist ein konkretes Beispiel für diese Linsen (vgl. [Ras97, LA94]).

Damit ist eine große Vielfalt lokaler Anpassungen möglich, insbesondere auch durch die Kombination von Linsen. Darum sollen Linsen auch im Rahmen dieser Arbeit genutzt werden. Dabei sind Linsen wie folgt definiert:

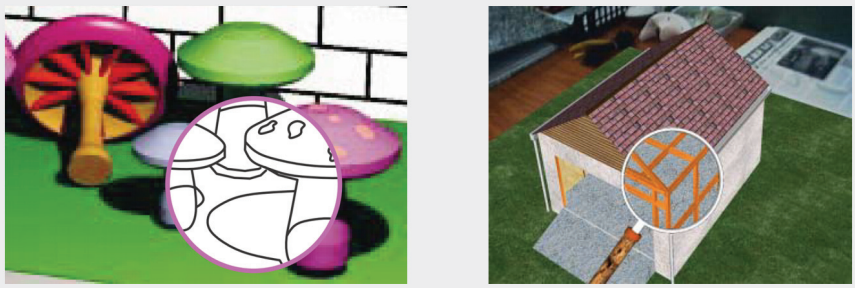

Abbildung 4.11: Unterschiedliches Mapping im Bereich der Linse, aus [BSP97, LBC04]

4.4.1 Definition von Linsen

Linsen sind allgemein definiert durch ihre *Position*, *Form* und ihre *Linsenfunktion*.

Position Die Position der Linse ist spezifiziert durch einen Punkt in der visuellen Repräsentation. Die Angabe erfolgt durch einen 2D- oder 3D-Punkt.

Form Die Form der Linse spezifiziert den Bereich, der von der Linsenfunktion beeinflusst wird. Für gewöhnlich ist die Form der Linse durch einfache geometrische Formen wie Quadrate oder Kreise bestimmt bzw. im 3D-Fall durch Würfel oder Kugeln. Aber es sind auch komplexere Formen möglich, wie beispielsweise Polygone, die zudem Löcher enthalten können.

Linsenfunktion Die Linsenfunktion wird als ein Operator oder eine Zusammenfassung von Operatoren des Data-State-Reference-Model aufgefasst. Sie kann wie jeder andere Operator auch, auf den Stufen des DSRM Datenwerte hinzufügen, entfernen oder diese modifizieren. Durch die Linsenfunktion ist demzufolge spezifiziert, wie die Datenwerte im Bereich der Linse angezeigt werden.

Dieses Konzept soll im Folgenden systematisiert werden, so dass sich beliebige Anpassungen sowohl auf Daten- als auch auf Darstellungsebene vornehmen lassen. Insbesondere wird mit dem neuen Ansatz der „intelligenten Linsen“ ein intelligenter Steuermechanismuss für diese Anpassungen vorgeschlagen. Bei der Realisierung der lokalen Adaption mit Hilfe der soeben eingeführten Linsen sind zwei Probleme zu lösen:

1. Werden Linsen auf allen Stufen des DSRM eingesetzt, muss die Region, welche eine Linse beeinflusst, auf allen Stufen definiert sein.

2. Neu ist die Kombination von mehreren Linsen, die als Operatoren auf dem DSRM definiert sind. Dadurch ergeben sich auch hier neue Herausforderungen.

4.4.2 Definition der Linsenregion auf allen Stufen des Data-State-Reference-Model

Die Linsenregion ist der Bereich der Daten, der von der Linsenfunktion beeinflusst wird und leitet sich direkt aus der Position und Form der Linse ab. Um eine große Flexibilität zu gewährleisten, wird in dieser Arbeit die *Linsenregion* auf jeder Stufe des DSRM als eine Schnittmenge von Halbräumen spezifiziert.
Allgemein betrachtet ist ein Halbraum der Bereich eines n-dimensionalen Raums, welcher auf der einen Seite einer $(n-1)$-dimensionalen Hyperebene liegt, die den Raum in zwei Teile zerlegt. Eine beliebige Region im n-dimensionalen Raum kann durch die Schnittmenge von geeigneten Halbräumen beschrieben werden. Halbräume lassen sich sowohl im Daten- als auch im Darstellungsraum definieren und einfach ineinander überführen.

4.4.3 Kombination von Linsen

Bisher wurde nur die Definition einer einzelnen Linse betrachtet, theoretisch können aber auch mehrere Linsen auf unterschiedlichen Stufen definiert werden. Um die Operatoren der Linsen in die vorliegende Visualisierungspipeline zu integrieren, gibt es zwei Varianten:

1. Es werden zwei separate Visualisierungspipelines verwendet. Eine, um die eigentliche visuelle Repräsentation zu erstellen, die andere um die visuelle Ausgabe im Linsenbereich zu berechnen. Im Anschluss werden beide visuellen Repräsentationen im Bildraum vereinigt (siehe Abbildung 4.12).

2. Es wird nur eine Visualisierungspipeline verwendet. Dabei manipulieren die Operatoren der Linse die Daten der Pipeline direkt. Sie haben somit einen unmittelbaren Effekt, und es kann zu Wechselwirkungen zwischen mehreren Linsen kommen (siehe Abbildung 4.13).

Während die erste Variante nicht zum Kombinieren von mehreren Linsen auf allen Stufen des DSRM verwendet werden kann, da die berechneten visuellen Repräsentationen erst auf der Stufe der Bilddaten zusammengeführt werden, ist dies bei der zweiten genannten Variante möglich. Aus diesem Grund wird

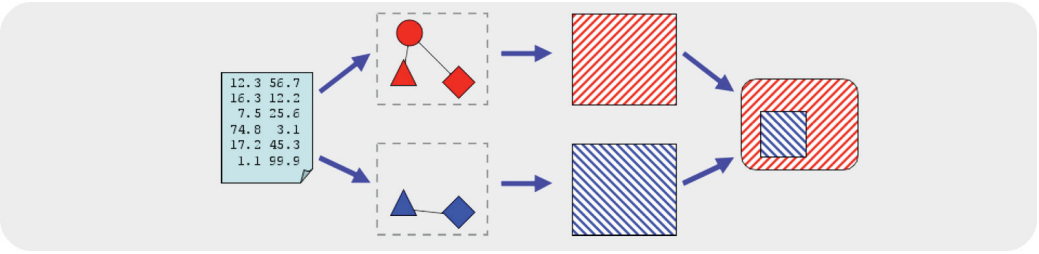

Abbildung 4.12: Integration einer Linse mit zwei getrennten Visualisierungspipelines, aus [EBD05].

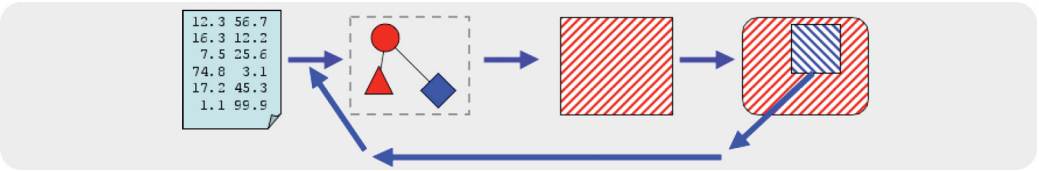

Abbildung 4.13: Integration einer Linsen mit nur einer Visualisierungspipeline, aus [EBD05].

in dieser Arbeit die zweite der beiden Varianten verwendet. Dabei kann es zu Wechselwirkungen der Linsen kommen, wenn sich deren Linsenregionen überlappen. In diesem Fall müssen die Linsen kombiniert werden. Hierbei sind zwei Schritte erforderlich:

1. Es sind die entsprechenden Schnittmengen der Linsenregionen zu bestimmen. Da die Linsenregionen als Schnittmenge von Halbräumen definiert sind, ist eine Verschneidung einfach umzusetzen. Am Beispiel von zwei Linsen sind jeweils die Regionen zu bestimmen, die nur von einer der beiden Linsen beeinflusst werden und die Region, in der sich beide Linsen überlappen und somit auch beide Linsen einen Einfluss haben.

2. Die zugehörigen Linsenoperatoren müssen gleichzeitig auf der sich überlappenden Region ausgeführt werden. Dabei sind unterschiedliche Ansätze vorstellbar:

   Auswahl durch Prioritäten Nur eine Linsenfunktion wird auf die sich überlappenden Regionen angewendet. Welche Linsenfunktion dies ist, wird durch Prioritäten bestimmt.

   Verschmelzung der Linsenfunktionen In diesem Fall wird eine neue Linsenfunktion für den sich überlappenden Bereich spezifiziert. Die Basis dieser Funktion bilden die Funktionen der zu kombinierenden Linsen. Dafür ist eine passende und konsistente Beschrei-

bung der Linsenfunktion erforderlich wie z. B. eine Beschreibung mit Hilfe der Prädikaten-Logik.

Kombinieren der Ausgaben der Linsenfunktionen Die Linsenfunktionen werden unabhängig voneinander auf den überlappenden Bereich angewendet. Anschließend werden die Veränderungen, welche die Linsenfunktionen an den Datenwerten vornehmen, umgesetzt.

Da Prioritäten zur Auswahl einer Linsenfunktion zum Überschreiben der Linsen mit niedriger Priorität führen und nicht immer eine Beschreibung der Linsenfunktion unter Verwendung von Prädikaten-Logik möglich ist, wurde in dem hier vorliegenden Konzept das *Kombinieren der Ausgabe der Linsenfunktionen* weiter untersucht.
Die Kombination von zwei oder mehr Ausgabemengen von Linsen bringt Probleme mit sich. Beim Kombinieren können drei unterschiedliche Konflikte auftreten:

1. **Schreibkonflikt** Dabei überschreiben die Linsen dieselben Datenwerte einer Stufe des DSRM.

2. **Abhängigkeitskonflikt** Eine Linse $L1$ verwendet die Datenwerte, die von einer anderen Linse $L2$, verändert werden. In diesem Fall ist die Linse $L1$ von der Linse $L2$ abhängig.

3. **Lesekonflikt** Existieren zwei Linsen, zwischen denen ein *Schreibkonflikt* auftritt und weiter eine dritte Linse, die eben auf diesen Datenwerten arbeitet, so tritt ein *Lesekonflikt* auf. Für die dritte Linse muss entschieden werden, auf welchen der beiden Fassungen der Daten die Linse arbeitet.

Unter der Annahme, dass die Visualisierungspipeline selbst konfliktfrei ist, kann eine einzelne Linse keine Konflikte verursachen, da die Operatoren der Linsen per Definition immer eine höhere Priorität haben als die Operatoren in der gegebenen Visualisierungspipeline.
Weiterhin können *Lesekonflikte* nur dann auftreten, wenn nicht aufgelöste *Schreibkonflikte* existieren. Wird der auslösende *Schreibkonflikt* aufgelöst, so löst sich der betreffende *Lesekonflikt* ebenfalls auf. *Abhängigkeitskonflikte* lassen sich ebenfalls automatisch auflösen, indem die Bearbeitungsreihenfolge der Operatoren entsprechend umgestellt wird. Ausschließlich *Schreibkonflikte* lassen sich nicht automatisch auflösen. Diese treten aber nur bei einem Bruchteil der zu kombinierenden Linsen auf. Dennoch ist eine Strategie erforderlich, um *Schreibkonflikte* auflösen zu können. Dabei kann entweder direkt

der Benutzer konsultiert werden, oder die einzelnen Linsenfunktionen werden mit Prioritäten versehen, welche ebenfalls von den Benutzern in Skripten der Linsen hinterlegt werden. Die Prioritäten werden ausschließlich verwendet, um auftretende *Schreibkonflikte* zwischen den Linsen aufzulösen.
Bevor Konflikte aufgelöst werden können, ist es erforderlich, sie zu erkennen. Zu diesem Zweck wird eine formale Beschreibung der Resultate der Linsenfunktionen sowie formale Tests zum Erkennen der unterschiedlichen Konflikte eingeführt. Diese können im Anhang A.2 nachgelesen werden.

4.4.4 Intelligente Linsen in Smart-Meeting-Rooms

Üblicherweise spezifiziert der Benutzer entsprechend seiner Interessen die Parameter einer Linse interaktiv. Neu an dem im Rahmen dieser Dissertation entwickelten Konzept sind die *intelligenten Linsen*, die als austauschbare Operatoren auf dem Data-State-Reference-Model definiert sind (vgl. [CR98, Chi00]).
Das Konzept der *intelligenten Linsen* sieht vor, dass viele dieser Parameter soweit wie möglich automatisch gesetzt werden können. Auch im Szenario eines *Smart-Meeting-Rooms* sollen die Parameter der Linse entsprechend der vorliegenden Rahmenbedingungen automatisch angepasst werden. Für eine automatische Anpassung ist es notwendig, Kriterien aufzustellen, nach denen die Anpassung erfolgen kann. Hierfür gibt es prinzipiell drei Möglichkeiten:

1. **Benutzer-bezogen** Im Allgemeinen wird dieser Aspekt durch ein Benutzerprofil spezifiziert, welches Informationen wie das spezifische Interesse eines Nutzers oder Vorlieben, wie die Form der Linse umfasst. Auch im Umfeld eines *Smart-Meeting-Rooms* können automatische Anpassungen bezogen auf ein Nutzerprofil vorgenommen werden[7]. So lässt sich beispielsweise die Auswahl der verwendeten Linsenfunktionen steuern.

2. **Daten-bezogen** Hierbei wird die Linse unter Berücksichtigung spezifischer Eigenschaften der Daten adaptiert. Dazu werden z. B. Ähnlichkeiten von Datenwerten verwendet, um die Linsenregion um weitere Datenwerte zu erweitern. Eine weitere Möglichkeit ist die automatische Positionierung über Extremwerten oder bei einem vorliegendem Clustering der Daten über den einzelnen Clusterrepräsentanten.

[7] In der zweiten Phase des GRK's ist die Entwicklung eines allgemeinen Nutzermodells ein Ziel, so dass Wechselwirkungen zwischen Intensionen, Aktionen und der Informationsanzeige ausgenutzt werden können.

3. **Geräte-bezogen** Bei Veränderungen der Displayfläche lassen sich automatisch Linsen positionieren, um interessante Bereiche in der visuellen Repräsentation höher aufzulösen. Es können auch Metriken verwendet werden, um lokale Bereiche mit erhöhter Dichte in der visuellen Repräsentation zu identifizieren (vgl. [BS04]) und dann über diesen eine Vergrößerungslinse zu positionieren.

Es bleibt die Frage, wie die automatische Adaption der Linsenparameter realisiert wird. In der *HCI*[8] werden für Adaptionen der Benutzerschnittstelle komplexe Modelle verwendet(vgl. [BDF+06, RWF06]). In dem hier vorliegenden Konzept soll ein einfacherer Skript-basierter Ansatz verwendet werden, um die für die Parametrisierung der Linsen relevanten Aspekte zu beschreiben. Das Skript enthält derzeit Nutzer-bezogene Informationen, kann aber beliebig erweitert werden, um auch Daten- oder Geräte-bezogene Informationen zu berücksichtigen:

- die bevorzugte Form der Linse
- den *Bereich von Interesse*
- die Linsenfunktion

Ein *Bereich von Interesse* wird zunächst Nutzer-bezogen festgelegt, kann aber Daten-bezogen ausgewertet werden. Ein Beispiel ist das Erkennen von Maximalwerten und die automatische Positionierung der Linse über Bereichen, in denen lokale Maxima auftreten. Der Bereich entspricht damit einer Menge von Datenwerten, welche in der Linsenregion liegen. Die Linsenregion lässt sich durch das Konzept der Halbräume beschreiben. Um diese zu spezifizieren, werden geeignete Filterbedingungen verwendet.
In Abbildung 4.14 sind Linsen dargestellt, die sich automatisch über den Gebieten positionieren, in denen eine Grippewelle ihr Maximum erreicht. Die betreffenden Gebiete werden durch eine Linse visuell hervorgehoben, und durch eine weitere Linse wird der Monat in der die stärkste Grippewelle auftrat, in die Textur kodiert.
Eine weiter Spielart ist in Abbildung 4.16 dargestellt.
Auch in Abbildung 4.17 wurde ein weiteres Beispiel einer Linse angegeben.
An dieser Stelle wurden die Skripte in einem XML-Dialekt spezifiziert. Zur Beschreibung der Filterbedingungen wurde eine Notation verwendet, die durch die *OGC-Filter-Spezifikation* inspiriert wurde (vgl. [Ope05]). Die Beschreibung erlaubt es, komplexe geschachtelte Bedingungen auszudrücken inklusive räumlichen, arithmetischen und logischen Einschränkungen. Das Beispiel

[8] Human Computer Interaction

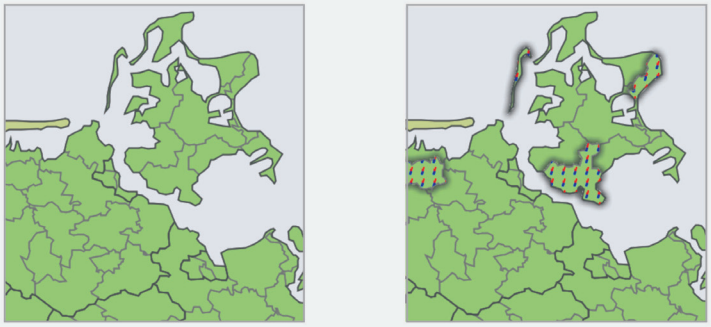

Abbildung 4.14: Darstellung einer kombinierten Linse, die sich automatisch über den Gebieten eines Landes positioniert, die von einer aktuellen Grippewelle am stärksten betroffen sind. Auf der linken Seite sind die Gebiete ohne Linse als Referenz dargestellt.

eines Skriptes einer Linse ist in Abbildung 4.15 dargestellt. Die Konzepte zu den intelligenten Linsen wurden auf der Smart Graphics 2008 [TFS08a] veröffentlicht.
Ein Vorteil der intelligenten Linsen ist, dass mehrere der Linsen auf eine

```
...
<shape type="fixed">
    <circle>
        <center>0 0</center>
        <radius>50</radius>
    </circle>
</shape>
<!-- data-driven position: lens position updated internally by operator class -->
<position type="data-driven"/>

<operator class="lvis.smartLenses.MagicEyeLens">
    <paramDefaults>
        <magnify>2.0</magnify>
        <maxdist>50</maxdist>
    </paramDefaults>
</operator>
<operator class="lvis.smartLenses.RiverIcon"/>
...
```

```
...
<shape type="data-driven"/>
<position type="user-driven"/>
<operator class="lvis.smartLenses.TextureLens">
    <Filter>
        <And>
        <PropertyIsEqualTo>
            <PropertyName>krankheit_nr</PropertyName>
            <Literal>5</Literal>
        </PropertyIsEqualTo>
        <PropertyIsBetween>
            <PropertyName>datum</PropertyName>
            <LowerBoundary>2000-01-01</LowerBoundary>
            <UpperBoundary>2000-12-31</UpperBoundary>
        </PropertyIsBetween>
        </And>
    </Filter>
</operator>
...
```

Abbildung 4.15: XML-Skript einer intelligenten Linse.

visuelle Repräsentation angewendet werden können. Jedem Benutzer kann demzufolge seine eigene *intelligente Linse* zur Verfügung gestellt werden, die sich je nach Vorlieben und Interessen automatisch über der visuellen Repräsentation positioniert und diese anpasst.
Ein weiterer Vorteil ist die leichte Austauschbarkeit der Linsen. Diese können je nach Bedarf jederzeit in eine bestehende visuelle Repräsentation eingefügt oder wieder aus dieser entfernt werden, beispielsweise in einer Diskussion, in der ein neuer Aspekt in der visuellen Repräsentation durch eine Linse verifiziert werden soll.
Dabei bedient jeder Operator eine definierte Schnittstelle im DSRM und lässt

sich somit beliebig austauschen. Zudem lassen sich mehrere Linsen kombinieren, und aus Linsen mit vergleichsweise einfacher Funktionalität können Linsen mit weitaus komplexeren Linsenfunktionen *On-the-Fly* erstellt werden.
Linsen stellen somit ein sehr mächtiges Werkzeug zur Adaption von visuellen Repräsentationen in *Smart-Meeting-Rooms* dar.

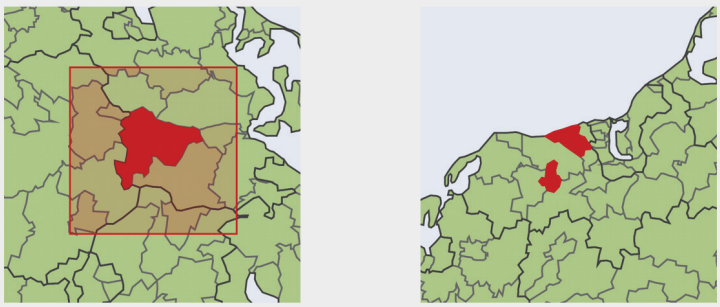

Abbildung 4.16: Eine „Maximum Linse“, welche die Füllfarbe eines Gebietes einer Karte entsprechend der aufgetretenen Fallzahlen modifiziert. Die Form der Linse ist entweder benutzerdefiniert (links) oder Daten-gesteuert (rechts) und wird automatisch über dem Gebiet mit der maximal auftretenden Fallzahl positioniert.

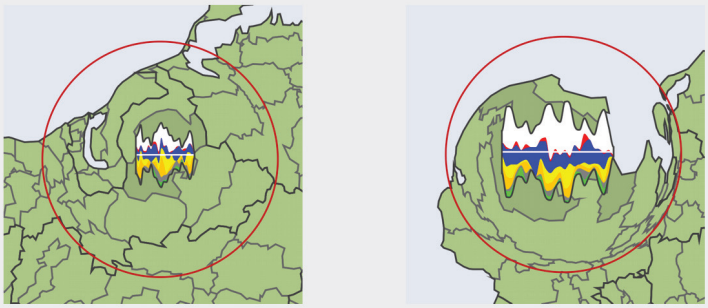

Abbildung 4.17: Dargestellt ist eine Linse, die eine Themenfluss-Ikone [HHN00] in die Kartendarstellung für genauere Analysen einbettet. Diese wird mit einer Fish-Eye Verzerrungslinse kombiniert, bei der der Vergrößerungsfaktor Nutzer-gesteuert spezifiziert wird. Beide Linsen werden automatisch über dem Wohngebiet des Benutzers positioniert.

4.5 Progressive Informationsdarstellung

Bisher wurde von einer adaptiven Informationsdarstellung ausgegangen, das heißt, die visuelle Repräsentation wird *global* oder *lokal* angepasst und dementsprechend auf verschiedenen Ausgabegeräten angezeigt. In diesem Abschnitt soll das neue Konzept der *progressiven Informationsdarstellung* eingeführt werden.
Der Ansatz der *Progression* ist ein anerkanntes Prinzip in der Computergrafik, um Engpässe der Rechenleistung oder der Bandbreite zu kompensieren (vgl. [RS07, SVH08, RS09]).
Besonders verbreitet ist der Ansatz der *progressiven Bilddarstellung*. Die Grundidee der *progressiven Bilddarstellung*, wie sie auch im WWW (World Wide Web) genutzt wird, besteht darin, die Kodierung der Bilddaten so zu wählen, dass die Dekodierung eines abgeschnittenen Datenstroms der kodierten Daten das Bild mit weniger Details wiederherstellt [TA08]. Dadurch kann, während noch die Übertragung läuft und erst wenige Daten empfangen wurden, eine Vorschau erzeugt werden. So können bereits sehr früh erste Eindrücke über die Daten gewonnen werden.
Zur Veranschaulichung des Prinzips der *progressiven Bilddarstellung* sind in Abbildung 4.18 drei Bilder mit unterschiedlichen Detailstufen dargestellt.
Bisher gibt es kaum Ansätze zur Progression in der Informationsdarstellung. Erste Beispiele finden sich bei Rosenbaum [RS09]. Diese wurden im Rahmen dieser Arbeit und in Kooperation mit René Rosenbaum, UC Davis, ausgebaut und angewendet.

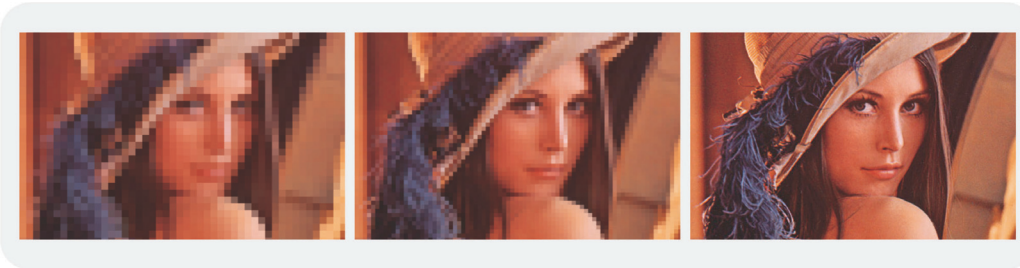

Abbildung 4.18: Progressive Verfeinerung der Details in einem Rasterbild

4.5.1 Progression in der Informationsdarstellung

Bei der *progressiven Informationsdarstellung* wird die visuelle Repräsentation analog zur progressiven Bildanzeige erzeugt und auf den verschiedenen Ausgabegeräten angezeigt, bis sie vollständig erstellt ist. Die Idee ist dabei, dass die progressive Übertragung individuell für jedes Gerät gestoppt wird,

wenn verfeinerte Informationen nicht mehr darstellbar sind.
Im *Smart-Meeting-Room* wird ein Gerät als Server ausgewählt, und von diesem werden die zu visualisierenden Daten in Form eines progressiven Datenstroms für andere Ausgabegeräte (Clients) bereitgestellt.
Die Besonderheit eines progressiven Datenstroms ist, dass jeweils nur die neuen Details in Form von Differenzen bezüglich der bereits übertragenden Daten gesendet werden. Bei der Progression gilt das Prinzip: „Kodiere einmal, aber verwende mehrfach!“ [RS09]. Bei der Anpassung der zu visualisierenden Daten auf mehrere unterschiedliche Ausgabegeräte wird immer derselbe Datenstrom verwendet. Je nach vorliegenden Geräteeigenschaften wird die progressive Übertragung und Ausgabe der zu visualisierenden Daten früher oder später abgebrochen.
Der Abbruch erfolgt, wenn weitere Details auf dem aktuellen Ausgabegerät nicht mehr wahrnehmbar sind oder wenn deren Darstellung zu viel Zeit beanspruchen würde. Der erste Fall ist gegeben, wenn neu hinzukommende Details im Subpixel-Bereich[9] liegen. Der zweite Fall tritt ein, wenn die zu visualisierende Datenmenge auf Grund ihrer Größe nicht in der vorgegeben Zeit verarbeitet und ausgeben werden kann. In Abbildung 4.19 wird dieses Prinzip anhand von der Treemap-Technik[10] veranschaulicht.
Eine grundlegenden Herausforderung bei *progressiven Informationsdarstellungen* in *Smart-Meeting-Rooms* ist die Wahl geeigneter Abbruchkriterien,

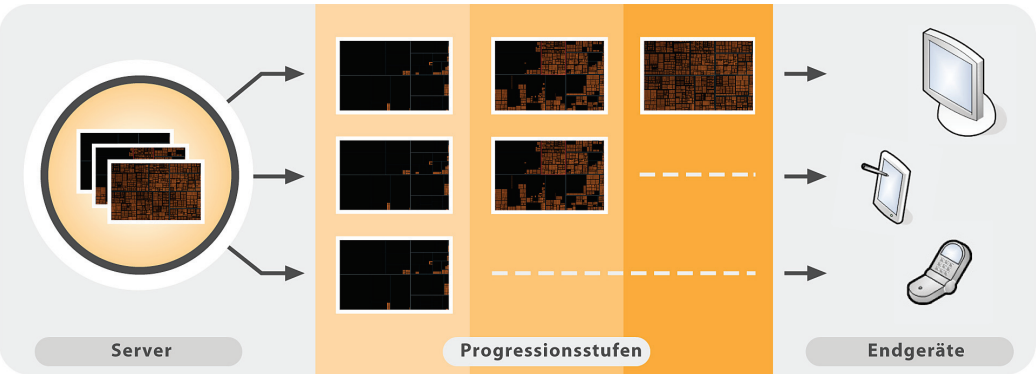

Abbildung 4.19: Adaptive progressive Informationsdarstellung auf unterschiedlichen Ausgabegeräten am Beispiel von Treemaps

[9] Die auszugebenen Informationen liegen im Subpixel-Bereich, wenn weitere Details nur noch Änderungen von kleiner als ein Pixel hervorrufen würden.

