Bauhaus-Universität Weimar
Informatik in der Architektur | InfAR

Arbeitspapiere
Working Papers

Nr. 5, April 2011

Entwerfen Versionieren

Sven Schneider, Jörg Braunes,
Torsten Thurow, Reinhard Koenig,
Christian Tonn

ISSN 2191-2416

Bauhaus-Universität Weimar, Professur Informatik in der Architektur, Belvederer Allee 1, 99425 Weimar
Fon: +49/3643/584201, caad@architektur.uni-weimar.de, http://infar.architektur.uni-weimar.de

Sven Schneider, Jörg Braunes, Torsten Thurow, Reinhard Koenig, Christian Tonn
Entwerfen Versionieren:
Probleme und Lösungsansätze für die Organisation verteilter Entwurfsprozesse

Weimar 2011
Arbeitspapiere (Working Papers) Informatik in der Architektur,
Bauhaus Universität Weimar, Nr. 5
Herausgegeben von Prof. Dr. Dirk Donath und Dr. Reinhard König
ISSN 2191-2416

Bauhaus-Universität Weimar, Professur Informatik in der Architektur
Belvederer Allee 1, 99425 Weimar
http://infar.architektur.uni-weimar.de

Redaktionelle Anmerkung:

Dipl.-Ing. Sven Schneider, Dipl.-Ing. Jörg Braunes, Dr. Torsten Thurow und Dipl.-Ing. Christian, Tonn sind wissenschaftliche Mitarbeiter an der Professur Informatik in der Architektur an der Bauhaus-Universität Weimar. Dr. Reinhard König ist Vertretungsprofessor der Professur Informatik in der Architektur an der Bauhaus-Universität Weimar.

Der vorliegende Text wurde in englischer Sprache bei der Konferenz „CAAD-Futures 2011“ eingereicht.

Entwerfen Versionieren

Probleme und Lösungsansätze für die Organisation verteilter Entwurfsprozesse

Sven Schneider[1], Jörg Braunes[2], Torsten Thurow[3], Reinhard Koenig[4], Christian, Tonn[5]

[1]sven.schneider@uni-weimar.de, [2]joerg.braunes@uni-weimar.de, [3]torsten.thurow@uni-weimar.de, [4]reinhard.koenig@uni-weimar.de, [5]christian.tonn@uni-weimar.de

Abstract

Entwerfen ist ein komplexer Vorgang. Soll dieser Vorgang nicht allein, sondern räumlich verteilt mit mehreren Beteiligten gemeinsam stattfinden, so sind digitale Werkzeuge zur Unterstützung dieses Prozesses unumgänglich. Die Verwendung von Werkzeugen für Entwurfsprozesse bedeutet jedoch immer auch eine Manipulation des zu unterstützenden Prozesses selbst. Im Falle von Werkzeugen zur Unterstützung der Kollaboration mehrerer Beteiligter stellen die implementierten Koordinationsmechanismen solche prozessbeeinflussenden Faktoren dar. Damit diese Mechanismen, entsprechend der Charakteristika kreativer Prozesse, so flexibel wie möglich gestaltet werden können, liegt die Anforderung auf technischer Ebene darin, ein geeignetes Konzept für eine nachvollziehbare Speicherung (Versionierung) der stattfindenden Entwurfshandlungen zu schaffen.

Der vorliegende Artikel beschäftigt sich mit dem Thema der Entwurfsversionierung in computergestützten kollaborativen Arbeitsumgebungen. Vor dem Hintergrund, dass die Versionierung den kreativen Entwurfsprozess möglichst wenig manipulieren soll, werden technische sowie konzeptionelle Probleme der diskutiert und Lösungsansätze für diese vorgestellt.

Keywords: Kollaboratives Entwerfen; Versionierung; computerbasierte Entwurfsprozesse; digitale Produktmodellierung; Freac.

1. Zusammen-Entwerfen organisieren

Entwerfen Versionieren meint im vorliegenden Kontext das Speichern verschiedener Entwurfsvarianten, welche mittels digitaler Werkzeuge erstellt wurden. Folglich befassen wir uns in diesem Beitrag ausschließlich mit computerbasierten Entwurfsprozessen. Unsere Auseinandersetzung konzentriert sich auf Methoden zum Erstellen und Verwalten von Varianten während dieser Prozesse.