[10] Treemaps sind eine Visualisierungstechnik zur Visualisierung von hierarchischen Netzwerken. Jeder Knoten wird dabei als Rechteck dargestellt. Das umschreibende Rechteck repräsentiert den Wurzelknoten und wird entsprechend der Kindknoten weiter unterteilt. Dieser Prozess wird rekursiv wiederholt, bis alle Rechtecke nur noch Blattknoten repräsentieren (vgl. [Wat99, Shn92]).

ab derer keine weiteren Details mehr an das betreffende Ausgabegerät übertragen werden dürfen. Die zweite Herausforderung ist, dass für jedes Ausgabegerät, welches als Client im Sinne der progressiven Informationsdarstellung agiert, zu bestimmen ist, ob ein Abbruchkriterium bereits erreicht wurde.
In der vorliegenden Arbeit wurden im Folgenden beide Fragestellungen untersucht und Konzepte für deren Lösungen erarbeitet.

4.5.2 Bereitstellen geeigneter Abbruchkriterien

Progressive Informationsdarstellungen können verwendet werden, um an die Ausgabegeräte angepasste Informationsdarstellungen zu erzeugen. In diesem Kontext kommen die Auflösung und Größe des Displays als begrenzender Faktor hinzu. Zusätzlich kann das Prinzip der Progression verwendet werden, um Engpässen bei der Bandbreite und der Rechenleistung zu begegnen. Auch hieraus ergeben sich begrenzende Faktoren.

Auflösung des Displays

Ein Abbruchkriterium ist die Auflösung der Displays. Die Übertragung weiterer Details ist nicht sinnvoll, wenn die Auflösung des Ausgabegerätes nicht ausreichend dafür ist, derart feine Details auszugeben. Es sei also definiert: Das Abbruchkriterium ist erfüllt, wenn weitere zu übertragende Details im Subpixel-Bereich des Ausgabegerätes liegen.
Dieses Abbruchkriterium kann erweitert werden, indem die Abmessung des Displays und der Abstand der Betrachter zum Ausgabegerät mit herangezogen werden. Durch diese Kenngrößen kann die Sehschärfe des menschlichen Auges berücksichtigt werden. Die Sehschärfe beschreibt den Winkel, unter dem Abbilder zweier Gegenstände ins Auge einfallen müssen, damit diese noch als unterschiedlich wahrgenommen werden können. Der Durchschnittswert der Sehschärfe (auch als Visus bezeichnet) beträgt unter optimalen Bedingungen eine Bogenminute (1/60°) (vgl. [Gol00]). Das entspricht einem Abstand der beiden Gegenstände von 1,5 mm zueinander bei 5 m Abstand des Betrachters zu den Gegenständen. Hierfür müssen die Auflösung und die Abmessung des Displays bekannt sein. Steht der Betrachter in 5 m Entfernung zum Display und haben zwei benachbarte Pixel auf dem Display einen Abstand kleiner als 1,5 mm, so können diese nicht mehr als getrennt wahrgenommen werden. Die wahrnehmbare Auflösung des Displays ist in diesem Fall geringer. Allgemein kann die *wahrnehmbare Auflösung* durch die nachstehenden Formeln berechnet werden:

Abstand zweier Pixel (DPD) auf dem Ausgabegerät:

$$
\begin{aligned}
DPD &= Min(DPD_x; DPD_y) && \text{mit} \\
DPD_x &= \frac{Size_x}{Res_x} && \text{mit} \\
& && Size_x: \text{ Breite des Displays in } mm \\
& && Res_x: \text{ Auflösung des Displays in} \\
& && \quad \text{X-Richtung in } mm \\
DPD_y &= \frac{Size_y}{Res_y} && \text{mit} \\
& && Size_y: \text{ Höhe des Displays in } mm \\
& && Res_y: \text{ Auflösung des Displays in} \\
& && \quad \text{Y-Richtung in } mm
\end{aligned}
$$

Abstand zweier Pixel wahrnehmungsbasiert (DPP):

$$
\begin{aligned}
DPP &= visus \cdot UD && \text{mit} \\
& && visus: \text{ entspricht einem Wert von 0,0003} \\
& && UD: \text{ Abstand des Benutzers} \\
& && \quad \text{vom Ausgabegerät in } mm
\end{aligned}
$$

Wahrnehmbare Auflösung ($PRes$) in Pixel:

$$
\begin{aligned}
DP &= Max(DPP; DPD) \\
PRes_x &= \frac{Size_x}{DP} \\
PRes_y &= \frac{Size_y}{DP}
\end{aligned}
$$

Sind also der Betrachterabstand und die Abmessungen des Displays zusätzlich zur Auflösung des Displays bekannt, so ist das Abbruchkriterium erfüllt, wenn weitere zu übertragende Details ausschließlich im Subpixel-Bereich der *wahrnehmbaren Auflösung* liegen.

Verarbeitungszeit

Engpässe bei der Bandbreite und der Rechenleistung resultieren letztendlich in längeren Wartezeiten, bevor neue Details angezeigt werden. Bekanntlich [Kor09] sind Benutzer aber nur bereit, ca. 20 Sekunden ohne jede Reaktion der visuellen Ausgabe zu warten. Ein Abbruchkriterium für die Rechenleistung und Bandbreite kann also durch die Zeit angegeben werden, die zwischen der Ausgabe zweier Vorschauen der *progressiven Informationsdarstellung* vergeht. Es sei definiert: Das Abbruchkriterium ist erfüllt, wenn zwischen der Ausgabe zweier visueller Repräsentationen mehr als 20 Sekunden liegen.

4.5.3 Überprüfen der Abbruchkriterien

An dieser Stelle wird ein Konzept beschrieben, das hilft zu ermitteln, ob eines der drei genannten Abbruchkriteren erreicht ist. In diesem Rahmen wurde ein *On-the-Fly* Ansatz verwendet. Das Ziel ist es, anhand der bekannten Abbruchkriterien zu bestimmen, ob bei der *progressiven Informationsdarstellung* weitere Details in Form von weiteren Daten verarbeitet und dargestellt werden können oder nicht.
Die zu übertragenen Daten lassen sich dabei in der Regel nicht beliebig fein unterteilen, stattdessen bilden sie eine Hierarchie von Daten. Bei der *progressiven Informationsdarstellung* werden die Daten hierarchisch strukturiert. Dabei nehmen die Details pro Ebene zu. Es ist die Besonderheit zu beachten, dass in den Ebenen jeweils nur die Differenzen zur vorherigen Ebene gespeichert werden. Um alle Daten in einer visuellen Repräsentation darzustellen, ist es nicht ausreichend, nur die letzte Datenhierarchie-Ebene zu übertragen, sondern alle Datenhierarchie-Ebenen. Die visuelle Repräsentation wird also verfeinert, indem eine weitere Datenhierarchie-Ebene übertragen, verarbeitet und dargestellt wird.
Ziel des Ansatzes ist es folglich, anhand der Abbruchkriterien zu entscheiden, ob eine weitere Datenhierarchie-Ebene verarbeitet werden kann oder nicht.

On-the-fly-basiertes Vorgehen

Im Rahmen dieser Dissertation wurde ein Ansatz erarbeitet, den Ressourcenbedarf zur Laufzeit zu bestimmen, den eine Datenhierarchie-Ebene auf einem spezifizierten Ausgabegerät hat. Für diese Betrachtungen werden aktuelle Messungen der Displayauflösungen des Ausgabegerätes verwendet sowie die benötigte Übertragungs- und Dekodierzeiten der bereits übertragenen und

dargestellten Datenhierarchie-Ebenen. Stehen die notwendigen Kenngrößen zur Bestimmung der verfügbaren *wahrnehmbaren Auflösung* des Ausgabegerätes zur Verfügung, so wird diese ebenfalls berücksichtigt.
Allgemein werden bei der *On-the-Fly*-Abschätzung folgende Schritte abgearbeitet:

1. Die Messwerte zu den bereits übertragenen Datenhierarchie-Ebenen werden durch eine Filterfunktion geglättet. Dadurch wird der Einfluss von Ausreißern, die durch kurze Schwankungen bei der Bandbreite oder der Rechenleistung entstehen, reduziert. Erste Tests zeigen, dass hierfür ein Median-Filter verwendet werden kann (vgl. [TSR09]).

2. Anschließend werden die Messwerte mit einem „Wachstumsfaktor“ kombiniert, welcher den Zuwachs der Komplexität in den Daten repräsentiert. Dabei kann der Wachstumsfaktor entweder konstant sein, Meta-Daten-gesteuert, oder er kann durch eine Vorschau abgeschätzt werden. Dadurch ist es möglich, den Ressourcenbedarf der zu übertragenden Datenhierarchie-Ebene abzuschätzen.

3. Der abgeschätzte Ressourcenbedarf wird mit den gegebenen Abbruchkriterien verglichen. Ist dabei ersichtlich, dass die Übertragung der nächsten Datenhierarchie-Ebene das aktuelle Ausgabegerät überlastet, so wird diese nicht mehr übertragen.

4.5.4 Ergebnisse der progressiven Informationsdarstellung

Der hier vorgestellte Ansatz wurde exemplarisch auf die progressive Darstellung von JPEG2000 Bilddaten [TM01] sowie die progressive Informationsdarstellung mit Hilfe von Treemaps [Wat99, Shn92] übertragen. Damit werden beide Bereiche, Adaption auf der Bildebene und Adaption auf der Prozessebene (vgl. Abschnitt 4.1) exemplarisch angesprochen. In einer vorliegenden Untersuchung konnte gezeigt werden, dass das hier vorgestellte Konzept zur Abschätzung des Ressourcenbedarfs eingesetzt werden kann (vgl. Abbildung 4.20), um auf dieser Basis zu entscheiden, ob ein oder mehrere Abbruchkriterien erfüllt sind. Vorteil bei der *On-the-Fly*-Abschätzung des Ressourcenbedarfs ist, dass im Vergleich zu deren Genauigkeit nur ein geringer Berechnungsaufwand nötig ist.
Der Ansatz der *On-the-Fly*-Abschätzung des Ressourcenbedarfs geht von zwei Voraussetzungen aus.

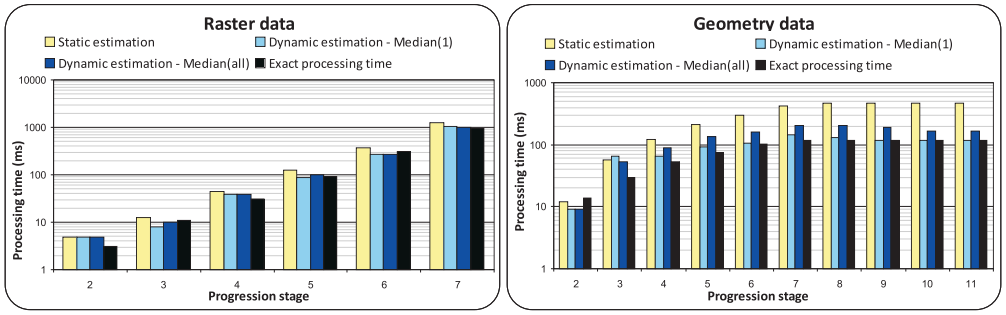

Abbildung 4.20: Gegenüberstellung der *On-the-Fly*-Abschätzung des Ressourcenbedarfs zum wirklichen Ressourcenbedarf für *links* Bilddaten und *rechts* progressive Treemaps, aus [TSR09]

1. Es müssen mindestens Datenhierarchie-Ebenen übertragen werden. Erst dann liegen ausreichend viele Messwerte vor, auf deren Basis eine Abschätzung durchgeführt werden kann.

2. Der „Wachstumsfaktor“ muss bekannt sein, mit dem die Komplexität der Daten zunimmt. Dieser kann aber bereits beim Erstellen der Datenhierarchie bestimmt werden.

Erst wenn diese Voraussetzungen erfüllt sind, werden brauchbare Abschätzung des Ressourcenbedarfs möglich.
Für die untersuchten Standardfälle zeigt der Ansatz gute Ergebnisse, offen ist, wie sich der Ansatz bei Sonderfällen, wie sehr stark veränderlichen Ressourcen verhält. Aus diesem Grund wird dem Benutzer die Möglichkeit gegeben, die Übertragung weiterer Datenhierarchie-Ebenen abzubrechen oder umgekehrt mehr Datenhierarchie-Ebenen zu übertragen, als von der automatischen Adaption vorgesehen sind.
Das vorgestellte Konzept wurde auf der IMC 2009 veröffentlicht (vgl. [TSR09]). Prinzipiell können *progressive Informationsdarstellungen* viele der Probleme, die bei Geräte-basierten Anpassungen von visuellen Repräsentationen auftreten, adressieren. Zudem lässt sich das Konzept erweitern und kann z. B. zur Übertragung zwischen jeder der Stufen des Data-State-Reference-Models eingesetzt werden.

4.6 Diskussion

Entsprechend der eingefügten Systematisierung wurde exemplarisch ein *Adaptionsmechanismus* pro Variante vorgestellt. Die Vielfalt der möglichen *Adaptionsmechanismen* ist weitaus größer. Je mehr *Adaptionsmechanismen* zur

Verfügung stehen, desto komplexer werden aber auch die *On-the-Fly*-Entscheidungen. Insbesondere werden zwei Probleme *On-the-Fly* entschieden:

Messen der Effektivität *On-the-Fly* Jede Informationsdarstellung erfordert andere Schwellwerte, gegebenenfalls sogar andere Maße und damit eine speziell zugeschnittene *Adaptionsstrategie*. In dieser Arbeit wurden vorrangig Scatterplots betrachtet. Auch für die müssen Maße und Schwellwerte bereitgestellt und *On-the-Fly* ausgewertet werden.

Abschätzung des Ressourcenverbrauchs Der Ressourcenverbrauch der *progressiven Informationsdarstellung* wird *On-the-Fly* bestimmt. Dafür sind sehr viele Messungen erforderlich, um eine adäquate Basis für die Abschätzung zu erhalten, die auch entsprechend der zugrunde liegenden Daten bzw. der verwendeten Visualisierungstechnik angepasst werden muss.

Die Vielzahl dieser Entscheidungen macht eine *On-the-Fly*-Auswertung schwierig. Insbesondere kann damit die Echtzeitfähigkeit der notwendigen Adaptionen nicht mehr gewährleistet werden.
Deshalb wird in der vorliegenden Arbeit mit Voreinstellungen gearbeitet, die basierend auf einer Charakterisierung von Geräteklassen und Voreinstellungen im Setup des *Smart-Meeting-Room* gespeichert und *On-the-Fly* ausgewertet werden, um entsprechende Anpassungen in Echtzeit vorzunehmen. Diese Einstellungen lassen sich jederzeit dynamisch und individuell ändern, um damit neben einer schnellen Verarbeitung auch eine große Flexibilität zu ermöglichen. Dabei gibt es vier Optionen, die Einstellungen zu ändern:

1. **Nutzer-bezogen:** Mit Hilfe von interaktive Eingabe über ein webbasiertes Interface durch die Nutzer des *Smart-Meeting-Room*.

2. **proaktiv:** Durch Auswerten von Intensionserkennung und Zuordnung von bestimmten Einstellungen zu Nutzeraktionen (vgl. [Bur10]).

3. **Aufgaben-bezogen:** Durch Auswertung des Aufgabenmodells eines entsprechenden Anwendungsszenarios im *Smart-Meeting-Room*. Das Aufgabenmodell wird von begleitenden Dissertationen der ersten Phase des GRK MuSAMA aufgestellt (vgl. [Pro10, Wur10]).

4. **Geräte-bezogen:** Durch Kopplung mit dem Displaymapper (vgl. [Hei09]). Der Displaymapper spezifiziert, auf welchem Gerät eine visuelle Repräsentation ausgegeben wird. Dabei lassen sich gerätespezifische Einstellungen definieren und verwenden.

Im Rahmen der vorliegenden Dissertation wurde die erste Option umgesetzt. Die anderen drei Optionen ergeben sich durch drei zeitgleich entwickelte Dissertationen in der ersten Phase des GRK MuSAMA und werden in der zweiten Phase des GRK's im Rahmen der vorgesehenen Experimentalstruktur zusammengeführt. Wichtig ist aber anzumerken, dass das hier entwickelte Framework zur Informationsdarstellung in *Smart-Meeting-Room* so flexibel entworfen wurde, dass es problemlos mit weiteren Tools verknüpft werden kann und damit auch weitere Einstellungen auswerten kann. Darauf wird im folgenden Kapitel genauer eingegangen.

Werden die genannten Punkte bei der Entscheidung mit berücksichtigt, ist kein Echtzeitverhalten mehr möglich oder die Optionen werden sehr stark eingeschränkt.
Eine Lösungsstrategie für eine Adaption in einem *Smart-Meeting-Room* macht es erforderlich, dass dessen aktueller Zustand bekannt ist.

- Die aktuelle Aufgabe wird in Form einer Aufgabenbeschreibung bereitgestellt. Diese wird auf der Basis des *page feature concepts* erstellt (vgl. [FRSF06]).

- Die nächsten Aktionen des Benutzers werden durch eine Intensionserkennung bereitgestellt. Mit dieser befassen sich andere Stipendaten im GRK MuSAMA (vgl. [Bur10]).

- Ein Displaymapper spezifiziert, auf welchem Ausgabegerät eine visuelle Repräsentation ausgegeben wird.

In der ersten Phase des GRK MuSAMA wurden für die einzelnen Bereiche Lösungen entworfen. Ziel ist es, diese zu nutzen und Geräteklassen zu erstellen. Die Geräteklassen werden anschließend verwendet, um für diese Voreinstellungen und Standards festzulegen und so zu bestimmen, welcher *Adaptionsmechanismus* verwendet wird. Bevor geeignete Geräteklassen abgeleitet werden können, ist es erforderlich, die möglichen Eigenschaften der Ausgabegeräte näher zu betrachten.
In der Informationsvisualisierung wichtige Eigenschaften der Ausgabegeräte wurden bereits in 2.2.2 angesprochen. Für die dort genannten Eigenschaften Bandbreite, Auflösung und Abmessung des Ausgabegerätes wurden jedoch noch keine Klassen angegeben. Das soll an dieser Stelle nachgeholt werden.

Die Bandbreite, mit welcher die einzelnen Geräte im *Smart-Meeting-Room*

miteinander verbunden sind, kann je nach verwendeter Technologie variieren. In Tabelle A.1 im Anhang wurden gebräuchliche Bandbreiten der unterschiedlichen Technologien zusammengetragen. Zur besseren Einordnung sollen für die Bandbreite folgende Klassen definiert werden:

mobile Verbindung Hierzu gehören Verbindungen wie GSM, UMTS, Bluetooth und WLAN.

feste Verbindung Die festen Verbindungen zeichnen sich dadurch aus, dass sie die Verbindung über ein Kabel aufbauen.

Die Auflösung verschiedener Displays variiert zwischen sehr geringen Auflösungen von beispielsweise Smartphones bis hin zu sehr hohen Auflösungen wie bei einer Powerwall. Zudem ist auch die physikalische Abmessung der einzelnen Displays sehr unterschiedlich. Zu den im Anhang in Tabelle A.2 genannten Vertretern wird daher neben der Auflösung auch deren physikalische Abmessung angegeben.
Bei den Ausgabegeräten ist auch der Abstand des Betrachters von Interesse. In [Ros09b] werden hierzu einige empirisch ermittelte Werte angegeben, welche in Tabelle A.3 im Anhang abgebildet sind.
Eine Klassifizierung der Ausgabegeräte, bei der die Eigenschaften des Displays berücksichtigt werden, ist nachfolgend angegeben:

Ultramobil In diese Klasse fallen Ausgabegeräte des Typs Smartphone, PDAs und Handys.

Mobil Hierzu gehören handelsübliche Laptops sowie TabletPCs.

Stationär In diese Klasse fallen die ortsgebundenen Ausgabesysteme wie Monitore, Projektoren und Powerwalls. Auch wenn sich für die Auflösung und den Abstand des Betrachters zum Ausgabegerät große Unterschiede ergeben, so sind die wahrnehmbaren Auflösungen der stationären Ausgabegeräte dennoch ähnlich.

Eine Zuordnung von *Adaptionsmechanismen* zur Geräteklassen bilden die Voreinstellungen. Diese müssen dynamisch änderbar sein. Um dies zu realisieren, ist ein flexibles, dynamisch zu konfigurierendes Visualisierungsframework erforderlich, für welches im nachfolgenden Kapitel ein Konzept entworfen wird. Die Zuordnung zwischen Geräteklasse und *Adaptionsmechanismen* soll ebenfalls im nachfolgenden Kapitel vorgenommen werden.

Kapitel 5

Framework Entwurf

5.1 Vorbetrachtung

Aus der Anforderungsanalyse können die folgenden Punkte für das Framework abgeleitet werden:

- *On-the-Fly*-Konfiguration der Visualisierungspipeline. Auftretende Veränderungen bezogen auf die zur Verfügung stehenden Ausgabegeräte müssen dabei automatisch erkannt und *On-the-Fly* berücksichtigt werden.
- Unterstützung einer gleichzeitigen Informationsdarstellung auf multiplen, heterogenen Ausgabegeräten.

Die Visualisierungspipeline, so wie sie mit dem Data-State-Reference-Model beschrieben ist, ist modular aufgebaut (vgl. 2.2.1). Deshalb können einzelne Module auch auf unterschiedlichen Geräten ausgeführt und somit ein verteiltes Geräteensemble, wie es in einem *Smart-Meeting-Room* zur Verfügung steht, genutzt werden. Um aber die Abarbeitung der Pipeline sicherzustellen, müssen die dynamischen Veränderungen im Ensemble berücksichtigt werden. Daraus ergeben sich besondere Anforderungen an die Software-Architektur.

Im Grundlagenkapitel wurden zur möglichen Realisierung einer softwaretechnischen Basis für Informationsdarstellungen in *Smart-Meeting-Room* agentenbasierte und serviceorientierte Ansätze vorgestellt (vgl. 2.3). Beide Ansätze haben Vor- und Nachteile und lassen sich prinzipiell zur Umsetzung des Framework nutzen. Da aber im GRK MuSAMA die Mehrzahl der entwickelten Lösungen auf serviceorientierten Ansätzen basiert (vgl. z. B. [Mar10, Plo10]), soll auch in der hier vorliegenden Arbeit eine serviceorientierte Architektur verwendet werden, um so die Schnittstelle zu anderen Arbeiten zu verbessern

und zu vereinfachen.
Eine serviceorientierte Architektur (SOA) verfügt über die folgenden Eigenschaften:

1. *Flexibilität*: Eine SOA vereinfacht die Adaption der Software an veränderte Anforderungen, da sie stark modular aufgebaut ist.

2. *Reduzierung der Komplexität*: Die Services einer SOA stellen einen einfachen Funktionsumfang zur Verfügung, erst indem mehrere Services kombiniert werden, können komplexe Aufgaben gelöst werden.

3. *Wiederverwendbarkeit*: Services sind nicht an eine konkrete Aufgabe gebunden und können daher wiederverwendet werden.

4. *Kompatibilität*: Services bedienen ein definiertes Interface. So kann die Kompatibilität zwischen unterschiedlichen Implementierungen eines Services gewährleistet werden.

Anstatt eine fest zusammengestellte Visualisierungspipeline zu verwenden, wird bei der im nächsten Abschnitt vorgestellten serviceorientierten Architektur die Visualisierungspipeline dynamisch zur Laufzeit aus Services zusammengestellt und dabei die aktuelle Situation des *Smart-Meeting-Room* berücksichtigt.

5.2 Konzept der Framework Architektur

5.2.1 Eine servicebasierte Visualisierungspipeline

Die Ausgangsbasis für die Framework-Architektur bildet das Data-State-Reference-Modell (DSRM)(vgl. 2.2.1). Die grundlegende Idee besteht nun darin, jeden Operator des Data-State-Reference-Modells als Service einer serviceorientierten Visualisierungsarchitektur aufzufassen. Prinzipiell lässt sich jeder Operator auf einem unterschiedlichen Gerät ausführen. Das DSRM unterscheidet vier verschiedene Klassen von Operatoren auf den jeweiligen Stufen und drei Klassen von Operatoren zwischen den Stufen.
Das Framework baut demnach auf einzelnen Services auf, die sieben verschiedenen Operatorklassen zugeordnet werden können und aus denen die Visualisierungspipeline *On-the-Fly* gebildet wird.
Die Anzahl der verfügbaren Operatoren kann sehr groß sein, so dass es schwierig wird, in Echtzeit die entsprechenden Operatoren auszuwählen und in die

Visualisierungspipeline einzubinden. Aus diesem Grund werden in dieser Arbeit drei verschiedene Abstraktionsstufen eingeführt, um die Visualisierungspipeline zu beschreiben:

- Operatoren: Auf dieser Ebene werden die Operatoren explizit angegeben. Beispiele für Operatoren auf der Stufe der analytischen Abstraktionen sind die Operatoren *minimal linkage* und *Hierarchisches Clustering* auf der Stufe der visuellen Abstraktionen z. B. das *Binning*.

- Operator-Interfaces: Hierbei werden ähnliche Operatoren unter einem gemeinsamen Interface zusammengefasst. Ziel ist es, beim Auswahlprozess auf eine prinzipielle Funktionalität und nicht nur auf einem einzelnen Algorithmus zugreifen zu können. Das bedeutet auch, dass zu einem Operator-Interface mehrere Operatoren existieren, die unterschiedliche Methoden bereitstellen, aber dasselbe Ziel verfolgen. So lassen sich beispielsweise die beiden Operatoren der Stufen aus dem vorhergehenden Beispiel durch das Operator-Interface *Clustering* verallgemeinern.

- Pipeline-Templates: Abstrahieren noch weiter und beschreiben dabei eine gesamte Visualisierungspipeline unter Verwendung von Operator-Interfaces mit den zugehörigen Verknüpfungen. Darüber hinaus werden für die verwendeten Operator-Interfaces *Default-Settings* angegeben, die eine grundlegende Parametrisierung erlauben und damit eine schnelle Konfiguration der Visualisierungspipeline erlauben.

Die Pipeline-Templates müssen in einem Präprozess durch den Domainexperten erstellt werden.
Wird also z. B. der *Smart-Meeting-Room* zur Diskussion der globalen Erwärmung genutzt und die beteiligten Klimaforscher arbeiten vorrangig mit Parallelen Koordinaten, Scatterplots und farbkodierten Karten, so müssen im Vorfeld Pipeline-Templates zur Verfügung gestellt werden, die diese Ausgaben realisieren.
Bei einer Sitzung von Medizinexperten werden dagegen z. B. Pipeline-Templates benötigt, die eine Darstellung von CT-Daten realisieren. Das heißt also, ein Domainexperte stellt für bestimmte Setups eines *Smart-Meeting-Room* die entsprechenden Pipeline-Templates zusammen, die dann je nach Anwendungsfall initialisiert werden. Damit muss *On-the-Fly* beim Aufbau der Visualisierungspipeline nur noch zwischen einer kleinen Anzahl von Templates entschieden werden anstelle einer Auswahl von Operatoren aus einem großen Operatorpool.
Die Templates beinhalten Operator-Interfaces. Die Zuordnung von konkreten Operatoren erfolgt dann entsprechend des verfügbaren Geräteensembles.

Wie dabei genau vorgegangen wird, soll später im Abschnitt Binding (vgl. 5.3.3) erläutert werden.
In der hier verwendeten serviceorientierten Architektur werden die Operatoren jeweils durch einen separaten Service realisiert und damit auch die Operator-Interfaces durch entsprechende Service-Interfaces. Durch die Verwendung von Services in dieser Form ergeben sich mehre abstrakte Schichten im Software-Framework, die nachfolgend erläutert werden sollen.

5.2.2 Schichtenmodell

Das Softwareframework ist in drei abstrakte Schichten unterteilt.

- *Serviceschicht* umfasst die konkrete Implementierung der Operatoren. Das heißt, sie besteht aus mehreren Operatoren, die in einzelnen Services gekapselt sind und jeweils ein bestimmtes Service-Interface bedienen. Sie stellt somit den in einer SOA erforderlichen Serviceprovider dar (vgl. [HS05, SG08]). Die Services im Framework verfügen über keine Kenntnisse von anderen Services. Der gesamte Datenaustausch und die Steuerung der Services wird über die Steuerschicht abgewickelt.

- Die *Steuerschicht* übernimmt die Steuerung des Framework und der einzelnen Services. Konkret fungiert sie als Servicebroker und als Serviceconsumer (vgl. [HS05, SG08]). Zudem hat sie die Aufgaben neu hinzukommende und wegfallende Geräte zu identifizierten. Realisiert wird dies über die Services auf den Geräten. Melden sich Services eines neuen Gerätes bei der Steuerschicht an, so wird dadurch die Steuerschicht gleichzeitig über die Existenz des neuen Gerätes informiert. Umgekehrt wird die Steuerschicht auch indirekt über den Wegfall eines Gerätes informiert, wenn Services auf diesem Gerät nicht mehr erreichbar sind. Zudem überwachen die Services die Eigenschaften der Geräte, auf denen sie ausgeführt werden und informieren entsprechend die Steuerschicht über Veränderungen. Verändert sich das Geräteensemble des *Smart-Meeting-Room* oder die Eigenschaften einzelner Ausgabegeräte, wird ein Update der Visualisierungspipeline durchgeführt. Das heißt, die aktuellen Settings der Operatoren in den einzelnen Services werden angepasst.

- Die *Modellschicht* umfasst die Beschreibungen der in Abschnitt 2.2.2 identifizierten Rahmenbedingungen. Durch diese Schicht wird eine Schnittstelle zu anderen Arbeiten im GRK realisiert. Die Modellschicht beeinflusst die Default-Settings und kann somit die Services indirekt anpassen.

In Abbildung 5.1 ist der Zusammenhang zwischen den Schichten schematisch abgebildet.
Im Rahmen dieser Arbeit wurde hauptsächlich auf die Service- und Steuer-

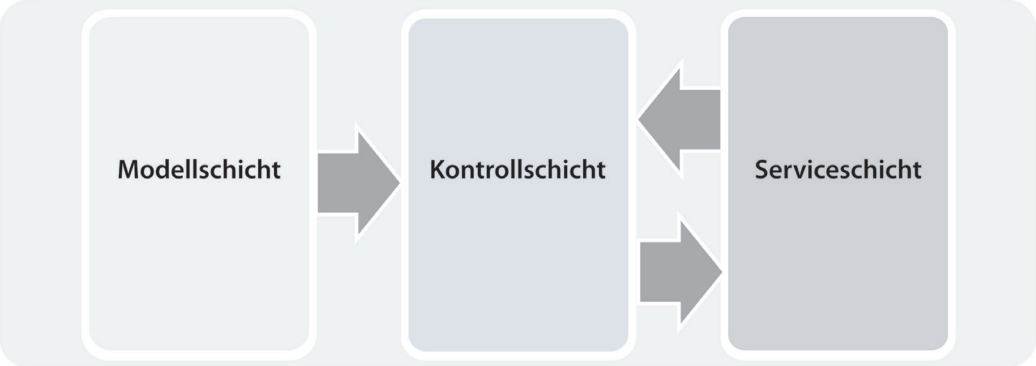

Abbildung 5.1: Schematische Darstellung des Zusammenspiels der drei Schichten im Framework

schicht im Framework fokussiert. Diese realisieren zusammen eine Visualisierungspipeline zur Erzeugung einer visuellen Repräsentation. Die Beschreibungen der Modellschicht können aus dem Tupel-Space gewonen werden. Der Tupel-Space erfasst die aktuelle Situation eines *Smart-Meeting-Room* (vgl. [MK88]).
Das Konzept des Tupel-Space wurde zeitlich parallel zu der vorliegenden Arbeit im Rahmen des GRK's eingebunden, so dass er für die konkrete Umsetzung nicht zur Verfügung stand. Aber durch die Orientierung auf sinnvolle Standard-Settings ist es jederzeit möglich, diese Standardwerte durch Werte aus dem Tupel-Space zu modifizieren bzw. zu ersetzen. Der im Rahmen dieser Arbeit entwickelte Mechanismus zur Erzeugung der Visualisierung muss dabei nicht verändert werden.

5.3 Adaption und Interaktion

5.3.1 Setup

In einem *Smart-Meeting-Room* gibt es ein initiales Setup, welches die stationären Geräte und bereits aktive Services enthält.
Daneben stehen entsprechend des konkreten Anwendungsfalls Pipeline-Templates zur Verfügung, die mit Standardwerten durch einen Domainexperten initialisiert sind. Da man nicht immer davon ausgehen kann, dass geeignete Templates durch einen Domainexperten auch im Vorfeld bereitgestellt werden, wurde zudem ein so bezeichnetes Default-Template definiert.

Dieses Default-Template, welches nur jeweils ein Operator-Interface anbietet, um die Daten zu verarbeiten, ein Mapping durchzuführen und dann mit Hilfe des Rendering eine visuelle Repräsentation zu erzeugen, kann als Basis genutzt werden, um weitere Templates zu erzeugen, die besser an die vorliegende Situation angepasst sind. Dieser Schritt kann prinzipiell interaktiv oder automatisch erfolgen. Darauf wird im Folgenden (vgl. 5.3.4) noch genauer eingegangen.
Wichtig an dieser Stelle ist der Hinweis, dass dabei insbesondere die Dynamik im *Smart-Meeting-Room* berücksichtigt wird. Dynamische Veränderungen können sich bezogen auf die zu verarbeitenden Daten, die vorliegende Fragestellung, bei den Anforderungen durch die Informationsdarstellung oder durch die Geräte ergeben. Bei auftretenden Veränderungen wird eine Adaption des Default-Templates vorgenommen, indem Service-Interfaces entfernt oder weitere eingefügt werden. Eine voll automatische Anpassung ist ein schwieriges Problem, da prinzipiell sehr viele dynamische Veränderungen möglich sind. Nähere Ausführungen gehen über eine Erläuterung zu einem einfachen Setup hinaus und sollen aus diesem Grund in einem separaten Unterabschnitt behandelt werden (vgl. 5.3.6).

5.3.2 Rollenverteilung

An der Erzeugung einer Visualisierung in einem *Smart-Meeting-Room* sind verschiedenen Personen beteiligt, die je nach ihrer Rolle die Prozesse unterschiedlich beeinflussen.

Domainexperte Seine Aufgabe besteht darin, den Anwendungskontext der Informationsvisualisierung zu definieren. Hierzu wählt er entweder aus verfügbaren Pipeline-Templates ein geeignetes aus oder erstellt bei Bedarf ein neues Template. Es ist auch die Aufgabe des Domainexperten die Default-Settings in den Pipeline-Templates zu setzen und so ein initiales Setup durchzuführen.

Visualisierungsautor Er hat die Aufgabe, die Informationsvisualisierung zu erzeugen. Er ergänzt hierzu die vom Domainexperten gelieferten Pipeline-Templates bzw. hat die Möglichkeit, für die aktuelle Situation besser geeignete Templates auszuwählen.
Er ist außerdem für die Zuordnung von Services zu den Services-Interfaces zuständig. Hierbei wird er durch das Framework unterstützt. Es obliegt aber dem Autor, hierbei getroffene automatische Entscheidungen des Frameworks durch ein manuelles Eingreifen zu verändern. Hierbei kann er sich aus der Liste der angemeldeten Services bedienen.

Betrachter Er hat die Möglichkeit, sich die vom Autor erzeugte Visualisierung anzusehen, kann aber auch persönliche Einstellungen an der Informationsvisualisierung vornehmen. Konkret kann er die verwendeten Settings anpassen. Nimmt der Benutzer Veränderungen vor und sind diese widersprüchlich zu gesetzten Settings von anderen Benutzern und nur für die Ausgabe auf seinem eigenen Gerät gedacht, so wird die Pipeline automatisch verzweigt. Das heißt, für diesen Benutzer wird eine eigene angepasste visuelle Repräsentation erzeugt. In diesem Fall erfolgt auch eine benutzer-angepasste Zuordnung der Services zu den Services-Interfaces.

Um die drei verschiedenen Rollen an einem Beispiel zu beleuchten, soll eine Diskussion zur globalen Klimaveränderung betrachtet werden. Der Domainexperte hat bisher dass Visualisierungstool „Vegetation Visualizer“ [NHP+09] in das Setup des *Smart-Meeting-Room* eingespielt. Tom ist der Visualisierungsexperte. Er generiert auf seinem Laptop einige visuelle Repräsentationen, indem er bereits vorhandene Pipeline-Templates anpasst. Die generierte Scatterplot-Matrix zeigt ein interessantes Verhalten des klassifizierten Datensatzes. Tom entscheidet, die visuelle Repräsentation auf einem Projektor auszugeben und so allen Teilnehmern die visuelle Repräsentation zugänglich zu machen. Dazu ergänzt Tom entsprechend das vorhandene Pipeline-Template für die Ausgabe auf dem Projektor.
Um die Ausgabe besser zu studieren und eigenen Hypothesen zu überprüfen, fordern einige Betrachter die Scatterplot-Matrix an. Das Framework verzweigt die Pipeline entsprechend, und die Betrachter können die Settings ihrer visuellen Repräsentation anpassen.

Wenn kein Domainexperte im Vorfeld das Setup des *Smart-Meeting-Room* entsprechend initialisiert hat, muss man voraussetzen, dass mindestens ein Visualisierungsautor an der Sitzung teilnimmt. Bezogen auf das obige Beispiel würde das bedeuten, dass Tom auf seinem Laptop den *Vegetation Visualizer* zur Verfügung hat und die entsprechenden Bilder erzeugt. Es könnte aber auch bedeuten, dass Tom ausgehend vom Default-Template verfügbare Services anbindet.
Wenn an der Sitzung kein Visualisierungsexperte teilnimmt, muss vorausgesetzt werden, dass im Vorfeld ein Domainexperte die entsprechenden Templates bereitgestellt hat, so dass sie für die Betrachter zur Verfügung stehen. Wir können dann also davon ausgehen, dass für ein gegebenes Anwendungsszenario geeignete Pipelinetemplates mit entsprechenden Service-Interfaces zur Verfügung stehen.
Die Frage ist nun, wie diese Service-Interfaces konkreten Services zugeordnet

werden. Dies kann einerseits durch den Visualisierungsexperten erfolgen, der die Funktionsweise der einzelnen Operatoren im DSRM und damit die entsprechenden Services kennt und geeignet zuordnen kann. Es muss aber auch automatisch erfolgen können, um eine flexible *On-the-Fly*-Konfigurierung zu ermöglichen und damit auch dem automatischen Adaptionsprozess (vgl. 4) zu unterstützen. Darauf wird im nächsten Abschnitt genauer eingegangen.

5.3.3 Binding

Unter einem Binding wird die Zuordnung einer Service-Implementierung zu einem gegebenem Service-Interface verstanden. Das Binding stellt ein wesentliches und schwerwiegendes Problem im Adaptionsprozess von Informationsvisualisierungen dar. Aus diesem Grund wurden unterschiedliche Optionen bereitgestellt, wie bei der Zuordnung von Services zu Service-Interfaces vorgegangen werden kann. Dabei wird vorausgesetzt, dass über das Geräteensemble verteilt die verschiedenen Services der Serviceschicht zur Verfügung stehen. Beim Binden der Service-Implementierungen sind dann die folgenden Strategien möglich, die jeweils unterschiedliche Schwerpunkte setzen.

Option 1 geräte-orientiert Beim Binding werden die Service-Implementierungen bevorzugt, die einen geringen Kommunikationsaufwand verursachen. Prinzipiell ist der Komunikationsaufwand am geringsten, wenn alle Service-Implementierungen auf einem Gerät ausgeführt werden. Dieser Fakt impliziert zudem, dass dieses Ausgabegerät zum bevorzugten Gerät wird, auf dem die Services ausgeführt werden, da nur so eine Übermittlung der visuellen Repräsentation auf das Ausgabegerät vermieden werden kann.

Diese Strategie hat ihren Vorteil in der verbesserten Performance bei der Erzeugung und der Aktualisierung der visuellen Repräsentation. Die geräte-übergreifenden Kommunikationen entfallen oder werden reduziert.

Diese Strategie ist allerdings bei einem ressourcenschwachen Ausgabegerät nicht vorteilhaft. In diesem Fall soll die Erzeugung von der Ausgabe der visuellen Repräsentation getrennt werden. Ausschlaggebend ist dabei die Geräteklasse des Ausgabegerätes.

Liegt ein Ausgabegerät der Leistungsklasse *Kleinstgerät*(vgl. 5.3.6) vor, wird die visuelle Repräsentation auf einem anderen Gerät erzeugt und nur die Ausgabe erfolgt auf dem gewählten Ausgabegerät. Bei dieser Strategie liegt demnach der Fokus auf einer geräte-basierten Anpassung

der visuellen Repräsentation, speziell die Performance und die schnelle Ausgabe der visuellen Repräsentation stehen im Vordergrund.

Option 2 Nutzer-orientiert Ein Template wird entweder von einem Benutzer ausgewählt oder von diesem interaktiv modifiziert, in beiden Fällen werden beim Binding die Service-Implementierungen bevorzugt, die sich aus seinem User-Profil ergeben bzw. auf dem Gerät dieses Benutzers aktiv sind.
Der Grundgedanke dabei ist, dass jeder Benutzer seine bevorzugten Service-Implementierungen selbst in den *Smart-Meeting-Room* einbringt. Werden diese Service-Implementierungen beim Binding bevorzugt, resultiert daraus eine visuelle Repräsentation, die den Bedürfnissen des Benutzers besser gerecht wird. Diese Strategie stellt den Benutzer in den Vordergrund und passt die visuelle Repräsentation konkret an seine Interessen an.

Option 3 umgebungs-bezogen Neben den beiden genannten Optionen können auch die Informationen genutzt werden, die andere Quellen zur Beschreibung der Umgebung zur Verfügung stellen.
Im Rahmen des GRK MuSAMA wird die Kommunikation dabei über den *Tupel Space* abgewickelt. Ein Tupel Space kann bildlich gesprochen als eine Pinnwand aufgefasst werden. Benötigen einzelne Prozesse eine Lösung für ein Problem, für das bei diesem Prozess keine Expertise vorliegt, so stellt er über den Tupel Space eine allgemeine Anfrage. Er macht über die „Pinnwand“ seine Anfrage öffentlich. Andere Prozesse überwachen den Tupel Space, sollten diese bei den gestellten Problemen eines identifizieren, welches sie mit ihrer Expertise lösen können, so nehmen sie das Problem aus dem Tupel Space, lösen es und legen die Lösung anschließend wieder auf dem Tupel Space ab. Der Prozess, der ursprünglich das Problem in den Tupel Space gestellt hat, wird darüber informiert und entnimmt die Lösung zu seinem Problem.
Mit diesem Funktionsprinzip kann die erste der genannten Optionen zusätzliche Anfrage nach „Bandbreiten technisch“ nahen Rechnern in den Tupel Space legen. Service-Implementierungen auf Geräten, die über eine sehr hohe Bandbreite mit dem Ausgabegerät verbunden sind, können dann beim Binding bevorzugt behandelt werden.
Die 2. Option kann dagegen Anfragen zur aktuellen Nutzer-Situation stellen. So kann z. B. der Tupel Space verwendet werden, um eine Anfrage nach den aktuellen Aufgaben eines Nutzers zu stellen, die im Taskmodell abgelegt sind. Ein Prozess, der über die notwendigen Kenntnisse verfügt, kann dann Parameter Settings liefern, um die Default

Settings entsprechend zu modifizieren. Hierdurch kann z. B. die aktuelle Aufgabe bei der Auswahl der Service-Implementierungen berücksichtigt werden.

Die vorgestellten Optionen stellen grundsätzliche Strategien dar, um zu entscheiden, welcher konkrete Service aus dem Service-Pool zur Anwendung kommen soll. Daneben müssen bei allen drei Optionen die im Kapitel 4 entwickelten *Adaptionsmechanismen* berücksichtigt werden, welche die Informationsdarstellung automatisch an die gegebenen Ausgabegeräte anpasst. Darauf wird im Abschnitt 5.4 noch genauer eingegangen. Neben automatischen Anpassungen muss es aber nach wie vor möglich sein, dass der Nutzer als Visualisierungsautor oder Betrachter interaktive Anpassungen vornimmt.

5.3.4 Interaktive Adaption

Die Adaption der visuellen Repräsentation in einem *Smart-Meeting-Room* muss interaktiv durch die Benutzer beeinflusst werden können, um in dynamischen Situationen besser reagieren und automatisch generierte Entscheidungen anzupassen oder zu verändern. Für diesen Zweck wurde im Rahmen dieser Arbeit ein spezieller Editor bereitgestellt. Dessen Funktionalität geht über den eines herkömmlichen Editors hinaus. Neben den üblichen Funktionen wie Erstellen, Speichern und Laden von Pipeline-Templates und den zu den Templates gehörenden Settings verfügt der Editor über die Funktion, aktuell verwendete Templates *On-the-Fly* zu modifizieren. Das heißt, alle Manipulationen eines solchen Templates im Editor haben einen unmittelbaren und direkten Einfluss auf die aktuell dargestellten visuellen Repräsentationen und geben dem Visualisierungsexperten bzw. Betrachter ein unmittelbares Feedback. So ist eine interaktive Adaption der visuellen Repräsentation an unterschiedliche Rahmenbedingungen (vgl. 2.2.2) möglich, ohne dass weitreichende Kenntnisse über das darunter liegende Visualisierungsframework erforderlich sind.
Der Editor kann sowohl von Visualisierungsexperten als auch vom Betrachter entsprechend genutzt werden, um während einer Sitzung die visuelle Repräsentation wie gewünscht zu adaptieren. Der Domainexperte verwendet aber auch den Editor, um Templates und passende Default-Settings zu erstellen. Der vorliegende Editor ist im Rahmen einer Studienarbeit entstanden (vgl. [Fra10] und Abbildung 5.2).

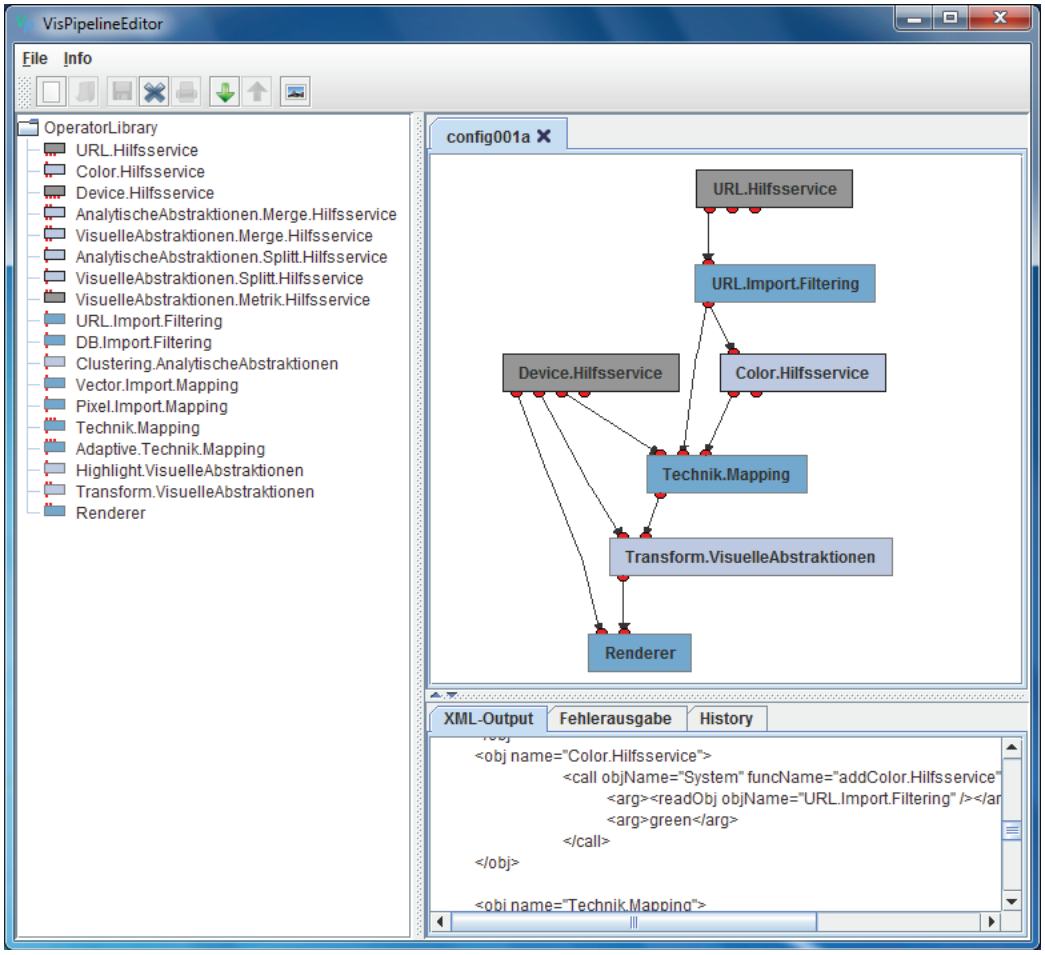

Abbildung 5.2: Screendump des Editors zum interaktiven Konfigurieren der Visualisierungspipeline.

5.3.5 Multiple Ausgabegeräte

In einem *Smart-Meeting-Room* stehen mehrere Ausgabegeräte zur Verfügung, auf denen eine oder mehrere visuelle Repräsentationen angezeigt werden können.
Deshalb müssen vom Framework die folgenden zwei Fälle behandelt werden:

Gleiche visuelle Repräsentation Dieselbe visuelle Repräsentation wird auf unterschiedlichen Geräten ausgegeben: In diesem Fall muss eine gerätebezogene Adaption erfolgen. Die Adaption erfolgt durch Hinzufügen oder Entfernen von Service-Interfaces in dem zu adaptierenden Pipeline-Templates durch Anpassung der Settings bzw. durch Anpassen des Bindings der Service-Implementierung an die Service-Interfaces

(vgl. 4 und 5.3.3). Somit kann eine benutzer- und geräteangepasste Adaption erreicht werden. Abbildung 5.3 zeigt dies an einem Beispiel.

Unterschiedliche visuelle Repräsentationen In diesem Fall wird die Visualisierungspipeline verzweigt, das heißt, es werden zusätzliche Services eingespeist, die eine weitere visuelle Repräsentation erzeugen, z. B. auf dem persönlichen Gerät eines Betrachters und unter Berücksichtigung seiner individuellen Interessen. Die Ansatzpunkte für die Verzweigung können konzeptionell auf jeder Stufe der Pipeline definiert werden. Abbildung 5.4 veranschaulicht dieses Prinzip.
Die hier vorliegende Implementierung unterscheidet aktuell zwei Fälle:

1. Der Nutzer fordert eine andere Darstellung an. In diesem Fall setzt die Verzweigung der Visualisierungspipeline beim Mapping an, um eine veränderte Darstellung zu realisieren. Das bedeutet, dass für zwei verschiedene Darstellungen, die aber auf den gleichen zu visualisierenden Daten beruhen, ab dem Mapping zwei separate Zweige ausbildet. Gegebenenfalls wird eine gerätebezogene Anpassung der visuellen Repräsentation vorgenommen.
2. Der Nutzer fordert die gleiche Darstellung auf einem anderen, aber ähnlichem Gerät an. Tritt dieser Fall auf, muss die visuelle Repräsentation eigens für das neue Ausgabegerät erzeugt werden. Prinzipiell wird dabei die Visualisierungspipeline ab dem Rendering verzweigt, um eine visuelle Repräsentation für das neu hinzukommende Ausgabegerät zu erzeugen und auf diesem auszugeben.

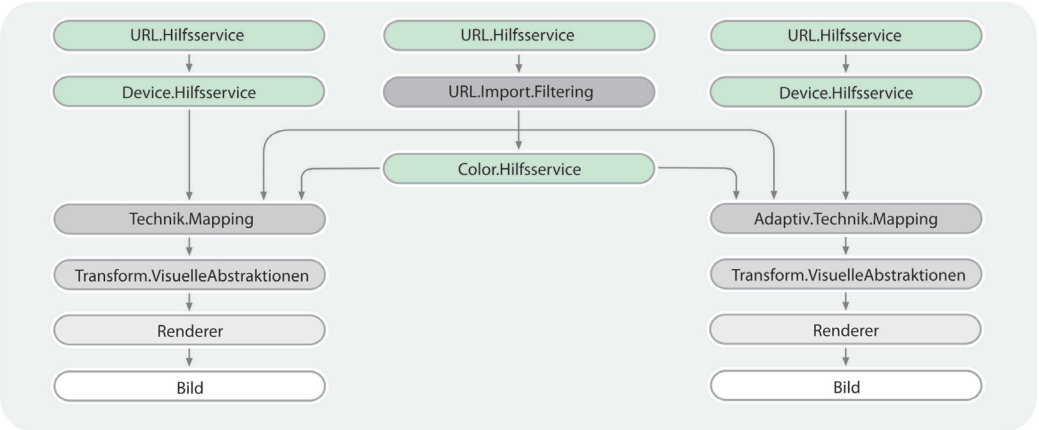

Abbildung 5.3: Darstellung der gleichen Daten, angepasst an verschiedene Ausgabegeräte. Die Verzweigung der Visualisierungspipeline erfolgt ab dem Mapping.

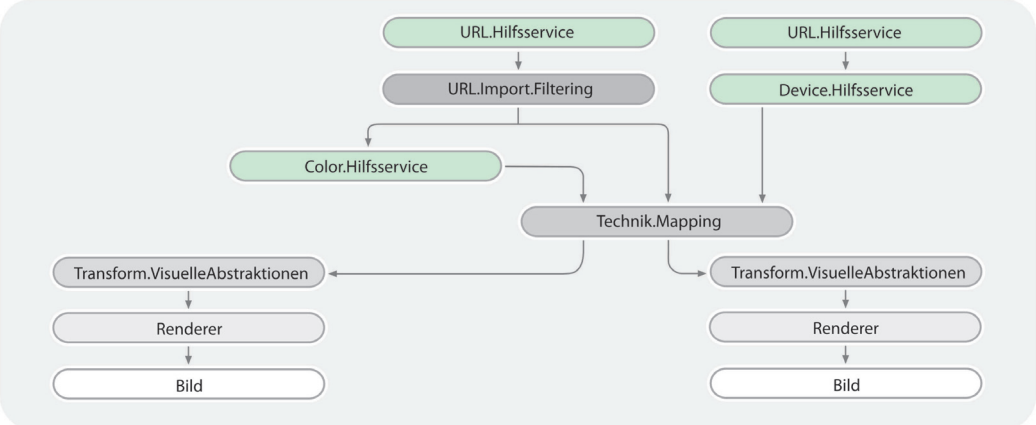

Abbildung 5.4: Verzeigung der Visualisierungspipeline abhängig von den Ausgabegeräten, diese erfolgt ab dem Rendering.

5.3.6 Dynamik der Ausgabegeräte

Bei der in *Smart-Meeting-Room* auftretenden Dynamik bezogen auf Ausgabegeräte ist eine automatische On-the-Fly-Konfigurierung der Visualisierungspipeline eventuell sehr zeitaufwendig.
Um dennoch zu einer praktikablen Lösung zu kommen, werden Default-Settings verwendet. Hierbei handelt es sich um vordefinierte Parametermengen, durch welche die notwendigen Anpassungen des Framework beschrieben werden können. Diese Parametermengen geben ein Binding von Service-Implementierungen zu den vorliegenden Services-Interfaces vor, weiterhin sind Steuerparameter enthalten, um die Service-Implementierungen an die aktuelle Situation anzupassen.
Die prinzipielle Idee besteht darin, eine kleine Menge an vordefinierten Default-Settings vorrätig zu haben, um damit die am häufigsten auftretenden Situationen bei sich verändernden Ausgabegeräten abdecken zu können. Ist also eine Ausgabe auf einem der nachfolgend genannten Geräteklassen erforderlich, weil ein Gerät ausfällt, hinzukommt oder die visuelle Repräsentation zusätzlich auf diesem Ausgabegerät dargestellt werden soll, so wird anhand der für das Ausgabegerät hinterlegten Geräteklasse das geeignete Default-Stetting ausgewählt und die visuelle Repräsentation erzeugt und ausgegeben. Für die Definition der drei nachfolgenden Default-Settings wird auf die Geräteklassen auf Abschnitt 4.6 zurückgegriffen:

- *Ultra Mobil Geräte*: Hierzu werden Handys oder Smartphones gerechnet. Diese zeichnen sich durch eine sehr kleine Displayfläche aus. Die geringe Leistungsfähigkeit dieser Geräte unterstützt nicht das Rendering und die Ausgabe von großen Mengen an komplexen visuellen Primitiven

in angemessener Zeit. Das für diese Gerätesituation definierte Default-Setting stellt sicher, dass die Erzeugung der visuellen Repräsentation auf ein leistungsstarkes Gerät im *Smart-Meeting-Room* ausgelagert wird. Das Gerät, was ersatzweise das Rendering übernimmt, kennt die Display-Parameter des Kleinstgerätes und erzeugt abhängig von den Daten und der aktuellen Situation eine visuelle Repräsentation. Diese wird in Form eines Bildes gespeichert, auf das Ausgabegerät übertragen und dort ausgegeben. Das Default-Setting beeinflusst das Binding. Die gegenwärtige Implementierung enthält als Standard-Setting ein Clustering, um die Menge der darzustellende Informationen einzuschränken und das Binning um die Komplexität der visuellen Repräsentation durch geeignete Abstraktionen zu reduzieren.

- *Mobile Geräte*: Hierzu gehöre Geräte wie PDAs oder Netbooks. Bei der Erstellung der visuellen Repräsentation muss auch hier die geringere Displayfläche und die geringere Leistungsfähigkeit berücksichtigt werden. Im Unterschied zu den Kleinstgeräten sind diese aber in der Lage, selbst die Erzeugung einfacher visueller Repräsentation durchzuführen. Das bedeutet, die Darstellung darf eine gewisse Komplexität nicht überschreiben. Deshalb ist auch hier das verwendet Default-Setting so eingestellt, dass die Komplexität der visuellen Repräsentation durch ein Binning reduziert wird. Abbildung 5.5 veranschaulicht dieses Vorgehen am Beispiel von Scatterplots.

- *Stationäre Geräte*: Diese verfügen über ein übliches Display, was im Allgemeinen zur Ausgabe der visuellen Repräsentationen genügt. Auch die Leistung des Gerätes ist ausreichend, um die visuelle Repräsentation zu erzeugen. Hierbei handelt es sich um den angenommenen Standardfall, und die im Abschnitt 5.3.3 beschriebenen drei Optionen kommen beim Binden der Service-Implementierungen zum Tragen.

Neben den drei genannten Default-Settings, welche die unterschiedlichen Ausgabegeräte berücksichtigen, sind weitere denkbar. An dieser Stelle wurde aber darauf verzichtet, weitere anzugeben, da sie sich aus der derzeitigen Ausstattung des zur Verfügung stehenden Smart Labs nicht ergeben.
Künftige Untersuchungen im GRK MuSAMA werden die Einbindung eines Large-Scale-Diplays einschließen. Hierzu müssen dann auch begleitend neue Standard-Settings für diese Geräteklassen gefunden werden.
Derzeit gibt es in Verbindung mit dem Editor die Option zur Laufzeit, die Default-Settings zu beeinflussen und diese so an besondere Situationen anzupassen. Auch lassen sich durch den Editor weitere dynamische Aspekte behandeln, die nicht durch ein Default-Setting abgedeckt werden können.

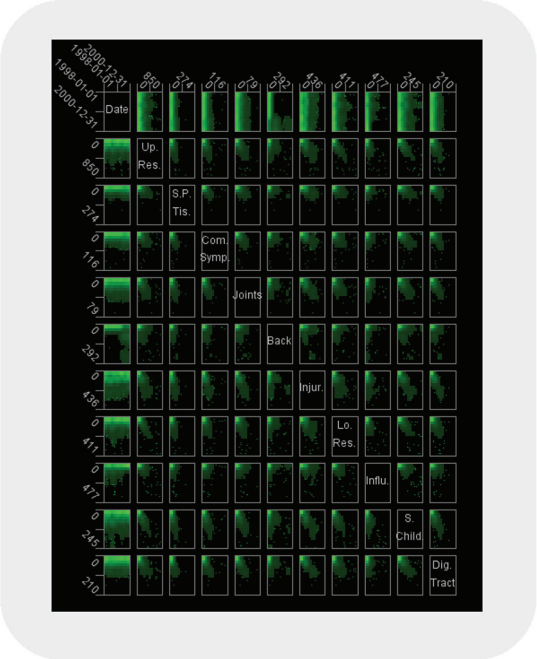

Abbildung 5.5: Erweitertes Binning am Beispiel von Scatterplots. Die Abbildung zeigt einen Gesundheitsdatensatz des Bundeslandes Mecklenburg-Vorpommern über die Jahre 1998 - 2000. Um die Verteilung innerhalb der einzelnen Cluster besser herauszustellen, wird eine dichtebasierte Repräsentation der Daten verwendet.

Konkrete quantitative Werte sind ein offener Forschungsgegenstand und stellen für sich ein komplexes Problem dar.
Durch begleitende Arbeiten des MuSAMA Stipendiaten Axel Radloff werden derzeit usability-Studien durchgeführt, die in einem ersten Schritt die Effektivität von visuellen Repräsentationen am Beispiel von Scatterplots untersuchen und damit das Definieren der Default-Settings absichern.

5.4 Implementierung

Das vorliegende Software-Framework unterteilt sich prinzipiell in die bereits kurz angesprochene Service- und Steuerschicht. Diese sollen nachfolgend näher beschrieben werden.

5.4.1 Service-Schicht

Allgemeine Betrachtung

Wird die Serviceschicht allgemein betrachtet, so kann diese als ein Pool von Services aufgefasst werden, welcher die für das Framework verfügbaren Services umfasst. Jeder Service kapselt dabei einen Operator des DSRM, wobei er ein festgelegtes Service-Interface bedient.
Die gesamte verfügbare Funktionalität eines Service wird ausschließlich über dessen Service-Interface beschrieben. Dadurch wird gewährleistet, das Services, die das gleiche Service-Interface bedienen, austauschbar sind.
Zur konkreten Umsetzung der Services wurde Java verwendet. Die Umsetzung der grundlegenden servicespezifischen Funktionalität wird durch das JINI-Framework gewährleistet. Die Abhängigkeiten vom JINI-Framework sind durch eine zusätzliche Abstraktionsschicht vollständig verborgen. Daher stellt es kein Problem dar, bei Bedarf die von JINI beigesteuerte Funktionalität durch eine andere gleichwertige zu ersetzen.
Wird ein einzelner Service aus softwaretechnischer Sicht betrachtet, so handelt es sich bei ihm um ein selbstständiges Programm, welches über ein Service-Interface angesprochen wird.
An einen Service können über dessen Interface Eingabedaten gesetzt werden. Dabei kann es sich um die zu visualisierenden Daten handeln oder um zusätzliche Steuerparameter.
Die eigentliche Berechnung des Services wird durch ein explizites Kommando durch die Steuerschicht ausgelöst. Dadurch wird sichergestellt, dass alle notwendigen Eingabedaten in den Service geschrieben wurden, bevor dieser mit seinen Berechnungen beginnt. Nach der erfolgreichen Berechnung liegen die Ausgabedaten des Services bereit. Diese werden dann durch die Steuerschicht zu nachgeschalteten Services übertragen.
Abbildung 5.6 zeigt die schematische Sichtweise eines Services im Framework. Zu sehen sind, dass nur Eingabedaten und Ausgabedaten zwischen den Services versendet werden, es erfolgt also immer eine atomare Abarbeitung der Daten. So wird verhindert, dass bei einem wegfallenden Service nur teilweise verarbeitete Daten auftreten und es zu inkonsistenten Zuständen kommt.

Neben der zugrundeliegenden Funktionalität, die je Service-Interface variiert, gibt es weitere allgemeine Funktionen, über die jeder Service verfügt. Sie dienen dazu, die Services über die Steuerschicht zu verwalten und weitere wichtige allgemeine Funktionen zu realisieren.
Konkret verfügt jeder Service über Funktionen, um Informationen über die Leistungsparameter das Gerätes, auf dem der Service ausgeführt wird, an

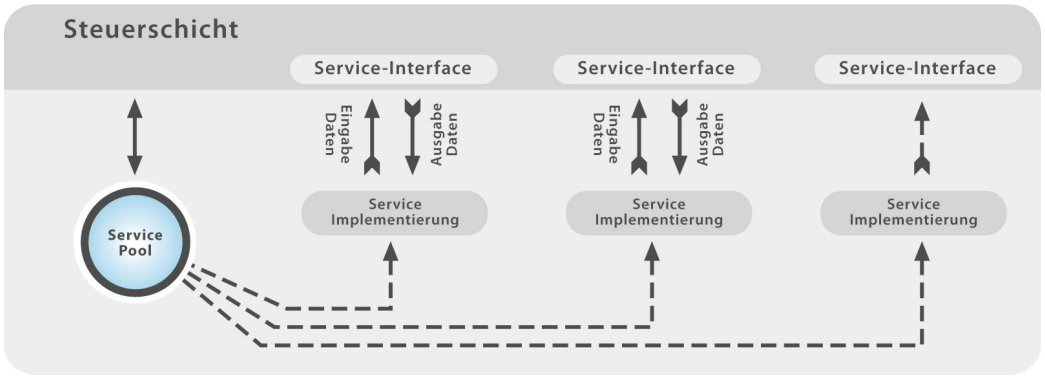

Abbildung 5.6: Schematische Darstellung der Einbindung mehrerer Services in das Framework.

die Steuerschicht zu übermitteln. Zudem verfügt jeder Service über eine allgemeine Schnittstelle zum Schreiben von Eingabedaten in den Service. Das genaue Datenformat und die einzelnen Kommandos zur Unterscheidung der geschriebenen Daten ist durch das implementierte Service-Interface definiert und nicht expliziter Teil der allgemeinen Schnittstelle.
Als Gegenstück verfügen die Services über eine Methode zum Auslesen der Ausgabedaten, damit diese durch die Steuerschicht an den nächsten Service in der Pipeline weitergegeben werden können.

Prinzipiell kann ein Service bei der Konfiguration der Visualisierungspipeline mehrfach verwendet werden. Beispielsweise kann ein Service, der in der Lage ist, Daten im CSV-Format einzulesen zweimal verwendet werden, um diese Daten aus zwei unterschiedlichen Quellen zu laden. Derselbe Fall kann auch bei Verzweigungen der Visualisierungspipeline auftreten, wenn z. B. in jedem Zweig der Service „Farbkodierung„ eingebunden werden muss.
Um dieser Situation gerecht zu werden, wurden die Services mit der Fähigkeit versehen, sich zu replizieren. In der Steuerschicht ist jedem gebundenen Service-Interface genau eine Service zugeordnet. Der Vorteil liegt auf der Hand, die Replikation ermöglicht es, einen Service an verschiedenen Stellen des Frameworks einzubinden, ohne dass Services des gleichen Typs explizit angemeldet werden müssen.
Mehre Services des gleichen Typs bieten zudem den Vorteil, dass berechnete Daten lokal eindeutig zwischengespeichert werden können, ohne dass es zu Überschneidungen bei den Daten der Services kommt. Dadurch ergeben sich Performance-Vorteile, weil neu Berechnungen vermieden werden können, indem auf gespeicherte Daten zurückgegriffen werden kann.
Allerdings muss jetzt auch beachtet werden, dass mit einem ausfallenden

Service alle von ihm lokal gespeicherten Daten mit verlorengehen. Um diese Situation zu berücksichtigen, gibt es zwei Optionen:

- *Option 1*: Die Architektur stellt Sicherungsmechanismen bereit, welche die Daten redundant auf mehreren Geräten im Geräteensemble ablegen und pflegen. Der größte Nachteil hierbei ist der dabei anfallende zusätzliche Kommunikationsaufwand, der während der gesamten Laufzeit des Services anfällt, unabhängig davon, ob der entsprechende Service tatsächlich ausfallen wird.

- *Option 2*: Es wird gewährleistet, dass bei einem ausfallenden Service dieser durch einen identischen Service ersetzt wird. Durch das Konzept der Service-Interfaces ist dies einfach zu realisieren, da jetzt nach einem Service gesucht werden kann, der das gleiche Service-Interface bedient.

  Anschließend wird dieser Services mit den gleichen Eingabedaten versorgt wie der ausgefallene Service, und die Berechnungen der Ausgabedaten werden wiederholt. Das erneute Übertragen der Eingabedaten ist ohne Weiteres möglich, da diese immer noch in den Services lokal zwischengespeichert sind, die diese ursprünglich bereitgestellt haben.
  Nach erfolgter Neuberechnung verhält sich der "Ersatzservice" identisch zum ausgefallenen Service. Auch hierbei fällt ein hoher Kommunikationsaufwand an, und die Neuberechnung der Ausgabedaten durch den Service beansprucht ebenfalls Zeit. Allerdings fallen diese Kosten nur dann an, wenn auch wirklich ein verwendeter Service in einem laufenden System ausfällt. Der verwendete Mechanismus stellt somit eine effiziente Methode dar, bei der sich das System *On-the-Fly* auf einen ausfallenden Service einstellt und diesen ersetzen kann.

Für das Framework wurde die zweite Option realisiert, das Geräteensemble wird zwar als dynamisch angenommen, aber bei dem vorliegenden Szenario eines *Smart-Meeting-Room* nicht derart hoch dynamisch, um den immer anfallende Kommunikationsaufwand der ersten Option zu rechtfertigen.

Services im Framework

In diesem Abschnitt wird auf die Services eingegangen, die im Rahmen dieser Arbeit entstanden sind. Zu jedem Service werden dabei die wichtigsten Eckdaten angegeben, insbesondere das verwendete Service-Interface, die zugehörige Operatorklasse, die Eingabedaten, die Ausgabedaten und eine kurze Beschreibung der Funktionalität.

Das Konzept sieht vor, dass die Service jeweils aus Operatoren des Data-State-Reference-Modells gebildet werden. Der Umsetzung im serviceorientierten Framework ist aber geschuldet, dass auch Services erforderlich sind, die nicht ohne weiteres als ein Operator im DSRM aufgefasst werden können. Bei dieser Klasse von Services handelt es sich um die Hilfsservices, die nachfolgend kurz beschrieben werden.

URL-Auswahl

Operatorklasse : -
Service-Interface: URL.Hilfsservice
Eingabedaten : -
Ausgabedaten : Eine gültige URL
Funktionalität :
Der Service ermöglicht die Auswahl einer URL durch den Benutzer und stellt diese anderen Services zur Verfügung.

Colormapping

Operatorklasse : -
Service-Interface: Color.Hilfsservice
Eingabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
Ausgabedaten : Colormapping Informationen
Funktionalität :
Der Service erlaubt die Auswahl einer Tabellenspalte, die beim Mapping berücksichtigt werden soll. Auf der Basis der Daten dieser Spalte wird jeder Tabellenzeile ein Farbwert zugeordnet. Diese Zuordnung wird in den Colormapping-Informationen abgelegt und weitergegeben.

Single-Colormapping

Operatorklasse : -
Service-Interface: Color.Hilfsservice
Eingabedaten : Analytische Abstraktionen in tabellarischer Form

: Indexmenge
Ausgabedaten : Colormapping Informationen
Funktionalität :
Es wird für alle Tabellenzeilen der gleiche Farbwert in den Colormapping Informationen abgelegt und entsprechend weitergegeben.

Geräteinformationen

Operatorklasse : -
Service-Interface: Device.Hilfsservice
Eingabedaten : URL, welche auf ein Geräteprofil zeigt
Ausgabedaten : Geräteinformationen
Funktionalität :
Der Service läd die Geräteinformationen aus dem übergebenen Geräteprofil und stellt diese dem Framework zur Verfügung.

Indexmenge für analytische Abstraktionen

Operatorklasse : -
Service-Interface: Index.Hilfsservice
Eingabedaten : Analytische Abstraktionen in tabellarischer Form
Ausgabedaten : Indexmenge
Funktionalität :
Diese ordnet jeder Tabellenzeile der analytischen Abstraktionen einen eindeutigen Index zu und gibt diese Indexmenge zurück.

Verschmelzen analytischer Abstraktionen

Operatorklasse : -
Service-Interface: AnalytischeAbstraktionen.Merge.Hilfsservice
Eingabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge

: Analytische Abstraktionen in tabellarischer Form
: Indexmenge
Ausgabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
Funktionalität :

Die beiden übergebenen Mengen an analytischen Abstraktionen werden verschmolzen. Mit den übergebenen Indexmengen wird analog verfahren.

Verschmelzen visueller Abstraktionen

Operatorklasse : -
Service-Interface: VisuelleAbstraktionen.Merge.Hilfsservice
Eingabedaten : Visuelle Abstraktionen
: Indexmenge
: Visuelle Abstraktionen
: Indexmenge
Ausgabedaten : Visuelle Abstraktionen
: Indexmenge
Funktionalität :

Die beiden übergebenen Mengen an visuellen Abstraktionen werden verschmolzen. Mit den übergebenen Indexmengen wird analog verfahren.

Teilen analytischer Abstraktionen

Operatorklasse : -
Service-Interface: AnalytischeAbstraktionen.Splitt.Hilfsservice
Eingabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
: Indexmenge Selektion
Ausgabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
: Analytische Abstraktionen in tabellarischer Form
: Indexmenge
Funktionalität :

Die übergebene Menge an analytischen Abstraktionen wird entsprechend der übergebenen Selectionsindexmenge in zwei Teilmengen zerlegt. Mit der übergebenen Indexmenge wird analog verfahren.

Teilen visueller Abstraktionen

Operatorklasse : -
Service-Interface: VisuelleAbstraktionen.Splitt.Hilfsservice
Eingabedaten : Visuelle Abstraktionen
: Indexmenge
: Indexmenge Selektion
Ausgabedaten : Visuelle Abstraktionen
: Indexmenge
: Visuelle Abstraktionen
: Indexmenge
Funktionalität :

Die übergebene Menge an visuellen Abstraktionen wird entsprechend der übergebenen Selectionsindexmenge in zwei Teilmengen zerlegt. Mit der übergebenen Indexmenge wird analog verfahren.

Data Density Metrik

Operatorklasse : -
Service-Interface: VisuelleAbstraktionen.Metrik.Hilfsservice
Eingabedaten : Visuelle Abstraktionen
: Indexmenge
: Geräteinformationen
Ausgabedaten : Metrik Ausgabe
Funktionalität :

Der Service bestimmt für die übergebenen visuellen Abstraktionen die Data Density (vgl. [Tuf06]) unter Berücksichtigung des Ausgabegerätes. Vom Service wird jeweils ein Soll- und ein Ist-Wert ausgegeben. Der Ist-Wert repräsentiert den aktuellen Wert der Metrik für die übergebenen visuellen Abstraktionen. Der Soll-Wert gibt an, welcher Wert für die Metrik erreicht werden soll, um eine verbesserte Ausgabe auf der visuellen Repräsentation

auf dem Ausgabegerät zu haben.

Nachfolgend werden Services beschrieben, die direkt auf dem Operatoren des Data-State-Reference-Modells beruhen.

CSV-Import

Operatorklasse : Filtering Operator
Service-Interface: URL.Import.Filtering
Eingabedaten : Erwartet eine URL, die auf eine gültige CSV-Datei zeigt
Ausgabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
Funktionalität :

Der Services ist für den Import von Daten, die im CSV-Format vorliegen, verantwortlich. Dabei werden die Daten gelesen und in die vom Framework verwendete allgemeine Datenstruktur konvertiert und bereitgestellt. Zusätzlich wird ein Index erzeugt, der jede Zeile in der Tabelle eindeutig bezeichnet.

Datenbank-Import

Operatorklasse : Filtering Operator
Service-Interface: DB.Import.Filtering
Eingabedaten : Erwartet eine Verbindungsstring zu einer Datenbank
: Eine SQL-Select-Statement zum Einschränken der gesuchten Daten
Ausgabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
Funktionalität :

Der Service ist für den Datenimport aus einer SQL-Datenbank zuständig. Die vom übergebenen SQL-Statement selektierten Daten werden vom Service eingelesen und in die vom Framework verwendete allgemeine Datenstruktur konvertiert und bereitgestellt. Zusätzlich wird ein Index erzeugt, der jede Zeile in der Tabelle eindeutig identifiziert.

Hierarchisches Clustering

Operatorklasse : Operator auf den Analytischen Abstraktionen
Service-Interface: Clustering.AnalytischeAbstraktionen
Eingabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
: Ausgabewerte einer Metrik
Ausgabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
Funktionalität :
Initial wird die gröbste Hierarchieebene ausgegeben. Diese wird später von einer Metrik bewertet, die Metrik gibt eine Rückmeldung mit Soll- und Ist-Wert an diesen Service. Anhand der Differenz zwischen Soll- und Ist-Wert wird entschieden, ob die nächste Ebene ausgegeben wird. Diese wird wieder von der Metrik bewertet. Erst wenn der Betrag der Differenz zwischen Soll- und Ist-Wert am kleinsten ist, wird keine weitere Ebene ausgegeben. Zusätzlich wird die Indexmenge entsprechend angepasst und ausgegeben.

Vector-Daten-Import

Operatorklasse : Mapping Operator
Service-Interface: Vector.Import.Mapping
Eingabedaten : Erwartet wird eine URL, die auf eine Datei
mit grafischen Vector-Daten zeigt
Ausgabedaten : Visuelle Abstraktionen
: Indexmenge
Funktionalität :
Die Vector-Daten werden vom Service geladen und in das vom Framework intern verwendete Datenformat zur Beschreibung von visuellen Primitiven konvertiert und bereitgestellt. Zusätzlich wird ein Index erzeugt, der jedes visuelle Primitiv eindeutig identifiziert.

Bilddaten-Import

Operatorklasse : Mapping Operator
Service-Interface: Pixel.Import.Mapping

Eingabedaten : Erwartet wird eine URL, die auf eine Datei mit Pixelbilddaten zeigt
Ausgabedaten : Visualle Abstraktionen
: Indexmenge
Funktionalität :

Die Pixeldaten werden vom Service geladen und in das vom Framework intern verwendeten Datenformat zur Beschreibung von visuellen Primitiven konvertiert und bereitgestellt. Zusätzlich wird ein Index erzeugt, der jedes visuelle Primitiv eindeutig identifiziert.

PDF-Import

Operatorklasse : Mapping Operator
Service-Interface: Vector.Import.Mapping
Eingabedaten : Erwartet wird eine URL, die auf eine PDF-Datei zeigt
Ausgabedaten : Visuelle Abstraktionen
: Indexmenge
Funktionalität :

Das PDF wird geladen und jede Seite in ihre Bestandteile zerlegt sprich Text, Vektordaten und Bilder. Diese werden entsprechend in visuelle Primitive umgeformt, die als visuelle Abstraktionen bereitgestellt werden. Zusätzlich wird ein Index erzeugt, der jedes visuelle Primitiv eindeutig identifiziert.

Parallele Koordinaten

Operatorklasse : Mapping Operator
Service-Interface: Technik.Mapping
Eingabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
: Geräteinformationen
: Colormapping Informationen
Ausgabedaten : Visuelle Abstraktionen
: Indexmenge
Funktionalität :

Die in den analytischen Abstraktionen vorliegenden Daten werden entsprechend der Visualisierungstechnik auf Kantenzüge gemappt. Dabei werden die Informationen aus dem Colormapping entsprechend berücksichtigt. Zusätzlich wird die Indexmenge um die Elemente erweitert, die zur Identifikation von Skalen und Achsen erforderlich sind.

Scatterplots

Operatorklasse : Mapping Operator
Service-Interface: Technik.Mapping
Eingabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
: Geräteinformationen
: Colormapping Informationen
Ausgabedaten : Visuelle Abstraktionen
: Indexmenge
Funktionalität :

Die in den analytischen Abstraktionen vorliegenden Daten werden entsprechend der Visualisierungstechnik auf Punktmatrizen gemappt. Dabei werden die Informationen aus dem Colormapping entsprechend berücksichtigt. Zusätzlich wird die Indexmenge um die Elemente erweitert, die zur Identifikation von Skalen und Achsen erforderlich sind.

Parallele Koordinaten mit Binning

Operatorklasse : Mapping Operator
Service-Interface: Adaptiv.Technik.Mapping
Eingabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
: Geräteinformationen
: Colormapping Informationen
Ausgabedaten : Visuelle Abstraktionen
: Indexmenge
Funktionalität :

Die in den analytischen Abstraktionen vorliegenden Daten werden entsprechend der Visualisierungstechnik auf Kantenzüge gemappt. Dabei werden die Informationen aus dem Colormapping entsprechend berücksichtigt. Anschließend wird ein Binning durchgeführt. Da das Binning nicht unabhängig von der Visualisierungstechnik durchgeführt werden kann, wurde beides in einem Service zusammengefasst. Zusätzlich wird die Indexmenge um die Elemente erweitert, die zur Identifikation von Skalen und Achsen erforderlich sind.

Scatterplots mit Binning

Operatorklasse : Mapping Operator
Service-Interface: Adaptiv.Technik.Mapping
Eingabedaten : Analytische Abstraktionen in tabellarischer Form
: Indexmenge
: Geräteinformationen
: Colormapping Informationen
Ausgabedaten : Visuelle Abstraktionen
: Indexmenge
Funktionalität :

Die in den analytischen Abstraktionen vorliegenden Daten werden entsprechend der Visualisierungstechnik auf Punktmatrizen gemappt. Dabei werden die Informationen aus dem Colormapping entsprechend berücksichtigt. Anschließend wird ein Binning durchgeführt. Da das Binning nicht unabhängig von der Visualisierungstechnik durchgeführt werden kann, wurde beides in einem Service zusammengefasst. Zusätzlich wird die Indexmenge um die Elemente erweitert, die zur Identifikation von Skalen und Achsen erforderlich sind.

Halo-Service

Operatorklasse : Operator auf den visuellen Abstraktionen
Service-Interface: Highlight.VisuelleAbstraktionen
Eingabedaten : Visuelle Abstraktionen mit visuellen Primitiven
: Indexmenge
: IndexmengeSelektion

Ausgabedaten : Visuelle Abstraktionen
: Indexmenge
Funktionalität :

Dieser Service hebt die durch die Indexmenge bezeichneten visuellen Primitive durch einen Halo-Effekt(vgl. [LS08]) hervor. Die Halos werden durch visuelle Primitive des Typs Picture beschrieben, ihnen wird entsprechend der Index des visuellen Primitives zugeordnet, welchen sie hervorheben.

Transformationsservice

Operatorklasse : Operator auf den visuellen Abstraktionen
Service-Interface: Transform.VisuelleAbstraktionen
Eingabedaten : Visuelle Abstraktionen mit visuellen Primitiven
: Indexmenge
: Geräteinformationen
Ausgabedaten : Visuelle Abstraktionen
: Indexmenge
Funktionalität :

Der Service bestimmt die maximale Ausdehnung der Menge der übergebenen visuellen Primitive und führt anschließend eine Skalierung durch, so dass die Primitive unter voller Ausnutzung der verfügbaren Displayfläche dargestellt werden können.

Color-Transformationsservice

Operatorklasse : Operator auf den visuellen Abstraktionen
Service-Interface: Transform.VisuelleAbstraktionen
Eingabedaten : Visuelle Abstraktionen mit visuellen Primitiven
: Indexmenge
: Geräteinformationen
Ausgabedaten : Visuelle Abstraktionen
: Indexmenge
Funktionalität :

Der Service bestimmt die maximale Ausdehnung der Menge der übergebenen visuellen Primitive und führt anschließend eine Skalierung durch, so dass die Primitive unter voller Ausnutzung der verfügbaren Displayfläche dargestellt werden können. Zusätzlich werden die verwendeten Farben auf die Farben des Gerätes angepasst. Kann das Ausgabegerät nur Graustufen ausgeben, wird der in [GOTG05] vorgestellte Algorithmus eingesetzt (vgl. Abbildung 5.7).

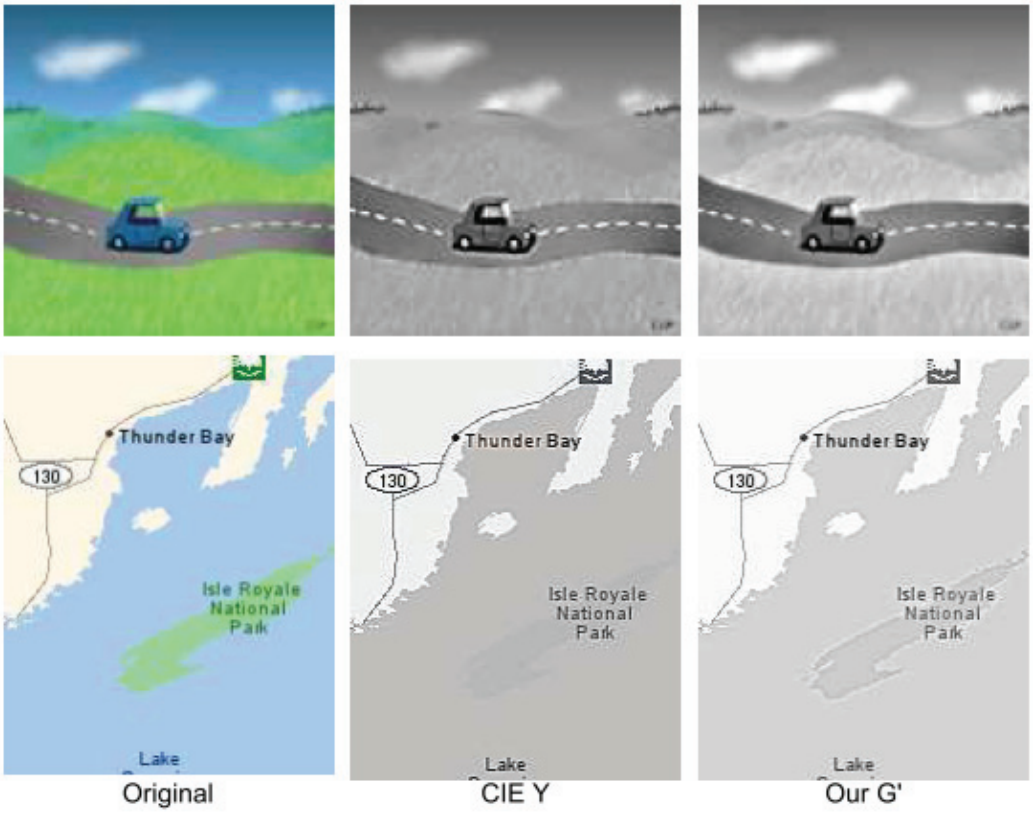

Abbildung 5.7: Angepasste Farb- zu Grauwert-Konvertierung, bei der unterschiedliche Farben auch auf unterschiedliche Grauwerte abgebildet werden. Aus [GOTG05]

Standard-Render-Service

Operatorklasse : Rendering-Operator
Service-Interface: Renderer

Eingabedaten : Visuelle Abstraktionen mit visuellen Primitiven
: Indexmenge
: Geräteinformationen
Ausgabedaten : Pixelausgabe der gerenderten visuellen Primitive auf dem verbundenen Ausgabegerät
Funktionalität :
Der Service gibt die visuellen Primitive auf dem Ausgabegerät aus. Dazu werden die übergebenen visuellen Primitive durch Java 2D-Zeichenfunktionen dargestellt.

Verwendung der Services Aus der Verkettung mehrerer der genannten Services können unterschiedliche visuellen Repräsentationen erzeugt werden. Nachfolgend sollen einige Beispiele gezeigt werden. Dabei wird jeweils ein Schema der verketteten Service-Interfaces gezeigt und zudem ein Screendump der visuellen Repräsentation auf dem Ausgabegerät.
Im ersten Beispiel wird eine herkömmliche *Parallele-Koordinatendarstellung* vorgestellt. Bei der Erstellung wird das in Abbildung 5.8 dargestellte Pipeline-Template verwendet. Die Ausgabe des Frameworks ist in Abbildung 5.9 dargestellt. Erst durch die Verkettung von mehreren Services kann die Ausgabe erzeugt werden. Im zweiten Beispiel wird der gleiche Datensatz durch eine

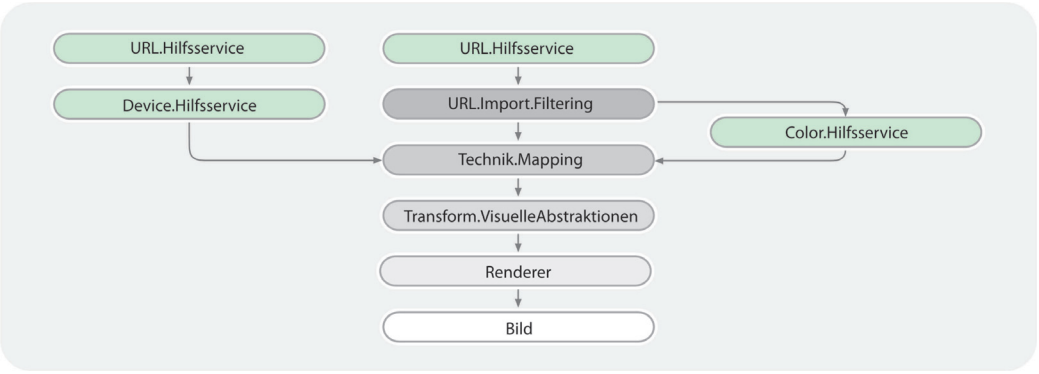

Abbildung 5.8: Pipeline-Template mit den verknüpften Service-Interfaces.

Scatterplot-Matrix visualisiert. Das verwendete Pipeline-Template ist dasselbe wie in Abbildung 5.8. Die Ausgabe des Frameworks ist in Abbildung 5.10 zu sehen. Für die unterschiedliche Ausgabe ist im Wesentlichen eine andere Entscheidung beim Binding verantwortlich (vgl. 5.3.3).
Im dritten Beispiel soll die Ausgabe der Daten auf einem kleineren Ausgabe-

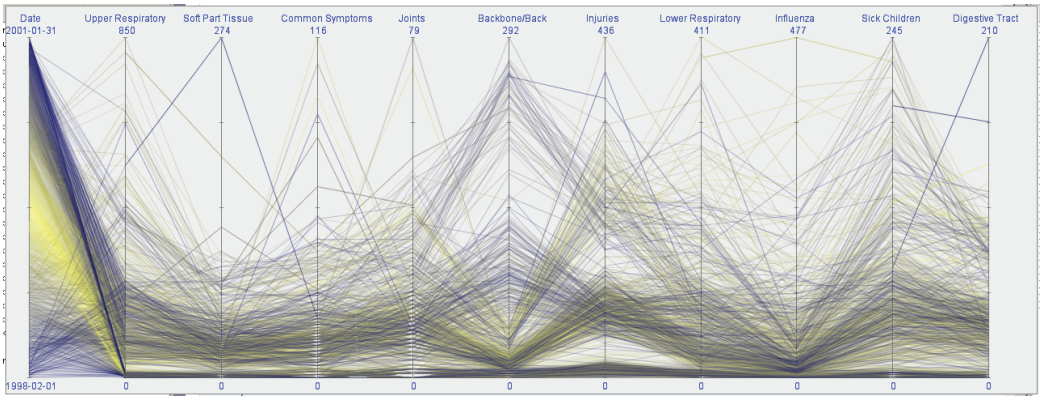

Abbildung 5.9: Parallele Koordinaten mit unterschiedlichen Krankheitsfällen aus einem AOK-Datensatz.

gerät erfolgen. Hierfür wird ein leicht angepasstes Pipeline-Template verwendet, welches in Abbildung 5.11 dargestellt ist. Entsprechend verändert sich die Ausgabe (Abbildung 5.12) der Scatterplot-Matrix, weil zusätzlich jetzt eine adaptive Technik durch das Service-Interface gefordert wird und somit das Binding eine andere Entscheidung treffen muss.

Dynamische Serviceschicht Die Serviceschicht wird zur Laufzeit dynamisch aufgebaut. Die hier vorgestellten Services gehören zur Grundfunktionalität des Frameworks. Zur Laufzeit werden die Service mit dem Framework zusammen aktiviert und stehen sofort zur Verfügung. Das Gerät, auf dem die Steuerschicht ausgeführt wird, stellt gleichzeitig auch die vorgestellten grundlegenden Services bereit. Ist das Framework in einem *Smart-Meeting-Room* noch nicht aktiv, kann dieses von einem Server im Internet heruntergeladen werden.
Bei Bedarf können weitere Services auch auf anderen Geräten im *Smart-Meeting-Room* gestartet werden. Zusätzlich dazu sind für das Framework die Services sichtbar, die von den Benutzern mit in den *Smart-Meeting-Room* gebracht werden.
Das Anmelden von Services über das Webinterface ist in Abbildung 5.13 dargestellt.

5.4.2 Steuerschicht

Die Steuerschicht ist in ein separates Programm gekapselt, es ist demzufolge erforderlich, dass dieses Programm mindestens einmal im *Smart-Meeting-*

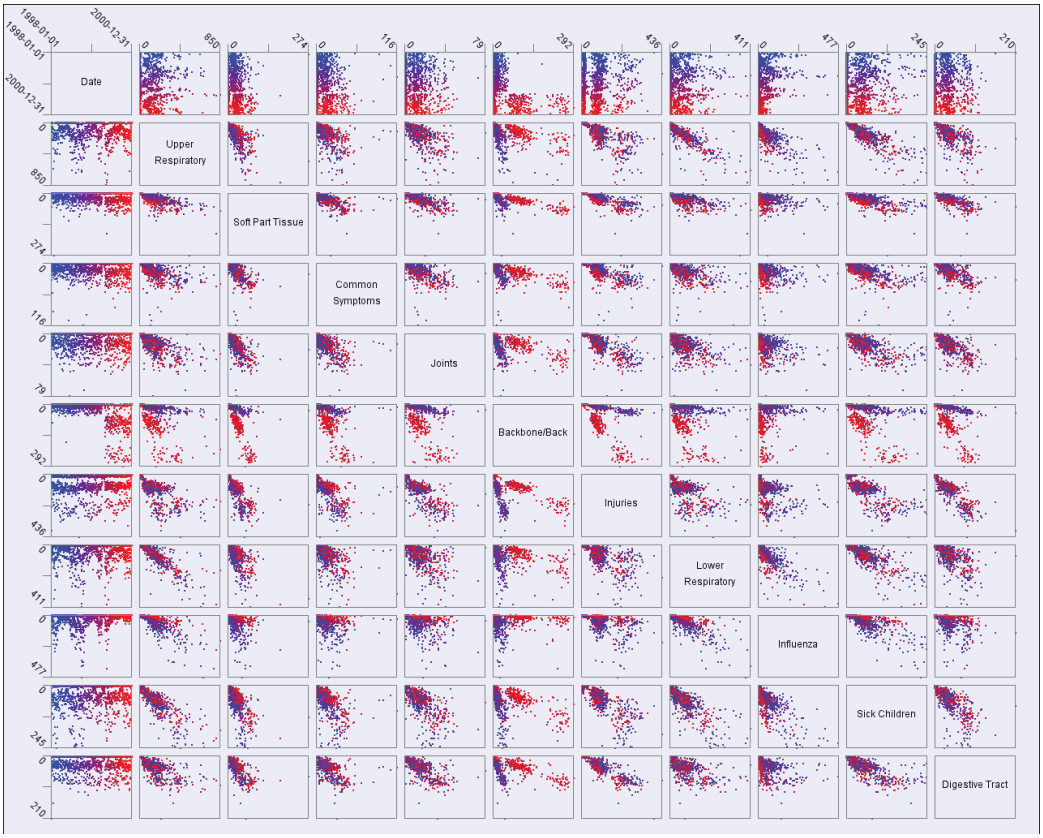

Abbildung 5.10: Scatterplot durch den unterschiedliche Krankheitsfälle zueinander in Bezug gesetzt werden.

Room gestartet wird, bevor die Steuerschicht arbeitet. Üblicherweise ist dies Aufgabe des Visualisierungsautors.
Entsprechend der in Abschnitt 5.3.2 vorgestellten Rollenverteilung wird ein Pipeline-Template ausgewählt, welches das Ziel der Betrachter unterstützt und welches prinzipiell in der Lage ist, die zu visualisierenden Daten zu verarbeiten. Hierfür stehen unterschiedliche Pipeline-Tempates zur Verfügung, welche den Import von unterschiedlichen Daten und die Erzeugung unterschiedlicher visueller Repräsentationen unterstützen.
Eine vollständige automatische Auswahl des Pipeline-Templates ist aufgrund der möglichen Heterogenität der Daten und der Ziele aktuell nicht Teil der Implementierung.

Das Geräteensemble im *Smart-Meeting-Room* ist dynamisch. Neue Geräte im Ensemble werden nach Anmeldung von mindestens einer Service-Implementierung auf diesen Geräten gefunden.

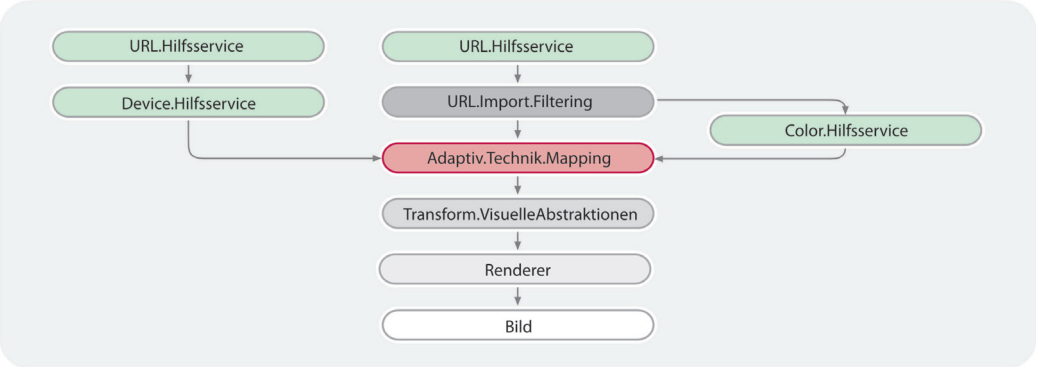

Abbildung 5.11: Pipeline-Template mit den verknüpften Service-Interfaces unter Verwendung einer adaptiven Technik.

Alle verfügbaren Services im *Smart-Meeting-Room* werden durch die Steuerschicht überwacht, fallen alle Service-Implementierungen auf einem Gerät aus, so geht die Steuerschicht davon aus, dass das betreffende Gerät den *Smart-Meeting-Room* verlassen hat und nicht mehr zur Verfügung steht.
Die Überwachung der Services erfolgt durch ein Testsignal, welches in einem festen Intervall von der Steuerschicht an jeden Service gesendet wird. Erfolgt keine Antwort auf dieses Testsignal wird der Service als nicht mehr vorhanden eingestuft. Dieses Vorgehen ermittelt indirekt, welche Geräte im Ensemble vorhanden sind. Dabei ist zu beachten, dass ein Gerät nicht gefunden wird, wenn auf diesem keine Service-Implementierungen ausgeführt werden.
Dieses Vorgehen stellt sicher, dass nicht die Steuerschicht die Kontrolle über die einzelnen Geräte im Ensemble hat und automatisch beliebige Service-Implementierungen auf diesen Geräten zur Ausführung bringen kann, sondern dass der Benutzer jederzeit die Kontrolle darüber hat, ob er sein Gerät in das der Steuerschicht zur Verfügung stehendem Geräteensemble einbringt oder nicht.
Das bedeutet, dass der Benutzer sich bewusst entscheiden kann, welche Service-Implementierungen auf seinem Gerät ausgeführt werden, indem er diese selbst anmeldet.
Möchte sich ein Benutzer aus dem Geräteensemble zurückziehen, auf welches das Framework zurückgreift, so deaktiviert er alle Service-Implementierungen, die auf seinem Gerät aktuell ausgeführt werden. Aus Sicht der Steuerschicht wird dieses Gerät dann “unsichtbar“. Die Steuerschicht wird bei diesem Szenario versuchen, auf andere Service-Implementierungen auf anderen Geräten auszuweichen.

Beim Binding der Service-Implementierungen an die Service-Interfaces kom-

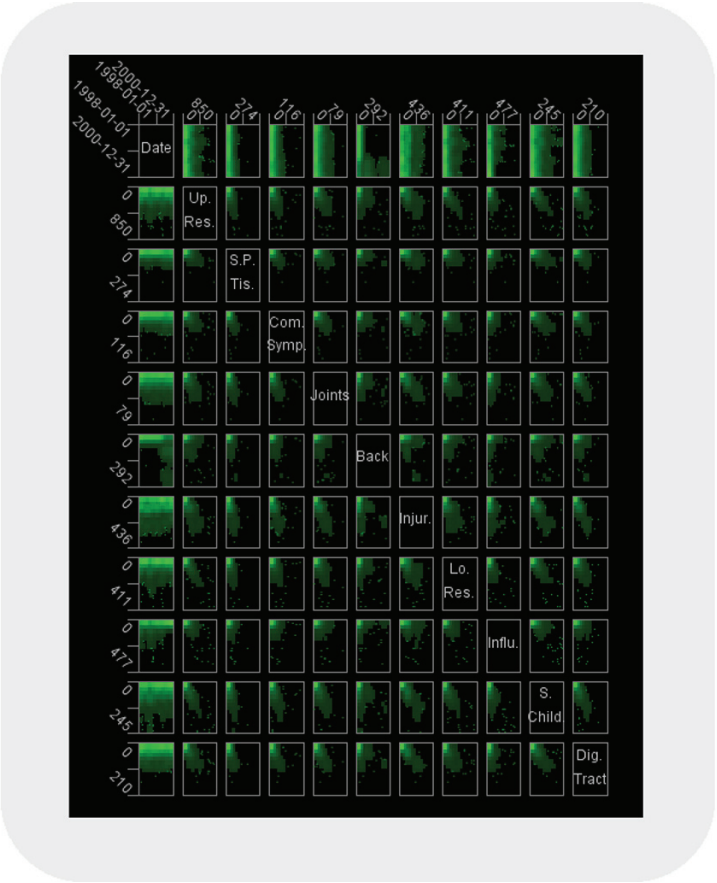

Abbildung 5.12: Erweitertes Binning am Beispiel von Scatterplots. Um die Verteilung innerhalb der einzelnen Cluster besser herauszustellen, wird eine Dichte-basierte Repräsentation der Daten verwendet.

men die Optionen zum Tragen, die im Abschnitt 5.3.3 diskutiert wurden.
Bei einem benutzerorientierten Vorgehen werden also erst die passenden Service-Implementierungen gesucht, die sich auf dem Gerät des Benutzers befinden. Werden hier mehrere Service-Implementierungen gefunden, wird der Visualisierungsautor konsultiert, wird keine Service-Implementierung gefunden, wird die Suche auf alle Geräte im *Smart-Meeting-Room* ausgeweitet.
Bei einem geräteorientierten Vorgehen wird analog vorgegangen. Statt aber die Service-Implementierungen eines Benutzers als Erstes zu überprüfen, werden stattdessen die Service-Implementierungen des Gerätes überprüft, auf dem aktuell die meisten Services-Implementierungen des Pipeline-Templates ausgeführt werden.
Die im Abschnitt 5.3.3 beschriebene dritte Option hat zum jetzigen Zeitpunkt noch keinen Eingang in die Steuerschicht gefunden.

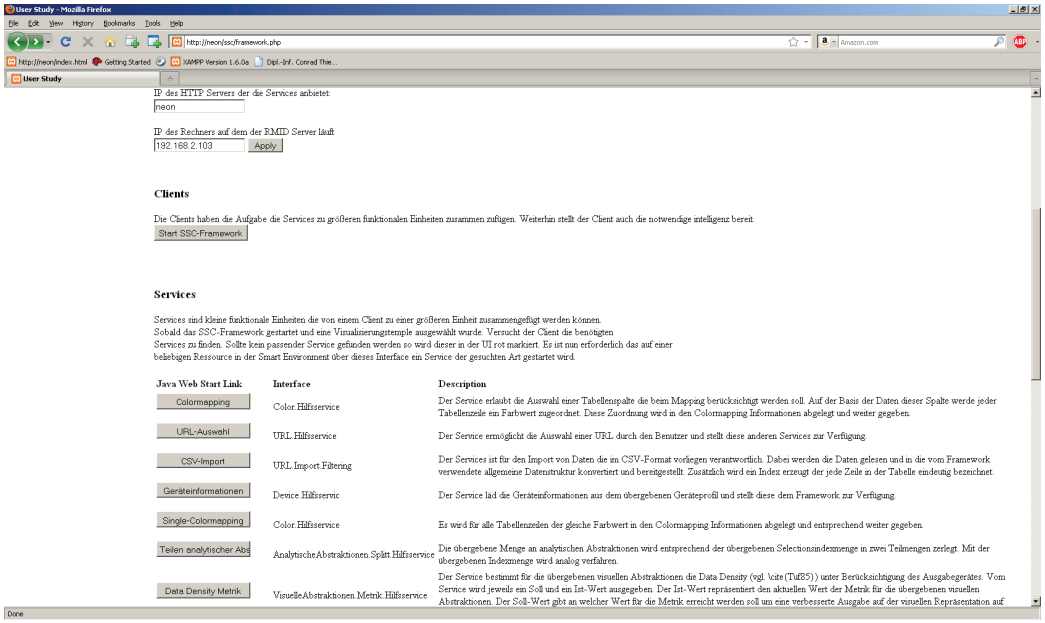

Abbildung 5.13: Das Webinterface zum Anmelden von Services am Framework.

Eine weitere wichtige Aufgabe der Steuerschicht ist die Synchronisation der Services.
Bei der Erzeugung einer visuellen Repräsentation von Daten wird in der Regel nach einem Pipeline-Modell vorgegangen. Es erfolgt demnach eine schrittweise Abarbeitung der einzelnen Stufen. Services stellen nun autonome Programme dar, diese können zudem auf unterschiedlichen Geräten ausgeführt werden. Somit ist es ohne weiteres möglich, dass mehrere Service gleichzeitig ihre Daten verarbeiten.
Durch das verwendete Pipeline-Template ist eine Verarbeitungsreihenfolge der Services gegeben. Dieser Zusammenhang ist nur für das aktuelle Template gültig und kann mit einem anderen Template anders gelagert sein.
Die Steuerschicht hat die Aufgabe, die Ausführung der Services zu synchronisieren. Andernfalls kann es zu schwerwiegenden Problemen in der Verarbeitung kommen. Als erste Lösung wurde die asynchrone Abarbeitung der Services unterbunden. Stattdessen wird eine synchrone Abarbeitung der Services sichergestellt.
Um dies zu gewährleisten, läuft die gesamte Kommunikation über die Steuerschicht, diese wird dadurch in die Lage versetzt, direkt zu bestimmen, wann exakt ein Service seine Eingangsdaten verarbeitet und die Ausgangsdaten bereitstellt. So kann sichergestellt werden, dass eine Abarbeitung eines Services nicht stattfinden kann, wenn noch nicht alle Eingangsdaten vorliegen.

Folglich Abschnitt 2.3.3 handelt es sich demnach um eine indirekte Kommunikation der Services.

Muss die visuelle Repräsentation auf einem anderen Ausgabegerät dargestellt werden, ist eine Adaption der visuellen Repräsentation an das neue Ausgabegerät erforderlich.
Verändert sich die Geräteklasse, wird eine Adaption durchgeführt. Konkret bedeutet dies, dass bei der Geräteklasse *Ultra Mobil* das Service-Interface *Technik.Mapping* durch die Steuerschicht in das Service-Interface *Adaptiv. Technik.Mapping* umgewandelt wird. Weiterhin wird per default das Service-Interface *Clustering.AnalytischeAbstraktionen* auf der Stufe der analytischen Abstraktionen zwischengeschaltet, und die Eingabedaten des Renderer Service-Interface werden zusätzlich zum *VisuelleAbstraktionen.Metrik.Hilfsservice* Service-Interface weitergeleitet. Entsprechend wird durch die Steuerschicht die Metrik mit dem Clustering-Interface verbunden. In einem letzen Schritt wird sichergestellt, dass die erzeugte visuelle Repräsentation auf das *Ultra Mobil* übertragen wird.
Anschließend liegt ein adaptiertes Pipeline-Template vor, entsprechend muss das Binding der Service-Implementierung neu durchgeführt werden, da nicht mehr alle Service-Implementierungen zu den verwendeten Service-Interfaces passen.
Für die Geräteklasse *Mobil* wird analog verfahren, nur mit dem Unterschied, dass per default auf ein Clustering im Datenraum verzichtet wird und entsprechend auch nicht das *VisuelleAbstraktionen.Metrik.Hilfsservice* Interface erforderlich ist. Zusätzlich wird die visuelle Repräsentation direkt auf dem Ausgabegerät erzeugt und auch dort ausgegeben.
Im Fall der Geräteklasse *Stationär* werden keine zusätzlichen Service-Interfaces hinzugefügt.
Im Fall, dass sich die Geräteklasse des Ausgabegerätes in die entgegengesetzte Richtung verändert, sprich von *Ultra Mobil* zu *Mobil* bzw. von *Mobil* zu *Stationär* werden die Service-Interfaces, die per default nicht in diesem Template zur Adaption verwendet werden, entsprechend entfernt.

Aktuell noch nicht in das Framework eingebettet, ist der Prototyp zur progressiven Informationsdarstellung. Dieser lässt eine weitere Adaption der visuellen Repräsentation zu (vgl. Abschnitt 4.5). Zudem ist diese Form der Adaption sehr flexibel und kann für alle drei der genannten Geräteklassen angewendet werden.
In Abbildung 5.14 ist die Ausgabe der gleichen visuellen Repräsentation auf drei unterschiedlichen Klassen von Ausgabegeräten dargestellt.
Zur lokalen Adaption der visuellen Repräsentation wurden im Abschnitt 4.4

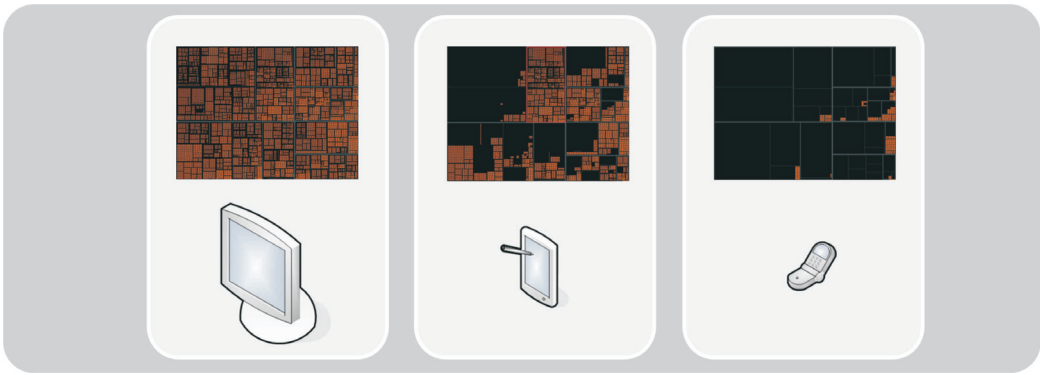

Abbildung 5.14: Resultate einer progressiven Informationsdarstellung von hierarchischen Daten auf drei unterschiedlichen Geräteklassen.

die *intelligenten Linsen* eingeführt. Der implementierte Prototyp ist ebenfalls noch nicht in das Framework eingebettet. Auf das Potenzial, welches sich durch die *intelligenten Linsen* ergibt, wurde bereits im Abschnitt 4.4 eingegangen. Beispiele sind ebenfalls an dieser Stelle angegeben worden.
Zur Implementierung der *intelligenten Linsen* bleibt anzumerken, dass deren Framework im Wesentlichen die Datenstrukturen des hier vorliegenden Framework verwenden. Somit ist der Aufwand zur Integration der *intelligenten Linsen* in dieses Framework überschaubar.

5.4.3 Das Framework in der Praxis

Bei der Umsetzung des Framework-Konzepts wurde bewusst auf Java gesetzt. Dadurch wird eine Portierbarkeit auf andere Plattformen wie Linux oder MacOS bzw. Windows erheblich vereinfacht. Bei der Umsetzung wurde konsequent auf die Java-Web-Start-Technologie gesetzt. Diese erlaubt es, eine Java-Anwendung durch einen einfachen Klick in einem internetfähigen Browser herunterzuladen und auszuführen. Hierfür ist nur der besagte Browser, eine Internetverbindung und ein installiertes Java JDK erforderlich. Statt das Framework im Internet bereitzustellen, kann es stattdessen auch auf einem Webserver in einem lokalen Netzwerk bereitgestellt werden. In der Praxis bedeutet dies, dass neue Benutzer in einem *Smart-Meeting-Room*, die ihre Geräte in das Geräteensemble einbringen wollen, nur eine Internetseite bzw. eine Seite im lokalen Netzwerk ansteuern müssen und die gewünschten Services per Klick aktivieren. Alle weiteren Schritte werden durch das aktuell laufende Framework übernommen. Ist kein aktives Framework vorhanden, kann dieses ebenfalls über Java-Web-Start mittels eines einfachen Mausklicks

gestartet werden. Durch das Ausnutzen der genannten Technologien wird eine Integration eines neuen Benutzers extrem vereinfacht und scheitert nicht bereits an einer aufwändigen Installation einer Software.

5.5 Diskussion

Das hier vorgestellte Konzept einer servicebasierten Visualisierung zur Adaption an multiple, dynamische, heterogene Ausgabegeräte zeigt Parallelen zu dem Ansatz von Zudilova-Seinstra und Yang [ZSY05] (vgl. auch 2.3.3).
Allerdings geht es bei wesentlichen Punkten über die Konzepte in der Literatur hinaus. Während die Architektur von Zudilova-Seinstra und Yang ebenfalls einzelne Stufen als Services realisiert, sieht das hier vorliegende Konzept vor, einzelne Operatoren des Data-State-Reference-Models als Services zu realisieren. Damit baut die hier vorgestellte Architektur auf ein in der Informationsvisualisierung bewährtes Modell auf und erlaubt gleichzeitig eine hohe Wiederverwendbarkeit der einzelnen Services.

Das Framework wird im Wesentlichen in zwei Schichten unterteilt. Zum einen die Serviceschicht, diese enthält alle Services, die vom Framework verwendet werden. Da den einzelnen Services jeweils der Kontext fehlt, um autonom eine Adaption der visuellen Repräsentation vornehmen zu können, ist eine Steuerschicht erforderlich, welche die einzelnen Services steuert.
Durch diese strikte Trennung wird die notwendige Intelligenz bei der Adaption in der Steuerschicht gebündelt, die Services können dadurch einfacher und allgemeiner gehalten werden.

Bei einem serviceorientiertem Ansatz ist ein Binding der Services notwendig, die für den Erzeugungsprozess der visuellen Repräsentation verwendet werden sollen. Prinzipiell wird beim vorliegenden Konzept eine explorative Komposition (vgl. 2.3.3) der Services vorgenommen. Dadurch wird eine höhere Flexibilität im Vergleich zu einer statischen Kompositionen erreicht, bei denen Services fest zugeordnet werden.
Zudem können beim Binding sehr unterschiedliche Strategien angewendet werden (vgl. 5.3.3). Abhängig von der verwendeten Strategie lässt sich bereits durch ein geschicktes Binding von Services auf unterschiedliche Rahmenbedingungen reagieren, woraus sich ein weiterer Vorteil einer serviceorienierten Architektur ergibt.
Die interaktive Adaption der verwendeten Visualisierungspipeline stellt einen weiteren wichtigen Punkt zur Adaption der visuellen Repräsentation in ei-

nem *Smart-Meeting-Room* dar. Dadurch kann sichergestellt werden, dass in jedem Fall eine gute Anpassung an die vorliegende Gerätesituation erfolgt. Zudem bietet die interaktive Adaption die Option, dass automatisch generierte Entscheidungen des Frameworks durch den Benutzer übersteuert werden können.
Trotz des Nutzens stellen die vorgestellten Konzepte nur einen ersten Grundstock an Methoden zur Adaption einer Informationsvisualisierung in *Smart-Meeting-Rooms* dar. In Folgearbeiten müssen diese Konzepte weiter ausgebaut werden.

Kapitel 6

Zusammenfassung und Ausblick

6.1 Zusammenfassung

Informationsvisualisierung in Multi-Display-Umgebungen mit heterogenen sich dynamisch verändernden Ausgabegeräten ist ein offener Forschungsgegenstand, für den es in der gängigen Literatur noch keine Ansätze gibt. Die Skalierung einer Präsentation, so dass sie sowohl auf einer Displaywall als auch auf einem Smart Phone dargestellt werden kann, wurde als Herausforderung und offener Forschungspunkt auf dem Dagstuhl-Seminar 10241 Information Visualization (vgl. [DS110]) benannt.
Das Ergebniss dieser Promotion ist vor allem die Identifikation wichtiger Probleme in diesem Umfeld sowie die Entwicklung erster exemplarischer Lösungen. Diese sollen im Folgenden kurz zusammengefasst werden.

Das Ziel der vorliegenden Arbeit war die Entwicklung von Konzepten für eine geeignete Informationsdarstellung in *Smart-ad-hoc-Environments* unter besonderer Berücksichtigung unterschiedlicher, sich dynamisch verändernder Ausgabegeräte.
Da die dynamische Veränderungen im Geräteensemble einer der *Smart-ad-hoc-Environment* jederzeit auftreten können, muss dabei Erzeugung und Anpassung der Informationsdarstellung *On-the-Fly* erfolgen.
In dieser Arbeit wurden insbesondere drei Schwerpunkte untersucht und hierfür exemplarisch Lösungen bereitgestellt:

1. Anpassung von Informationsdarstellungen an unterschiedliche Ausgabegeräte

2. Auswahl einer geeigneten *Adaptionsstrategie* in Abhängigkeit der aktuellen Situation

3. Entwurf und Umsetzung eines flexiblen, erweiterbaren Software-Frameworks zur dynamischen Informationsdarstellung in *Smart-Meeting-Room*.

In der vorliegenden Arbeit erfolgt die Anpassung der Informationsdarstellung vor allem gerätebasiert, das heißt bezogen auf die Eigenschaften des aktuellen Ausgabegerätes. Die gerätebasierte Adaption von Informationsvisualisierungen kann auf verschiedene Art und Weise erfolgen. Deshalb wurde in einem ersten Schritt ein Schema zur Einordnung möglicher *Adaptionsmechanismen* entwickelt. Dabei wurde unterschieden zwischen:

- Adaption auf Bildebene vs. Adaption auf Prozessebene
- lokale vs. globale Adaption
- Adaption vs. Progression

Adaption auf der Bildebene Liegt bereits eine visuelle Repräsentation vor, und man kann keinen Einfluss mehr auf die Bilderzeugung nehmen, muss die Adaption auf der Bildebene erfolgen.
Im Rahmen dieser Arbeit wurde hierfür die *inhaltbasierte Skalierung* fotorealistischer Bilder aus [AS07] zur Grundlage genommen. Dabei erfolgt eine Skalierung der einzelnen Bildbereiche in Abhängigkeit davon, wie wichtig der entsprechende Bereich ist. Das heißt, unwichtige Informationen werden aus dem Bild entfernt und schaffen so Platz für wichtige Informationen, bei denen deshalb keine Skalierung erforderlich ist. Die Wichtigkeit eines Bildteils wird dabei anhand einer geeigneten Energiefunktion, z. B. anhand des Informationsgehaltes des entsprechenden Bildteils entschieden.
Die neue Idee in dieser Arbeit ist es, die *inhaltsbasierte Skalierung* auch auf visuelle Repräsentationen von Daten und damit für die Informationsdarstellung anzuwenden.
Es konnte gezeigt werden, dass die *inhaltsbasierte Skalierung* mit der Einschränkung, dass in der Darstellung keine Informationen auf die Position abgebildet werden und unter Verwendung der Entropie als Energiefunktion akzeptable Ergebnisse liefert.

Adaption auf der Prozessebene Muss die visuelle Repräsentation erst erzeugt werden, so kann eine Adaption auf allen Stufen des Erzeugungsprozesses vorgenommen werden, also im *Datenraum*, im *Darstellungsraum* und auch beim *Mapping*.

Gerade wenn eine visuelle Repräsentation auf kleinen Ausgabegeräten dargestellt werden muss, ist es oft nicht möglich, alle Informationen darzustellen, so dass eine Reduzierung der Daten im *Datenraum* erfolgen muss. In der Informationsvisualisierung werden hierfür insbesondere Clustering- oder Information-Hiding-Verfahren eingesetzt. Da man ohne zusätzliche Vorgaben nicht ohne weiteres Informationen ausblenden kann, wurde im Rahmen dieser Arbeit exemplarisch das hierarchische Clustern dafür eingesetzt. Damit lassen sich auch auf kleinen Geräten und in unterschiedlicher Genauigkeit wichtige Charakteristika der zu Grunde liegenden Daten anzeigen.
Als Beispiel für eine Adaption beim *Mapping-Schritt* wurde die Helligkeitsadaption für Scatterplot-Darstellungen untersucht. Als Erweiterung wurde aber statt der linearen Farbskala eine logarithmische Farbskala verwendet. Dadurch konnte die Aussagekraft der visuellen Repräsentation erhöht werden (vgl. 4.3.2).
Für die Adaption im *Darstellungsraum* wurde exemplarisch das Binning eingesetzt und erweitert. Insbesondere wurden die beiden folgenden Verbesserungen bei der Darstellung geclusterter Daten eingeführt:

- Die Auflösung der Bins wird der Displaygröße angepasst. Außerdem wird bei der Unterteilung in Bins darauf geachtet, dass mehrere Cluster-Zentren in einem Bin nach Möglichkeit vermieden werden. Die Justierung der Auflösung der Bins trägt dabei wesentlich zur Adaption an unterschiedliche Ausgabegeräte bei.

- Integration einer rechteckigen Fish-Eye-Verzerrung zur Unterteilung der Bins anhand der gegebenen Cluster-Zentren, um so die visuelle Unterscheidbarkeit zu garantieren. Diese Erweiterung lässt sich wahlweise an- und abschalten

Für die Adaption auf der Prozessebene stehen unterschiedliche *Mechanismen* zur Verfügung. Zur Auswahl eines geeigneten *Mechanismus* wurde im Rahmen dieser Dissertation eine zweistufige Auswahlstrategie in Form eines Entscheidungsbaums erarbeitet. In einem ersten Schritt wird die Effektivität bestimmt, indem ermittelt wird, ob *Visual Clutter* auftritt, da dieses einen wesentlichen Einfluss auf die Effektivität der visuellen Repräsentation hat. Dafür wurde im Rahmen der vorliegenden Arbeit die Consistency [SNLH09] verwendet.
Tritt bei der vorliegenden visuellen Repräsentation *Visual Clutter* auf, so wird in einem zweiten Schritt ein geeigneter *Adaptionsmechanismus* ausgewählt, um die charakteristischen Eigenschaften des Ausgabegerätes zu berück-

sichtigen. Standardmäßig wird dabei eine Adaption im Bildraum vorgenommen, um so nur die Repräsentation der Daten zu verändern. Die Arbeiten zum Entscheidungsbaum wurden auf dem ersten IEEE CoVIS Workshop auf der VisWeek 2009 veröffentlicht [FTSS09].

Lokale Adaption Bisher wurden *Adaptionsmechanismen* diskutiert, die sich auf die gesamte visuelle Repräsentation auswirken. Aber nicht immer ist so eine globale Anpassung notwendig und angebracht. Aus dieser Intension heraus ist das Konzept der *Smart Lenses* entstanden. Allgemein handelt es sich bei Linsen um Funktionen, die auf einem lokal begrenzten Bereich definiert sind.
Linsen werden seit langem in der Informationsvisualisierung eingesetzt. Sie unterscheiden sich je nach Funktion und Wirkungsbereich. Die neue Idee der vorliegenden Arbeit ist es, Linsen auf allen Stufen der Visualisierungspipeline zur Adaption lokaler Bereiche einzusetzen (vgl. [FTS07]).
Während die Parameter der Linse wie Form, Position und Funktion üblicherweise durch den Benutzer direkt gesteuert werden, geht das in dieser Promotion entwickelte Konzept der *Smart Lenses* noch einen Schritt weiter und setzt diese Parameter automatisch in Abhängigkeit von den aktuellen Rahmenbedingungen.
Diese Rahmenbedingungen werden über ein XML-Script festgelegt und betreffen beispielsweise nutzerspezifische Wünsche, wie z. B. eine bevorzugte Linsenform oder ein spezielles Interesse bezogen auf die gegebenen Datenwerte, wie z. B. Extremwerte. Entsprechend dieser Angabe wird dann die Linse automatisch gesetzt.
Linsen lassen sich zudem kombinieren, so können aus Linsen mit vergleichsweise einfacher Funktionalität Linsen mit weitaus komplexeren Linsenfunktionen *On-the-Fly* erstellt werden.
Linsen stellen somit ein sehr mächtiges Werkzeug zur lokalen Adaption von visuellen Repräsentationen dar. Die Konzepte hierzu wurden auf der Smart Graphics 2008 [TFS08a] veröffentlicht.

Progressive Informationsdarstellung Bisher wurde von einer adaptiven Informationsdarstellung ausgegangen, das heißt, die visuelle Repräsentation wird global oder lokal angepasst und dementsprechend auf verschiedenen Ausgabegeräten angezeigt.
Ein alternativer Ansatz wird in dieser Arbeit mit der *progressiven Informationsdarstellung* vorgestellt. Dieser basiert auf der Grundidee der progressiven Bilddarstellung, wie sie auch im WWW (World Wide Web) genutzt wird. Dabei wird die Kodierung der Bilddaten so gewählt, dass die Dekodierung eines abgeschnittenen Datenstroms der kodierten Daten das Bild mit weniger

Details wiederherstellt [TA08].
In [RLTS08, RS09] wird diese Idee aufgegriffen und eine *progressive Informationsdarstellung* eingeführt, die analog zur progressiven Bildanzeige visuelle Repräsentationen von Daten progressiv verfeinert. Dieses Konzept wurde in einer Kooperation noch einen Schritt weiterentwickelt und an die Erfordernisse einer Multi-Display-Umgebung angepasst.
Die Idee ist dabei, dass die progressive Übertragung individuell für jedes Gerät gestoppt wird, wenn verfeinerte Informationen nicht mehr darstellbar sind.
Als geeignete Abbruchkriterien für diesen Prozess wurde die *Auflösung des Displays* und die benötigte *Verarbeitungszeit* identifiziert. Dazu wurde mit einem ersten Ansatz der Ressourcenverbrauch *On-the-Fly* abgeschätzt.
Der Vorteil dieser Methode besteht darin, dass ein- und derselbe Datenstrom für unterschiedliche Geräte so lange angezeigt und verfeinert wird, bis die technischen Grenzen des Gerätes erreicht sind. Das vorgestellte Konzept wurde auf der IMC 2009 veröffentlicht (vgl. [TSR09]).

Die Konzepte zur adaptiven und progressiven Informationsanzeige adressieren unterschiedliche Ausgabegeräte und wurden in einem Framework prototypisch umgesetzt. Ein wichtiges Resultat dieser Arbeit ist es, dass dieses Framework dabei in der Lage ist, dynamisch Änderungen im Geräteensemble zu erkennen und darauf zu reagieren. Die Ergebnisse zum Framework sollen im Folgenden noch einmal zusammengefasst werden.

Für das **Framework** wurde eine serviceorientierte Architektur zugrundegelegt. Das Framework ist in drei Schichten aufgeteilt: Die *Serviceschicht*, welche die einzelnen Services enthält, die *Steuerschicht*, welche die Steuerung und Verwaltung der Services übernimmt und die *Modellschicht*, welche die Beschreibungen der Rahmenbedingungen erfasst.
Dabei realisieren die Services die Operatoren des Data-State-Reference-Models. Aus diesen Services wird *On-the-Fly* die Visualisierungspipeline aufgebaut.
Die große Anzahl der möglichen Services erschwert aber die Auswahl der in die Visualisierungspipeline einzubindenden Services. Aus diesem Grund wurden in dieser Arbeit die drei verschiedene Abstraktionsstufen: *Services*, *Service-Interface* und *Pipeline-Template* eingeführt. Ein Service-Interface ist dabei die abstrakte Beschreibung eines Services, ein Pipeline-Template umfasst mehrere Service-Interfaces und deren Verknüpfungen untereinander, welche zusammen eine Visualisierungspipeline repräsentieren.
Das Binden der Services an ein gegebenen Service-Interface ist bei der Umsetzung einer serviceorientierten Visualisierung ein wichtiger Punkt. Im We-

sentlichen wurden hierfür zwei unterschiedliche Strategien identifiziert und umgesetzt:

1. **Geräte-orientiert** Hierbei werden beim Binding die Services bevorzugt, die auf Geräten ausgeführt werden, von denen bereits Services eingebunden wurden. Ziel dieser Strategie ist es, die Kommunikation zwischen den Services im Vergleich zu einer Kommunikation über ein heterogenes Netzwerk zu verbessern.

2. **Nutzer-orientiert** Hierbei werden Services beim Binding bevorzugt, die durch ein Nutzerprofil vorgegeben oder auf dem Gerät des betreffenden Benutzers ausgeführt werden.

Für die vorliegenden Konzepte wurden ausgewählte Services implementiert. Bereits in der ersten Phase des GRKs können Informationsvisualisierungen, wie Scatterplots und Parallele Koordinaten für dynamische und heterogene Ausgabegeräte *On-the-Fly* konfiguriert werden.
Um jedoch einerseits Echtzeitanforderungen zu garantieren und andererseits ganz unterschiedliche Konzepte einbinden zu können, werden in der Steuerschicht Defaultwerte verwendet, die sich interaktiv verändern lassen und künftig durch Vorgaben aus dem Tupelspace ersetzt werden können. So wird z. B. standardmäßig ein Binning bei Darstellungen auf *mobilen Geräten* durchgeführt.
Die Konzepte zum Framework[1] wurden auf der IV 2009 [TTS09] veröffentlicht.

Fazit: Die Kombination von Informationsvisualisierung und *Smart Environment* stellt einen neuen Forschungzweig dar, der erstmals auch auf dem CoVis-Workshop der VisWeek 2009 thematisiert wurde.
Die vorliegende Arbeit stellt einen ersten Schritt in diese Richtung dar. Für die adressierten Probleme wurden erste Konzepte entwickelt und prototypische Lösungen umgesetzt. Allerdings besteht noch ein erheblicher Bedarf an weiteren Forschungsarbeiten in diesem Bereich. Einige offene Punkte sollen im folgenden Abschnitt zusammengefasst werden.

6.2 Ausblick

In dieser Dissertation wurde ein Klassifikationsschema entworfen und für jede Kategorie exemplarisch ein Ansatz entwickelt. Ein Ziel künftiger Arbeiten

[1] Alleine die Grundfunktionen des Frameworks inklusive der globalen Anpassung umfasst zurzeit ca. 67.000 Programmzeilen. Hinzu kommt das Softwareartefakt zur lokalen Adaption.

kann daher die Entwicklung weiterer *Adaptionsmechanismen* sein.
Beispielsweise kann das *Information-Hiding* als ein weiterer *Adaptionsmechanismus* im Datenraum untersucht werden. Auch können weitere Untersuchungen zu einem *Smart-Color-Coding* als ein *Adaptionsmechanismus* für das Mapping untersucht werden. Aktuell wird dies in einer Folgepromotion von Axel Radloff untersucht.

In dieser Arbeit wird die Auswahl der *Adaptionsmechanismen* über geeignete Default-Settings und interaktive Anpassungen realisiert.
Bisher wurden aber keine Untersuchungen durchgeführt, wann eine globale Adaption, wann eine lokale Adaption oder wann eine progressive Informationsvisualisierung zur Adaption eingesetzt werden soll. Da es sich dabei um grundlegend verschiedene Ansätze handelt, sind demzufolge auch grundlegende Untersuchungen erforderlich, um diese Frage entscheiden zu können.
Innerhalb der Ansätze sind ebenfalls weitere Untersuchungen erforderlich. So wurden zwar bei der globalen Adaption von Scatterplots erste konkrete quantitative Werte ermittelt und Usability Tests durchgeführt, aber für weitere Visualisierungstechniken fehlen konkrete Untersuchungen. Somit ergeben sich gerade in diesem Bereich neue Herausforderungen.

Das Software-Architektur-Konzept in dieser Arbeit baut auf eine serviceorientierte Architektur auf. Die realisierten Services stellen aktuell nur eine Grundfunktionalität bereit. In weiteren Arbeiten muss der Service-Pool erweitert werden, um eine größere Bandbreite an Informationsvisualisierungen in *Smart-Meeting-Rooms* realisieren zu können.
Bisher wird die Auswahl der in einer Informationsvisualisierung verwendeten Services durch die Default-Settings gesteuert. Ziel von weiteren Arbeiten muss es sein, die Settings aus dem Tupelspace zu beziehen und so die gesamte Breite an Informationen, die in einem *Smart-Meeting-Room* zur Verfügung stehen, zur Adaption einzusetzen. Das heißt, dass der im Konzept dieser Arbeit eingeführten Modellschicht eine größere Bedeutung zukommen muss.
Weitere interessante Fragen ergeben sich aus den Interaktionen auf den visuellen Repräsentationen. Wird eine visuelle Repräsentation auf mehreren Ausgabegeräten dargestellt, muss die Frage gestellt werden, auf welchen der visuellen Repräsentation eine Interaktion eine Auswirkung hat?
So kann die Interaktion auf alle Ausgabegeräte wirken oder nur auf das Ausgabegerät, wo die Interaktion durchgeführt wurde.
Rücken die Interaktionen stärker in den Vordergrund, spielt auch die Performance der Informationsvisualisierung verstärkt eine Rolle. In diesem Zusammenhang bietet es sich an, die Parallelisierung der Erzeugung der visuellen Repräsentationen zu untersuchen, da ohnehin mehrere Geräte des *Smart-*

Meeting-Rooms in den Erzeugungsprozess einbezogen werden.

Ein weiteres Feld, das untersucht werden muss, ist die Integration der in dieser Arbeit vorgestellten Konzepte zur globalen Adaption, lokale Adaption und progressiven Informationsdarstellungen in neue Anwendungen.
Hierfür bieten sich Systeme wie beispielsweise *Caleydo* der TU Graz (vgl. [SLK+09]) an [2].

In dieser Arbeit steht die Adaption der visuellen Repräsentation an dynamische, heterogene Ausgabegeräte im Vordergrund. Weitere Untersuchungen zur Adaption von visuellen Repräsentationen sollten auch andere Rahmenbedingungen untersuchen.
So wird aktuell von Axel Radloff in einer Folgepromotion untersucht, wie der Nutzer stärker bei der Adaption der visuellen Repräsentation in *Smart-Meeting-Room* berücksichtigt werden kann.

Informationsvisualisierungen in *Smart-Environments* sind ein vergleichsweise junges und auch ein sehr komplexes Forschungsfeld, wo viele verschiedene Disziplinen zusammenarbeiten. In der Zukunft werden daher in diesem Bereich viele neue Fragen auftauchen, die gelöst werden müssen. Die hier vorliegende Arbeit mit ihren Konzepten und Implementierungen stellt einen Grundstein und ersten Schritt für diese zukünftigen Untersuchungen dar, weitere Arbeiten werden in diesem Bereich folgen.

[2] Hierbei ist das Framework Gegenstand einer Kooperation mit der TU Graz, und Bestandteil eines geplanten DACH-Antrages.

Anhang A

A.1 Begriffe

visuelle Repräsentationen Sind das Ergebnis der Visualisierung; das heißt, sie sind die bildliche Darstellung der in den Visualisierungsprozess eingespeisten Daten. (vgl. [Jun98]).

Visualisierungspipeline umfasst die Hauptschritte der Erzeugung einer visuellen Repräsentation. Dies sind (in dieser Reihenfolge) die Schritte Filtering, Mapping und Rendering (vgl. [Noc07])

Rahmenbedingungen Jede visuelle Repräsentation wird unter Berücksichtigen von gewissen Rahmenbedingungen erstellt. Diese lassen sich grundsätzlich in die vier Bereichen: *Ziele*, *Benutzer*, *Daten* und *Geräte* unterteilen (vgl. [TS07]).

Visualisierungstechnik Je nach vorliegenden Rahmenbedingungen werden unterschiedliche Visualisierungstechniken eingesetzt. Ein Visualisierungstechnik gibt dabei in der Regel eine Visualisierungspipeline vor, wie die gegebenen Daten in eine visuelle Repräsentation zu überführen sind. Bekannte Vertreter sind beispielsweise Skatterplotts, Parallelekoordinaten [ID90] oder Treemaps [Wat99].

Adaption bedeutet allgemein: Anpassung. In dieser Arbeit bezieht sich die Anpassung in der Regel auf den Erzeugungsprozess der visuellen Repräsentation.

Adaptionsmechanismus ist die konkrete Umsetzung einer Adaption der Visualisierungspipeline an die vorliegenden Rahmenbedingungen.

Adaptionsstrategie dient der Auswahl von geeigneten *Adaptionsmechanismen* unter Berücksichtigung der aktuellen Rahmenbedingungen.

visuelle Primitive Ein visuelles Primitiv ist die Repräsentation von ein oder mehr Datenwerten auf der Stufe der *visuellen Abstraktionen* des Data-State-Reference-Modells (vgl. [CR98, Chi00]). Beispielsweise sind Punkte, Linsen und Flächen grundlegende *visuelle Primitive*. Jedem visuellen Primitiv sind zudem visuelle Variablen wie Farbe, Position, Größe, Helligkeit, Richtung, Muster und Form zugeordnet (vgl. [Ber82]).

grafische Primitive Bei grafischen Primitiven handelt es sich um die Repräsentationen der *visuellen Primitive* in den Bilddaten.

visuelle Variablen Bei den *visuellen Variablen* handelt es sich folglich Bertin [Ber82] um Position, Form, Farbe, Helligkeit, Orientierung, Textur und Größe. Später wurde auch noch die Bewegung hinzugenommen. Siehe Abbildung A.1

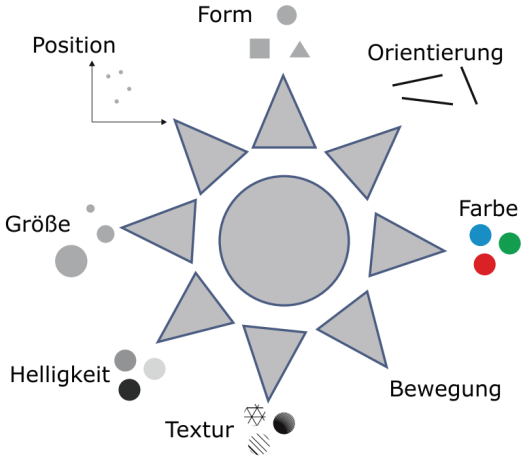

Abbildung A.1: Visuelle Variablen, wie sie von Bertin [Ber82] definiert wurden und zusätzlich Bewegung, aus [KEM07].

A.2 Konflikterkennung von Linsen

Um Konflikte zwischen einzelnen Linsen automatisch erkennen zu können, ist eine formale Beschreibung der Datenmenge auf der die Linsenfunktion arbeitet und die Datenmenge, welche die Linsenfunktion manipuliert, erforderlich. Zusätzlich sind formale Tests erforderlich, durch die sich die Konflikte zwischen zwei oder mehr Linsen aufspüren lassen, deren Linsenregionen sich überschneiden. Für die formale Beschreibung wird eine Mengen-Notation verwendet.
Jeder der vier Stufen im Data-State-Referenz Modell wird repräsentiert durch eine Menge A_i von Attributen a_{ik}; $1 \leq i \leq 4; 1 \leq k \leq |A_i|$ denen sich die Datenwerte auf den korrespondierenden Stufen zuordnen lassen [FTS07]. So korrespondiert A_1 zu den Attributen der gegeben Daten, A_2 repräsentiert Attribute auf der Stufe der *analytical abstractions* und so weiter.

Der Operator einer Linsenfunktion f transformiert Datenwerte mit den dazugehörigen Attributen einer Quell-Menge S_n^f (n^{ten} Stufe) zu einer Ziel-Menge T_m^f (m^{ten} Stufe):

$$S_n^f \subseteq \bigcup_{1 \leq i \leq n} A_i, \quad T_m^f \subseteq \bigcup_{m \leq i \leq 4} A_i$$

Dieser Formalismus erlaubt es, die zuvor erwähnten Konflikte (vgl. 4.4) zu erkennen und zum Teil automatisch aufzulösen. Sei:

$$f : S_n^f \rightarrow T_m^f; 1 \leq n \leq m \leq 4, \text{ und}$$
$$g : S_p^g \rightarrow T_q^g; 1 \leq p \leq q \leq 4.$$

Ein potenzieller *Abhängigkeitskonflikt* tritt auf, wenn $T_m^f \cap S_p^g \neq \emptyset$. In diesem Fall ist g von f abhängig. Um diesen *Abhängigkeitskonflikt* zu vermeiden, muss f vor g ausgeführt werden. Das garantiert, dass erst dann mit den Daten der betreffenden Attributen weitergearbeitet wird, wenn alle Änderungen der Linsenfunktion f an den Daten durchgeführt wurden.
Ein *Schreibkonflikt* tritt auf, wenn $T_m^f \cap T_q^g \neq \emptyset$. Wie bereits erwähnt wurde, können diese Konflikte nicht automatisch aufgelöst werden.

A.3 Eigenschaften konkreter Ausgabegeräte

Technologie	übliche Bandbreiten
GSM	55kBit/s - 220Bit/s
UMTS	384kBit/s - 7,2MBit/s
Bluetooth	732kBit/s - 2,1MBit/s
WLAN	11MBit/s - 300MBit/s
LAN	10MBit/s - 1GBit/s
Firewire	100Mbit/s - 3,2GBit/s

Tabelle A.1: Übliche Bandbreiten unterschiedlicher Kommunikationsmedien, aus [Ros09b]

Gerätetyp	Aufl. in Px	Abm. in mm
Smartphone(HTC Magic)	320x480	47x69
Smartphone(BlackBerry Curve 8520)	240x320	49x37
PDA (HP iPAQ rx3700)	240x320	55x72
PDA (Fujitsu Siemens Pocket LOOX)	480x640	54x73
Laptop (HP Compaq nx9420)	1680x1050	365x230
Laptop (Dell Precision M4400)	1920x1200	332x206
Monitor(Samsung SyncMaster 2494HS)	1920x1080	531x302
Powerwall (6x4 mal 1920x1200)	11520x4800	3840x1680
Beamer (NEC NP905)	1024x768	530-762 Dia.
Beamer (Panasonic PT-AE4000E)	1920x1080	100-500 Dia.

Tabelle A.2: Unterschiedliche Ausgabegeräte mit Angaben, zu deren Auflösung und Abmessungen des Displays

Gerätetyp	Entfernung des Betrachters in mm
Smartphone, PDA	300
Laptop	500
Monitore	800
Beamer	4000

Tabelle A.3: Empirisch ermittelte, durchschnittliche Abstände der Geräte zum Betrachter, aus [Ros09b]

Literaturverzeichnis

[AA02] Andrienko G. und Andrienko N. *Intelligent support of visual data analysis in descartes.* In *SMARTGRAPH '02: Proceedings of the 2nd international symposium on Smart graphics*, pages 21–26. ACM Press, New York, NY, USA, 2002. ISBN 1-58113-555-6.

[Abo99] Abowd G. *Classroom 2000: An experiment with the instrumentation of a living educational environment. IBM Systems Journal*, 38(4):508–530, 1999.

[AE06] Aarts E. und Encarnação J. *True Visions:: the Emergence of Ambient Intelligence.* Springer, 2006.

[AHF+02] Arnstein L., Hung C.Y., Franza R., Zhou Q.H., Borriello G., Consolvo S., und Su J. *Labscape: A smart environment for the cell biology laboratory. IEEE Pervasive Computing*, 1(3):13–21, 2002. ISSN 1536-1268.

[AHFS08] Ali K., Hartmann K., Fuchs G., und Schumann H. *Adaptive layout for interactive documents.* In *Proceedings of the 9th international symposium on Smart Graphics*, page 254. Springer, 2008.

[And06] Andrienko G. *Exploratory Analysis of Spatial And Temporal Data: A Systematic Approach.* Springer, 2006.

[ARL+06] Alimohideen J., Renambot L., Leigh J., Johnson A., Grossman R., und Sabala M. *Pavis–pervasive adaptive visualization and interaction service.* In *CHI Workshop on Information Visualization and Interaction Techniques for Collaboration Across Multiple Displays, Montreal, Canada.* 2006.

[AS07] Avidan S. und Shamir A. *Seam carving for content-aware image resizing. International Conference on Computer Graphics and Interactive Techniques*, 2007.

[BB04] Bala R. und Braun K. *Color-to-grayscale conversion to maintain discriminability.* In *Proc. SPIE*, volume 5293, pages 196–202. 2004.

[BBP+00] Bentley T., Balbo S., Paris C., Tarby J., und Johnston L. *Task modelling review of methods, tools, and other.* Technical Report, 2000.

[BBR02] Bauer M., Becker C., und Rothermel K. *Location models from the perspective of context-aware applications and mobile ad hoc networks. Personal and Ubiquitous Computing*, 6(5):322–328, 2002.

[BCC+05] Bavoil L., Callahan S., Crossno P., Freire J., Scheidegger C., Silva C., und Vo H. *Vistrails: enabling interactive multiple-view visualizations.* In *Visualization, 2005. VIS 05. IEEE*, pages 135–142. Oct. 2005.

[BCL+04] Bohn J., Coroama V., Langheinrich M., Mattern F., und Rohs M. *Living in a world of smart everyday objects – social, economic, and ethical implications. Journal of Human and Ecological Risk Assessment*, 10(5):763–786, October 2004.

[BDF+06] Biehl N., Düsterhöft A., Forbrig P., Fuchs G., Reichart D., und Schumann H. *Advanced Multi-Modal User Interfaces for Mobile Devices - Integration of Visualization, Speech Interaction and Task Modeling.* In *Proceedings 17th IRMA International Conference.* Washington D.C., USA, 2006. (Short Paper).

[BDG+04a] Brodlie K., Duce D., Gallop J., Walton J., und Wood J. *Distributed and collaborative visualization.* In *Computer Graphics Forum*, volume 23, pages 223–251. Blackwell Synergy, 2004.

[BDG+04b] Brodlie K., Duce J., Gallop J., Sagar M., Walton J., und Wood J. *Visualization in grid computing environments. Visualization, 2004. IEEE*, pages 155–162, Oct. 2004.

[BEPW96] Backhaus K., Erichson B., Plinke W., und Weiber R. *Multivariate Analysemethoden: Eine anwendungsorientierte*

Einführung. Springer Verlag, Berlin Heidelberg New York, 1996. ISBN 3-540-60971-2.

[Ber82] Bertin J. *Graphische Darstellungen und die graphische Weiterverarbeitung der Information*. Walter de Gruyter, 1982.

[BKDE00] Bender M., Klein R., Disch A., und Ebert A. *A functional framework for web-based information visualization systems. IEEE Transactions on Visualization and Computer Graphics*, 6:8–23, 2000. ISSN 1077-2626.

[BMK+00] Brumitt B., Meyers B., Krumm J., Kern A., und Shafer S. *Easyliving: Technologies for intelligent environments. Lecture notes in computer science*, pages 12–29, 2000.

[BQ08] Bennett M. und Quigley A. *Perceptual usability: predicting changes in visual interfaces & designs due to visual acuity differences*. In *Proceedings of the working conference on Advanced visual interfaces*, pages 380–383. ACM New York, NY, USA, 2008.

[BRH+08] Burghardt C., Reisse C., Heider T., Giersich M., und Kirste T. *Implementing scenarios in a smart learning environment*. In *Proceedings of 4th IEEE International Workshop on PervasivE Learning*. Hongkong, 2008.

[BS04] Bertini E. und Santucci G. *Quality metrics for 2d scatterplot graphics: automatically reducing visual clutter. Lecture notes in computer science*, pages 77–89, 2004.

[BS05a] Baar D. und Shoemaker G. *Pliable display technology for the common operational picture*. 2005.

[BS05b] Bertini E. und Santucci G. *Is it darker? improving density representation in 2 d scatter plots through a user study*. In *Proc. SPIE*, volume 5669, pages 158–167. 2005.

[BSP+93] Bier E., Stone M., Pier K., Buxton W., und DeRose T. *Toolglass and magic lenses: the see-through interface. Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 73–80, 1993.

[BSP97] Bier E., Stone M., und Pier K. *Enhanced illustration using magic lens filters. IEEE Comput. Graph. Appl.*, 17(6):62–70, 1997. ISSN 0272-1716.

[Bur10] Burghardt C. *Modelle und Deskriptoren für physische Umgebungen (Pre-final).* Ph.D. thesis, Universität Rostock, 2010.

[Cas91] Casner S. *A task-analytic approach to the automated design of graphic presentations. ACM TRANS. GRAPH.*, 10(2):111–151, 1991.

[CCC02] Carmo M., Cunha J., und Claudio A. *Ivprototype - an information visualization prototype.* pages 159–164. 2002. ISSN 1093-9547.

[CCT01] Calvary G., Coutaz J., und Thevenin D. *A unifying reference framework for the development of plastic user interfaces. Proceedings Engineering Human-Computer Interaction*, pages 173–192, 2001.

[CD05] Cook D. und Das S. *Smart environments: Technology, protocols and applications.* Wiley-Interscience, 2005.

[CD07] Cook D. und Das S. *How smart are our environments? an updated look at the state of the art. Pervasive and Mobile Computing*, 3(2):53–73, 2007.

[CDM+00] Cheverst K., Davies N., Mitchell K., Friday A., und Efstratiou C. *Developing a context-aware electronic tourist guide: some issues and experiences.* In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 17–24. ACM, New York, NY, USA, 2000. ISBN 1-58113-216-6.

[CG06] Celentano A. und Gaggi O. *Context-aware design of adaptable multimodal documents. Multimedia Tools and Applications*, 29(1):7–28, 2006.

[Che04] Chen H. *Towards design patterns for dynamic analytical data visualization. Proceedings Of SPIE Visualization and Data Analysis*, 2004.

[Chi00] Chi E. *A taxonomy of visualization techniques using the data state reference model. Information Visualization, 2000. InfoVis 2000. IEEE Symposium on*, pages 69–75, 2000.

[CLB+03] Chiu P., Liu Q., Boreczky J., Foote J., Fuse T., Kimber D., Lertsithichai S., und Liao C. *Manipulating and annotating slides in a multi-display environment.* In *Human-Computer Inter-*

action INTERACT '03: IFIP TC13 International Conference on Human-Computer Interaction. 2003.

[CMS99] Card S., Mackinlay J., und Shneiderman B. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999.

[CR98] Chi E.H. und Riedl J.T. *An operator interaction framework for visualization systems. InfoVis*, 00:63, 1998.

[DAB05] Duda I., Aleksy M., und Butter T. *Architectures for mobile device integration into service-oriented architectures. Mobile Business, 2005. ICMB 2005. International Conference on*, pages 193–198, 2005.

[DE02] Dix A. und Ellis G. *by chance enhancing interaction with large data sets through statistical sampling*. In *AVI '02: Proceedings of the Working Conference on Advanced Visual Interfaces*, pages 167–176. ACM, New York, NY, USA, 2002. ISBN 1-58113-537-8.

[DMKS01] Ding Y., Malaka R., Kray C., und Schillo M. *Raja: a resource-adaptive java agent infrastructure. Proceedings of the fifth international conference on Autonomous agents*, pages 332–339, 2001.

[DS110] *Dagstuhl-seminar 10241 information visualization*, Juni 2010.

[EBD05] Ellis G., Bertini E., und Dix A. *The sampling lens: making sense of saturated visualisations*. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1351–1354. ACM, New York, NY, USA, 2005. ISBN 1-59593-002-7.

[ED02] Ellis G. und Dix A. *Density control through random sampling: an architectural perspective*. In *Information Visualisation, 2002. Proceedings. Sixth International Conference on*, pages 82–90. 2002.

[ED06a] Ellis G. und Dix A. *Enabling automatic clutter reduction in parallel coordinate plots. IEEE Transactions on Visualization and Computer Graphics*, 12(5):717–724, 2006. ISSN 1077-2626.

[ED06b] Ellis G. und Dix A. *The plot, the clutter, the sampling and its lens: occlusion measures for automatic clutter reduction*. In *AVI*

'06: Proceedings of the working conference on Advanced visual interfaces, pages 266–269. ACM, New York, NY, USA, 2006. ISBN 1-59593-353-0.

[ED07] Ellis G. und Dix A. *A taxonomy of clutter reduction for information visualisation. IEEE Transactions on Visualization and Computer Graphics*, 13(6):1216–1223, 2007. ISSN 1077-2626.

[EEF+07] Eick S., Eick M., Fugitt J., Horst B., Khailo M., und Lankenau R. *Thin client visualization.* In *Visual Analytics Science and Technology, 2007. VAST 2007. IEEE Symposium on*, pages 51–58. 30 2007-Nov. 1 2007.

[Eic00] Eick S. *Visual discovery and analysis. IEEE Transactions on Visualization and Computer Graphics*, 6(1):44–58, 2000.

[EK05] Encarnação J. und Kirste T. *Ambient intelligence: Towards smart appliance ensembles. From Integrated Publication and Informations Systems to Virtual Information and Knowledge Environments*, pages 261–270, 2005.

[EVP01] Eisenstein J., Vanderdonckt J., und Puerta A. *Applying model-based techniques to the development of uis for mobile computers.* In *IUI '01: Proceedings of the 6th international conference on Intelligent user interfaces*, pages 69–76. ACM Press, New York, NY, USA, 2001. ISBN 1-58113-325-1.

[EWE09] Eissele M., Weiskopf D., und Ertl T. *Interactive context-aware visualization for mobile devices.* page 167, 2009.

[EZ98] ETH-Zentrum N. *Sehen und bildschirm*, 1998.

[FBBB03] Follin J., Bouju A., Bertrand F., und Boursier P. *Management of multi-resolution data in a mobile spatial information visualization system.* In *Web Information Systems Engineering Workshops, 2003. Proceedings. Fourth International Conference on*, pages 92–99. Dec. 2003.

[FD06] Fredj Z.B. und Duce D.A. *Grassml: Accessible smart schematic diagrams for all.* In *W4A'2006 International Cross-Disciplinary Conference on Web Accessibility*, volume 134 of *ACM International Conference Proceeding Series*, pages 57–60. ACM Press, Edinburgh, Scotland, UK, May 2006. ISBN 1-59593-281-X.

[FFH00] Flachsbart J., Franklin D., und Hammond K. *Improving human computer interaction in a classroom environment using computer vision.* In *IUI '00: Proceedings of the 5th international conference on Intelligent user interfaces*, pages 86–93. ACM, New York, NY, USA, 2000. ISBN 1-58113-134-8.

[FFIT00] Fujishiro I., Furuhata R., Ichikawa Y., und Takeshima Y. *Gadget/iv: A taxonomic approach to semi-automatic design of information visualization applications using modular visualization environment. infovis*, 00:77, 2000.

[FHRS08] Fuchs G., Holst M., Rosenbaum R., und Schumann H. *3d mesh exploration for smart visual interfaces.* In *VISUAL '08: Proceedings of the 10th international conference on Visual Information Systems*, pages 80–91. Springer-Verlag, Berlin, Heidelberg, 2008. ISBN 978-3-540-85890-4.

[FL08] Forlines C. und Lilien R. *Adapting a single-user, single-display molecular visualization application for use in a multi-user, multi-display environment.* In *AVI '08: Proceedings of the working conference on Advanced visual interfaces*, pages 367–371. ACM, New York, NY, USA, 2008. ISBN 1-978-60558-141-5.

[Fra10] Frank J. *Entwicklung eines Editors zur Beschreibung von Informationsdarstellungen (In Arbeit).* Master's thesis, Universität Rostock, 2010.

[FRSF06] Fuchs G.A., Reichart D., Schumann H., und Forbrig P. *Maintenance support – case study for a multimodal mobile user interface.* In *Multimedia on Mobile Devices II*, volume 6074 of *Proceedings of SPIE*. The International Society for Optical Engineering, January 2006. ISBN 0-8194-6114-8.

[FS09] Fuchs G. und Schumann H. *Smart visual interfaces: Adaptive anzeige graphischer modelldaten. CAD-CAM Report*, 1/2-09:50–53, Januar 2009.

[FTIN97] Fujishiro I., Takeshima Y., Ichikawa Y., und Nakamura K. *Gadget: goal-oriented application design guidance for modular visualization environments.* In *VIS '97: Proceedings of the 8th conference on Visualization '97*, pages 245–ff. IEEE Computer Society Press, Los Alamitos, CA, USA, 1997. ISBN 1-58113-011-2.

[FTS07] Fuchs G., Thiede C., und Schumann H. *Pluggable lenses for interactive visualizations.* In *Poster Compendium of InfoVis.* October 2007.

[FTSS09] Fuchs G., Thiede C., Sips M., und Schumann H. *Device-based adaptation of visualizations in smart environments.* In *Workshop Collaborative Visualization on Interactive Surfaces (CoVIS) IEEE VisWeek.* Atlantic City, USA, 11th-16th October 2009.

[Fuc10] Fuchs G. *Task-driven Adaptation of Graphical Content in Smart Visual Interfaces (Pre-final).* Ph.D. thesis, University of Rostock, 2010.

[FWR99] Fua Y.H., Ward M.O., und Rundensteiner E.A. *Hierarchical parallel coordinates for exploration of large datasets.* In *VIS '99: Proceedings of the conference on Visualization '99*, pages 43–50. IEEE Computer Society Press, Los Alamitos, CA, USA, 1999. ISBN 0-7803-5897.

[GAW04] Grimstead I., Avis N., und Walker D. *Automatic distribution of rendering workloads in a grid enabled collaborative visualization environment. Proceedings of the 2004 ACM/IEEE conference on Supercomputing*, page 1, 2004.

[Ger09] Germer T. *Selbstorganisierende Systeme für die Computergrapfik.* Ph.D. thesis, Otto-von-Guericke-Universität Magdeburg, 2009.

[GFF+07] Giersich M., Forbrig P., Fuchs G., Kirste T., Reichart D., und Schumann H. *Towards an integrated approach for task modeling and human behavior recognition. LECTURE NOTES IN COMPUTER SCIENCE*, 4550:1109, 2007.

[GFS05] Griethe H., Fuchs G., und Schumann H. *A classification scheme for lens techniques.* In *Proceedings WSCG*, pages 89–92. 2005.

[GMLC01] Gal C.L., Martin J., Lux A., und Crowley J.L. *Smart office: Design of an intelligent environment. IEEE Intelligent Systems*, 16(4):60–66, 2001. ISSN 1541-1672.

[Gol00] Golenhofen K. *Physiologie heute.* Urban, 2000.

[GOTG05] Gooch A., Olsen S., Tumblin J., und Gooch B. *Color2gray: salience-preserving color removal. Proceedings of ACM SIGGRAPH 2005*, 24(3):634–639, 2005.

[GRKM94] Goldstein J., Roth S., Kolojejchick J., und Mattis J. *A framework for knowledge-based interactive data exploration. Journal of visual languages and computing*, 5(4):339–363, 1994.

[GSGC08] Gilson O., Silva N., Grant P., und Chen M. *From web data to visualization via ontology mapping.* In *Computer Graphics Forum*, volume 27, pages 959–966. Blackwell Publishing, 2008.

[HBE+00] Hagen H., Barthel H., Ebert A., Divivier A., und Bender M. *A component-and multi agent-based visualization system architecture. IMC*, 2000.

[HCL05] Heer J., Card S.K., und Landay J.A. *prefuse: a toolkit for interactive information visualization.* In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 421–430. ACM, New York, NY, USA, 2005. ISBN 1-58113-998-5.

[HEHB08] Hau T., Ebert N., Hochstein A., und Brenner W. *Where to start with soa: Criteria for selecting soa projects. Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, pages 314–314, 2008.

[Hei09] Heider T. *A Unified Distributed System Architecture for Goal-based Interaction with Smart Environments.* Ph.D. thesis, University of Rostock, 2009.

[HHN00] Havre S., Hetzler B., und Nowell L. *Themeriver: Visualizing theme changes over time.* In *INFOVIS '00: Proceedings of the IEEE Symposium on Information Vizualization 2000*, page 115. IEEE Computer Society, Washington, DC, USA, 2000. ISBN 0-7695-0804-9.

[HK05a] Heider T. und Kirste T. *Multimodal appliance cooperation based on explicit goals: concepts & potentials. Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies*, pages 271–276, 2005.

[HK05b] Heider T. und Kirste T. *Smart environments and self-organizing appliance ensembles. Mobile Computing and Ambient Intelligence*, 2005.

[HNC+08] Hervás R., Nava S., Chavira G., Villarreal V., und Bravo J. *Pivita: Taxonomy for displaying information in pervasive and collaborative environments.* In *3rd Symposium of Ubiquitous Computing and Ambient Intelligence 2008*, page 293. Springer, 2008.

[HS05] Huhns M. und Singh M. *Service-oriented computing: key concepts and principles. IEEE Internet Computing*, 9(1):75–81, 2005.

[HW05] Hellenschmidt M. und Wichert R. *Goal-oriented assistance in ambient intelligence. Workshop on Experience Research in Ambient Intelligence. Eindhoven, The Netherlands*, 2005.

[ID90] Inselberg A. und Dimsdale B. *Parallel coordinates: a tool for visualizing multi-dimensional geometry.* In *VIS '90: Proceedings of the 1st conference on Visualization '90*, pages 361–378. IEEE Computer Society Press, Los Alamitos, CA, USA, 1990. ISBN 0-8186-2083-8.

[Ins85] Inselberg A. *The plane with parallel coordinates. The Visual Computer*, 1(2):69–91, 1985.

[JS05] Jung E. und Sato K. *A framework of context-sensitive visualization for user-centered interactive systems. Lecture notes in computer science*, 3538:423, 2005.

[Jun98] Jung V. *Integrierte Benutzerunterstützung für die Visualisierung in Geo-Informationssystemen.* Ph.D. thesis, University of Darmstadt, 1998.

[JW03] Jang S. und Woo W. *Ubi-ucam: A unified context-aware application model. Modeling and Using Context: 4th International and Interdisciplinary Conference, CONTEXT 2003, Stanford, CA, USA, June 23-25, 2003: Proceedings*, 2003.

[JWFS08] Jiang H., Wigdor D., Forlines C., und Shen C. *System design for the wespace: Linking personal devices to a table-centered multi-user, multi-surface environment. Proc. Tabletop 2008*, pages 105–112, 2008.

[Kea98] Keahey T. *The generalized detail-in-context problem. Information Visualization, IEEE Symposium on*, 0:44, 1998. ISSN 1522-404X.

[Kea99] Keahey T. *Area-normalized thematic views. Proceedings of International Cartography Assembly*, 1999.

[KEM07] Kerren A., Ebert A., und Meyer J. *Human-centered Visualization Environments: GI-Dagstuhl Research Seminar*. Springer, 2007.

[KF02] Kindberg T. und Fox A. *System software for ubiquitous computing. IEEE pervasive computing*, pages 70–81, 2002.

[KJDG+06] Kim H., Joslin C., Di Giacomo T., Garchery S., und Magnenat-Thalmann N. *Device-based decision-making for adaptation of three-dimensional content. The Visual Computer*, 22(5):332–345, 2006.

[KNS04] Kreuseler M., Nocke T., und Schumann H. *A history mechanism for visual data mining. infovis*, 00:49–56, 2004. ISSN 1522-404X.

[Kor09] Korsah K. *2009 ca web stress index*. Technical Report, Computer Associates International, 2009.

[KR90] Kaufman L. und Rousseeuw P. *Finding groups in data: An introduction to cluster analysis*. Wiley-Interscience; 9th edition, 1990.

[KR97] Keahey T.A. und Robertson E.L. *Nonlinear magnification fields. in IEEE Symposium on Information Visualization (INFOVIS '97)*, pages 51–58, 1997.

[Kre05] Kreuseler M. *Ein flexibles Framework zum Visuellen Data Mining*. Ph.D. thesis, University of Rostock, 2005.

[LA94] Leung Y. und Apperley M. *A review and taxonomy of distortion-oriented presentation techniques. ACM Transactions on Computer-Human Interaction (TOCHI)*, 1(2):126–160, 1994.

[LBC04] Looser J., Billinghurst M., und Cockburn A. *Through the looking glass: the use of lenses as an interface tool for augmented reality interfaces. Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and Southe East Asia*, pages 204–211, 2004.

[LED06] Lu Y., Ebert D.S., und Delp E.J. *Resource-driven content adaptation.* In C. A. Bouman, E. L. Miller, & I. Pollak, editor, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 6065 of *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, pages 167–177. February 2006.

[LH94] Loughlin M.M. und Hughes J.F. *An annotation system for 3d fluid flow visualization.* In *VIS '94: Proceedings of the conference on Visualization '94*, pages 273–279. IEEE Computer Society Press, Los Alamitos, CA, USA, 1994. ISBN 0-7803-2521-4.

[LNS06] Lange S., Nocke T., und Schumann H. *Visualisierungsdesign–ein systematischer Überblick. Proceedings of Simulation and Visualization (SimVis 06)*, 2006.

[LS08] Luboschik M. und Schumann H. *Illustrative halos in information visualization.* In *AVI '08: Proceedings of the working conference on Advanced visual interfaces*, pages 384–387. ACM, New York, NY, USA, 2008. ISBN 1-978-60558-141-5.

[Mac86] Mackinlay J. *Automating the design of graphical presentations of relational information. ACM Transactions on Graphics*, 5:2, 1986.

[Mar10] Marquardt F. *Dienstekomposition in intelligenten Umgebungen basierend auf KI-Planung (Pre-final).* Ph.D. thesis, Universität Rostock, 2010.

[MBNP03] Mao S., Bushmitch D., Narayanan S., und Panwar S. *Mrtp: a multiflow realtime transport protocol for ad hoc networks. Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, 4, 2003.

[MCEF94] Molnar S., Cox M., Ellsworth D., und Fuchs H. *A sorting classification of parallel rendering. Computer Graphics and Applications, IEEE*, 14(4):23–32, Jul 1994. ISSN 0272-1716.

[Mík07] Míkovec Z. *Adaptive interaction with graphical data in special environments.* Ph.D. thesis, Faculty of Electrical Engineering, Department of Computer Science and Engineering, Czech Technical University in Prague, 2007.

[MK88] Matsuoka S. und Kawai S. *Using tuple space communication in distributed object-oriented languages.* In *OOPSLA '88: Conference proceedings on Object-oriented programming systems, languages and applications*, pages 276–284. ACM, New York, NY, USA, 1988. ISBN 0-89791-284-5.

[Moe07] Moere A. *A model for self-organizing data visualization using decentralized multiagent systems. Advances in applied self-organizing systems*, page 291, 2007.

[MPS02] Mori G., Paterno F., und Santoro C. *Ctte: support for developing and analyzing task models for interactive system design. Software Engineering, IEEE Transactions on*, 28(8):797–813, 2002.

[MPS04] Mori G., Patern F., und Santoro C. *Design and development of multidevice user interfaces through multiple logical descriptions. IEEE Transactions on Software Engineering*, 30:507–520, 2004. ISSN 0098-5589.

[MSK+06] Merino C.S., Sips M., Keim D.A., Panse C., und Spence R. *Task-at-hand interface for change detection in stock market data.* In *AVI '06: Proceedings of the working conference on Advanced visual interfaces*, pages 420–427. ACM, New York, NY, USA, 2006. ISBN 1-59593-353-0.

[MuS09] *Musama homepage http://www.musama.de*, October 2009.

[MVL06] Massó; J.P.M., Vanderdonckt J., und López P.G. *Direct manipulation of user interfaces for migration.* In *IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces*, pages 140–147. ACM Press, New York, NY, USA, 2006. ISBN 1-59593-287-9.

[New95] *New york times business day, may 2*, 1995.

[NH06] Novotny M. und Hauser H. *Outlier-preserving focus+context visualization in parallel coordinates. IEEE Transactions on Visualization and Computer Graphics*, 12(5):893–900, 2006. ISSN 1077-2626.

[NHP+09] Nocke T., Heyder U., Petri S., Vohland K., Wrobel M., und Lucht W. *Visualization of biosphere changes in the context of climate change.* In *Information Technology and Climate Change*

: 2nd International Conference IT for empowerment [ITCC'08, Berlin, 2008]. 2009.

[Noc07] Nocke T. *Visuelles Data Mining und Visualisierungsdesign für die Klimaforschung*. Ph.D. thesis, University of Rostock, 2007.

[Nor06] North C. *Toward Measuring Visualization Insight. IEEE Computer Graphics and Applications*, 26(3):6–9, 2006. ISSN 0272-1716.

[NS02] Nocke T. und Schumann H. *Meta data for visual data mining. Proceedings Computer Graphics and Imaging, CGIM*, 2, 2002.

[Ope05] Open Geospatial Consortium. *OpenGIS Filter Encoding Implementation Specification, Version 1.1.0 (final)*. Open Geospatial Consortium Inc., 2005.

[OW05] Oh Y. und Woo W. *A unified application service model for ubihome by exploiting intelligent context-awareness. Ubiquitous Computing Systems: Second International Symposium, UCS 2004, Tokyo, Japan, November 8-9, 2004: Revised Selected Papers*, 2005.

[Pat00] *Model-based design of interactive applications. Intelligence*, 11(4):26–38, 2000. ISSN 1523-8822.

[PBB+08] Pike W., Bruce J., Baddeley B., Best D., Franklin L., May R., Rice D., Riensche R., und Younkin K. *The scalable reasoning system: Lightweight visualization for distributed analytics*. pages 131–138. Oct. 2008.

[Plo10] Plociennik C. *Spontane Kooperation von Multimedia-Geräten (Pre-final)*. Ph.D. thesis, Universität Rostock, 2010.

[PMM97] Paterno F., Mancini C., und Meniconi S. *ConcurTaskTrees: A diagrammatic notation for specifying task models*. In *Proceedings of Interact'97*, pages 362–369. Chapman & Hall, Sydney, 1997.

[Pro10] Propp S. *Usability-Evaluation in intelligenten Umgebungen*. Ph.D. thesis, University of Rostock, 2010.

[PS02] Paterno F. und Santoro C. *One model, many interfaces*. In *Proceedings of the 4th International Conference on Computer-Aided*

Design of User Interfaces, pages 143–154. Kluwer Academics Publishers, 2002.

[Rad08] Radloff A. *Ein erweitertes Mapping Konzept zur Visualisierung sehr großer Datenmengen.* Master's thesis, University of Rostock, 2008.

[Ras97] Rase W. *Fischauge-projektionen als kartographische lupen. Salzburger Kartographische Materialien*, 26:115–122, 1997.

[RH97] Robertson P.K. und Hutchins M. *An approach to intelligent design of color visualisation. G. Nielson, H. Hagen, H. Mueller: Scientific Visualisation. IEEE Computer Society, Los Alamitos*, pages 179–190, 1997.

[RJ06] Roard N. und Jones M.W. *Agents based visualization and strategies. WSCG 2006 conference proceedings*, 2006.

[RLS+96] Roth S., Lucas P., Senn J., Gomberg C., Burks M., Stroffolino P., Kolojechick A., und Dunmire C. *Visage: a user interface environment for exploring information. infovis*, 00:3, 1996.

[RLS10] Radloff A., Luboschik M., und Schumann H. *A new weaving technique for handling overlapping. ACM Conference for Advanced Visual Interfaces (AVI 2010), accepted*, 2010.

[RLTS08] Rosenbaum R., Luboschik M., Thiede C., und Schumann H. *Interactive poster: Progressive information visualization.* In *Poster Compendium of InfoVis'2008.* Ohio, USA, October 2008.

[RM90] Roth S.F. und Mattis J. *Data characterization for intelligent graphics presentation.* In *CHI '90: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 193–200. ACM, New York, NY, USA, 1990. ISBN 0-201-50932-6.

[RMSF09] Ramos C., Marreiros G., Santos R., und Freitas C. *Smart offices and intelligent decision rooms. Handbook of Ambient Intelligence and Smart Environments*, 1:851–880, 2009.

[Rob91] Robertson P.K. *A methodology for choosing data representations. IEEE Computer Graphics and Applications*, 11:56–67, 1991. ISSN 0272-1716.

[Ros09a] Rossol F. *Anwendung von Bildskalierungstechniken in der Visualisierung.* Master's thesis, University of Rostock, 2009.

[Ros09b] Rossol F. *Geräte-spezifische Informationsdarstellung für hierarchische Daten in Smart Environments.* Master's thesis, University of Rostock, 2009.

[RS07] Rosenbaum R. und Schumann H. *Chances and limits of progression in visualization.* In *Proceedings of SimVis2007.* 2007.

[RS09] Rosenbaum R. und Schumann H. *Progressive refinement: more than a means to overcome limited bandwidth.* In *Proceedings of SPIE*, volume 7243, page 72430I. 2009.

[RWF06] Rathsack R., Wolff A., und Forbrig P. *Using HCI-Patterns with Model-Based Generation of Advanced User Interfaces.* In *MDDAUI '06 - Model Driven Development of Advanced User Interfaces*, volume 214. 2006.

[Sch06] Schmitz R. *Ein Modell zur Beschreibung von Visualisierungszielen beim Einsatz von Farbkodierung zur Darstellung multivariater Daten.* Master's thesis, University of Rostock, 2006.

[SG02] Sousa Jo a.P. und Garlan D. *Aura: an architectural framework for user mobility in ubiquitous computing environments.* In *WICSA 3: Proceedings of the IFIP 17th World Computer Congress - TC2 Stream / 3rd IEEE/IFIP Conference on Software Architecture*, pages 29–43. Kluwer, B.V., Deventer, The Netherlands, The Netherlands, 2002. ISBN 1-4020-7176-0.

[SG08] Street J. und Gomaa H. *Software architectural reuse issues in service-oriented architectures. Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, pages 316–316, 2008.

[Shn92] Shneiderman B. *Tree visualization with tree-maps: 2-d space-filling approach. ACM Transactions on graphics (TOG)*, 11(1):92–99, 1992.

[SI94] Senay H. und Ignatius E. *A knowledge-based system for visualization design. IEEE Computer Graphics and Applications*, 14:36–47, 1994.

[SKKM+05] Skorin-Kapov L., Komericki H., Matijasevic M., Pandzic I., und Mosmondor M. *Muva: a flexible visualization architecture for multiple client platforms. Journal of Mobile Multimedia*, 1:03–17, 2005.

[SLK+09] Streit M., Lex A., Kalkusch M., Zatloukal K., und Schmalstieg D. *Caleydo: connecting pathways and gene expression. Bioinformatics*, 25(20):2760, 2009.

[SM00] Schumann H. und Müller W. *Visualisierung: Grundlagen und allgemeine Methoden.* Springer, 2000.

[SM04] Schadlich J. und Mukasa K. *An intelligent framework for user-oriented information visualization in the production automation area. iv*, 00:483–487, 2004. ISSN 1093-9547.

[SML04] Schroeder W., Martin K., und Lorensen B. *The Visualization Toolkit An Object-Oriented Approach To 3D Graphics, Kitware*, volume 2. 2004.

[SNLH09] Sips M., Neubert B., Lewis J., und Hanrahan P. *Selecting good views of high-dimensional data using class consistency.* In *Eurographics/ IEEE-VGTC Symposium on Visualization*. 2009.

[SPL06] Singh S., Puradkar S., und Lee Y. *Ubiquitous computing: connecting pervasive computing through semantic web. Information Systems and E-Business Management*, 4(4):421–439, 2006.

[SSK07] Schneidewind J., Sips M., und Keim D.A. *An automated approach for the optimization of pixel-based visualizations. Information Visualization*, 6:75—-88, 2007.

[STH02] Stolte C., Tang D., und Hanrahan P. *Polaris: A system for query, analysis, and visualization of multidimensional relational databases. IEEE Transactions on Visualization and Computer Graphics*, 08(1):52–65, 2002. ISSN 1077-2626.

[STH03] Stolte C., Tang D., und Hanrahan P. *Multiscale visualization using data cubes. IEEE Transactions on Visualization and Computer Graphics*, pages 176–187, 2003.

[SVH08] Schwenke D., Vetro A., und Hata T. *Server-driven progressive image transmission of jpeg 2000.* In *Proceedings of SPIE*, volume 6822, page 68220W. 2008.

[SXX+03] Shi Y., Xie W., Xu G., Shi R., Chen E., Mao Y., und Liu F. *The smart classroom: Merging technologies for seamless tele-education. IEEE Pervasive Computing*, pages 47–55, 2003.

[TA08] Tian D. und AlRegib G. *Batex3: Bit allocation for progressive transmission of textured 3-d models. IEEE Transactions on Circuits and Systems for Video Technology*, 18(1):23–35, 2008.

[TAvHS06] Tominski C., Abello J., van Ham F., und Schumann H. *Fisheye tree views and lenses for graph visualization.* In *IV '06: Proceedings of the conference on Information Visualization*, pages 17–24. IEEE Computer Society, Washington, DC, USA, 2006. ISBN 0-7695-2602-0.

[TFS08a] Thiede C., Fuchs G., und Schumann H. *Smart lenses.* In *Proceedings of the 9th international symposium on Smart Graphics*, pages 178–189. Springer, 2008.

[TFS08b] Tominski C., Fuchs G., und Schumann H. *Task-driven color coding.* In *Information Visualisation, 2008. IV'08. 12th International Conference*, pages 373–380. 2008.

[TG00] Tsoi K. und Gröller E. *Adaptive visualization over the internet.* Technical Report, TU Vienna, 2000.

[TM01] Taubman D.S. und Marcellin M.W. *JPEG 2000: Image Compression Fundamentals, Standards and Practice.* Kluwer Academic Publishers, Norwell, MA, USA, 2001. ISBN 079237519X.

[TM04] Tory M. und Möller T. *Human factors in visualization research. IEEE Transactions on Visualization and Computer Graphics*, 10(1):72–84, 2004. ISSN 1077-2626.

[TS07] Thiede C. und Schumann H. *Beschreibung des kontextes zur adaption visueller interfaces in multimedialen adhoc-umgebungen. Rostocker Informatik-Berichte*, 31:103, 2007.

[TSB04] Tang D., Stolte C., und Bosch R. *Design choices when architecting visualizations. Information Visualization*, 3:65–79, 2004.

[TSR09] Thiede C., Schumann H., und Rosenbaum R. *On-the-fly device adaptation using progressive contents.* In *International Conference on Intelligent Interactive Assistance and Mobile Multimedia Computing (IMC).* November 2009.

[TT85] Tantawi A.N. und Towsley D. *Optimal static load balancing in distributed computer systems. J. ACM*, 32(2):445–465, 1985. ISSN 0004-5411.

[TTS09] Thiede C., Tominski C., und Schumann H. *Service-oriented information visualization for smart environments. Information Visualisation, International Conference on*, 0:227–234, 2009.

[Tuf06] Tufte E. *The Visual Display of Quantitative Information*, volume 53. Graphics Press LLC, 2006.

[VWvH+07] Viegas F.B., Wattenberg M., van Ham F., Kriss J., und McKeon M. *Manyeyes: a site for visualization at internet scale. IEEE Transactions on Visualization and Computer Graphics*, 13(6):1121–1128, 2007. ISSN 1077-2626.

[Wat99] Wattenberg M. *Visualizing the stock market.* In *CHI '99: CHI '99 extended abstracts on Human factors in computing systems*, pages 188–189. ACM, New York, NY, USA, 1999. ISBN 1-58113-158-5.

[WBW96] Wood J., Brodlie K., und Wright H. *Visualization over the world wide web and its application to environmental data.* In *VIS '96: Proceedings of the 7th conference on Visualization '96*, pages 81–ff. IEEE Computer Society Press, Los Alamitos, CA, USA, 1996. ISBN 0-89791-864-9.

[Wea04] Weaver C. *Building highly-coordinated visualizations in improvise. Information Visualization, IEEE Symposium on*, 0:159–166, 2004. ISSN 1522-404X.

[WL90] Wehrend S. und Lewis C. *A problem-oriented classification of visualization techniques. Visualization, 1990. Visualization 90., Proceedings of the First IEEE Conference on*, pages 139–143, 1990.

[WLSS09] Waldner M., Lex A., Streit M., und Schmalstieg D. *Design considerations for collaborative information workspaces in multi-display environments.* In *Proceedings of the CoVIS 2009 Workshop on Collaborative Visualization on Interactive Surfaces (IEEE VisWeek)*. IEEE VisWeek, October 2009.

[WPF04] Winckler M.A., Palanque P., und Freitas C.M.D.S. *Tasks and scenario-based evaluation of information visualization techniques.* In *TAMODIA '04: Proceedings of the 3rd annual conference on Task models and diagrams*, pages 165–172. ACM Press, New York, NY, USA, 2004. ISBN 1-59593-000-0.

[Wur10] Wurdel M. *Spezifikation kontextabhängigen, kooperativen Arbeitens in mobilen Umgebungen (Pre-final).* Ph.D. thesis, Universität Rostock, 2010.

[WWB95] Wood J., Wright H., und Brodlie K. *CSCV-computer supported collaborative visualization.* In *Proceedings of BCS Displays Group International Conference on Visualization and Modelling*, pages 13–25. 1995.

[WWB97] Wood J., Wright H., und Brodlie K. *Collaborative visualization.* In *Proceedings of the 8th conference on Visualization'97.* IEEE Computer Society Press Los Alamitos, CA, USA, 1997.

[YHHC05] Youngblood G., Heierman E., Holder L., und Cook D. *Automation intelligence for the smart environment.* In *INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE*, volume 19, page 1513. 2005.

[YP02] Yang J. und Papazoglou M. *Web components: A substrate for web service reuse and composition. Proceedings of the 14th International Conference on Advanced Information Systems Engineering (CAiSE'02), Toronto, Canada*, 2002.

[ZCF02] Zhou M.X., Chen M., und Feng Y. *Building a visual database for example-based graphics generation. infovis*, 00:23, 2002.

[ZF96] Zhou M. und Feiner S. *Data characterization for automatically visualizing heterogeneous information. Information Visualization, IEEE Symposium on*, 0:13, 1996.

[ZHHM07] Zhao Y., Hu C., Huang Y., und Ma D. *Collaborative visualization of large scale datasets using web services. Internet and Web Applications and Services, International Conference on*, 0:62, 2007.

[ZSY05] Zudilova-Seinstra E. und Yang N. *Towards service-based interactive visualization. Proc. of the International Symposium on Ambient Intelligence and Life*, 00:15–25, July 2005.

Erklärung

Ich erkläre, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Rostock, 18. Januar 2011

Thesen

1. *Smart-ad-hoc-Environments* erweitern die *Multi-Display-Umgebungen* um zwei wichtige Aspekte *„Smart“* und *„ad-hoc“*. Der Aspekt *Smart* beschreibt in diesem Zusammenhang die Fähigkeit, selbständig Daten zu sammeln und anzuwenden. Der Begriff *ad-hoc* impliziert dabei eine gewisse Dynamik im System.

2. Informationsvisualisierung in Smart-ad-hoc-Environments stellen eine besondere Herausforderung dar, da eine gleichzeitige Ausgabe auf heterogenen, sich dynamisch verändernden Geräten erforderlich ist.

3. Um die Heterogenität der Ausgabegeräte zu adressieren, müssen geeignete Anpassungen der Informationsdarstellung vorgenommen werden, dabei wird unterschieden zwischen Adaption auf der Bildebene vs. Adaption auf der Prozessebene, zeitliche Adaption vs. räumliche Adaption und lokale Adaption vs. globale Adaption.

4. Zur Adaption auf der Bildebene, bei der nur die visuelle Repräsentation vorliegt, wurde das Konzept der *inhaltsbasierten Skalierung* für Informationsvisualisierungen entwickelt. Dieses vermeidet eine Skalierung von wichtigen Bildbereichen und entfernt dafür im Gegenzug unwichtige Bildbereiche vollständig. Dadurch bleibt der wesentliche Bildinhalt unverändert erhalten und kann besser durch den Benutzer interpretiert werden.

5. Bei der globale Adaption steht der gesamte Erzeugungsprozess der visuellen Repräsentation für eine Adaption zur Verfügung. Entsprechend können unterschiedliche *Adaptionsmechanismen* eingesetzt werden. Prinzipiell lassen sich diese unterteilen in *Adaptionsmechanismen*, die im Datenraum arbeiten, welche, die im Darstellungsraum arbeiten und *Adaptionsmechanismen*, die das Mapping beeinflussen.

6. Als Konzept für eine zeitliche Adaption wurde eine progressive Informationsdarstellung entwickelt. Hierbei wird die visuelle Repräsentation schrittweise verfeinert und dabei die Zwischenergebnisse auf den Ausgabegeräten dargestellt bis der maximale Detailgrad erreicht ist. Die Idee der *progressiven Informationsdarstellung* zur Adaption und heterogene Ausgabegeräte ist, die Übertragung für jedes Ausgabegerät zu stoppen, wenn noch feinere Informationen nicht mehr darstellbar sind. Diese Bedingung wird *On-the-Fly* auf der Basis von bereits übertragenen Zwischenergebnissen abgeschätzt und geprüft.

7. Es wurden *intelligente Linsen* als ein Konzept zur lokalen Adaption entwickelt. Dabei wird nur ein ausgewählter Bereich von Interesse durch die Linse entsprechend einer gewählten Linsenfunktion angepasst. Prinzipiell lassen sich Linsen auf allen Stufen der Informationsvisualisierung einsetzen. Intelligente Linsen zeichnen sich dadurch aus, dass der Bereich von Interesse und die Funktion zur Adaption automatisch gewählt wird. Weiterhin lassen sich vergleichsweise komplexe Linsenfunktionen durch das Kombinieren von einfachen Linsenfunktionen realisieren.

8. Zur Realisierung dieser Konzepte wurde ein serviceorientiertes Visualisierungsframework entwickelt, um adaptierte visuelle Repräsentationen zu erzeugen. Bei diesem wurden insbesondere die multiplen, dynamischen und heterogenen Ausgabegeräte berücksichtigt. Als Basis für die einzelnen Services dienen die Operatoren des Data-State-Reference-Models.

9. Die Frameworkarchitektur gliedert sich im Wesentlichen in zwei Schichten, die Serviceschicht stellt die Services bereits aus der die Visualisierungspipeline *On-the-Fly* zusammengesetzt wird. Dieser Prozess wird durch die Steuerschicht realisiert, welche dabei die Auswahl und die Verknüpfungen der Services steuert und bei Veränderungen des Geräteensembles bei Bedarf eine Adaption der visuellen Repräsentation einleitet.

10. Die multiplen Ausgabegeräte werden bei Bedarf durch eine Verzweigung der Visualisierungspipeline bedient. Es werden jeweils die Eigenschaften des Ausgabegerätes berücksichtigt, indem durch die Steuerschicht eine Anpassung der Visualisierungspipeline und damit eine Adaption der visuellen Repräsentation durchgeführt wird.

Wissenschaftliche Veröffentlichungen

- Conrad Thiede, Heidrun Schumann, René Rosenbaum: *On-the-fly device adaptation using progressive contents.* International Conference on Intelligent Interactive Assistance and Mobile Multimedia Computing (IMC) **2009**

- Georg Fuchs, Conrad Thiede, Mike Sips, Heidrun Schumann: *Device-based Adaptation of Visualizations in Smart Environments.* Workshop Collaborative Visualization on Interactive Surfaces (CoVIS) IEEE VisWeek, **2009**

- Conrad Thiede, Christian Tominski, Heidrun Schumann: *Service-Oriented Information Visualization for Smart Environments.* Proceedings, IEEE Computer Society, IV09 - 13th International Conference on Information Visualisation, Barcelona, Spain, **2009**

- Rene Rosenbaum, Martin Luboschik, Conrad Thiede, Heidrun Schumann: *Progressive Information Visualization Interactive Poster.* IEEE Information Visualization InfoVis'2008, Ohio, USA, Oct. **2008**

- Conrad Thiede, Georg Fuchs, Heidrun Schumann: *Smart Lenses.* 8th International Symposium on Smart Graphics, August 27th – 29th **2008**, Rennes, France.

- Conrad Thiede, Heidrun Schumann: *Beschreibung des Kontextes zur Adaption visueller Interfaces in multimedialen adhoc-Umgebungen.* Rostocker Informatik-Berichte, **2007**, Volume 31, Page 103

- Georg Fuchs, Conrad Thiede, Heidrun Schumann: *Pluggable Lenses for Interactive Visualizations.* Poster Compendium of InfoVis, October **2007**