Entwurfsprozesse sind grundsätzlich von arbeitsteiligem Charakter. So müssen in einer Vielzahl von Entwurfsschritten verschiedenste Aspekte (formale, funktionale, ökologische ökonomische, rechtliche, städtebauliche, etc.) berücksichtigt und aufeinander abgestimmt werden. Diese Aspekte werden darüber hinaus von unterschiedlichen Beteiligten (Architekt, Fachplaner, zukünftige Nutzer, Anwohnern, Denkmalschützern, Bauherren, Behörden, etc.) unterschiedlich wahrgenommen und bewertet (Latour & Yaneva, 2008; Tellioglu, 2009)). Die zentrale Herausforderung bei kollaborativen Planungsprozessen besteht nun darin, die unterschiedlichen Aspekte und Interessen frühzeitig zu integrieren und dabei das Wissen und die Kompetenzen der Beteiligten effektiv zu nutzen. Aus ökonomischen Gründen finden kollaborative Prozesse zunehmend räumlich (und zeitlich) verteilt statt. Der Erfolg solcher räumlich verteilter Zusammenarbeit hängt in hohem Maße von den eingesetzt Medien ab (Maher, Bilda, & GÜL, 2006). Die Qualität des kollaborativen Prozesses beeinflusst die Qualität der Planung, sprich: wie fließen Beiträge der unterschiedlichen Beteiligten ein, kommen die Absichten bei anderen Beteiligten an, wie können die Beteiligten miteinander in Interaktion treten, etc. Einer effektiven Kommunikation und Koordination zwischen den einzelnen Akteuren fällt eine zentrale Rolle zu, um den kollaborativen Prozess zielführend zu unterstützen.

Aufgrund der Komplexität von Entwurfsprozessen sind innovative Managementansätze für erfolgreiches kreativ-kollaboratives Arbeiten erforderlich (Sebastian, 2007). Betrachten wir jedoch aktuelle technische Entwicklungen im Bereich des kollaborativen Arbeitens mit CAAD-Systemen, so stellt zwar das Konzept der Bauwerks-Informations-Modellierung (BIM) die integrativen Aspekte eines gemeinsamen Modells in den Mittelpunkt, bleibt aber in seiner Ausprägung in alten Mustern verhaften. BIM-Systeme konzentrieren sich vorwiegend auf einen effektiven Datenaustausch und verlangen bei verteilter Planbearbeitung eine strikte Rollenverteilung (siehe Punkt 2). Das mag auch sinnvoll sein in fortgeschrittenen Phasen eines Projektes, da es den Informationsverlust gering hält und damit vor Missverständnissen und erhöhten Planungskosten vorbeugt, zum Entwickeln von Konzepten und Ideen, welche entscheidenden Einfluss auf das endgültige Produkt nehmen, ist ein rei-

ner Datenaustausch nur von geringem Nutzen. So haben sich zwar die Austauschzeiten zwischen den verschiedenen Akteuren verkürzt, die Bearbeitung selbst erfolgt jedoch weiterhin stark sequentiell. Bezugnehmend auf den Prozess des Entwerfens, welcher sich typischerweise mit komplexen, oder besser gesagt "wicked problems" (Rittel & Webber, 1973) beschäftigt, sind solche aprioi hierarchisch organisierten Strukturen eher hinderlich.

So zeichnet sich kreative Gruppenarbeit vor allem durch eine hohe Dynamik bei der Beteiligung verschiedener Akteure, die Unvorhersehbarkeit bestimmter Handlungen und eine große Vielfalt an entstehenden Lösungen aus. Der Ablauf eines Entwurfsprozesses folgt dabei keinen vordefinierbaren Schemata. Es existieren keine klaren Start- und Endbedingungen für einen solchen Vorgang. Das zu entwerfende Artefakt und die mit ihm verbundenen Ziele, Rahmenbedingungen und Organisationsstrukturen werden oft erst während des Entwurfsprozesses ausgehandelt*: "The major challenge to CSCW emerging from this is the very fact that ordering systems are constructed and maintained in a cooperative process"* (Schmidt & Wagner, 2004). Hierfür sind neue Managementmethoden gefragt, die die Charakteristik kreativer Prozesse akzeptieren (Sebastian, 2007). Bei der Ausgestaltung dieser Methoden kommt es darauf an, Entwurfsprozesse offen zu halten, die an einer Planung Beteiligten zur Mitarbeit zu motivieren und entsprechende technische Schwellen zu reduzieren mit dem Ziel, möglichst alle Kompetenzen der verschieden Beteiligten effektiv in eine Planung zu integrieren. Um ein solches Netzwerk zu koordinieren, dass aus unterschiedlichen Werkzeugen, Inhalten und Nutzern besteht, ist es notwendig die darin stattfindenden Handlungen abbilden und nachverfolgen zu können. Voraussetzung dafür ist die Abspeicherung bzw. Versionierung des gesamten Entwicklungsprozesses einer Planung. Um eine solche Versionierung adäquat in den kollaborativen Entwurfsprozess einzubinden, müssen bei ihrer Konzeption bestimmte entwurfsspezifische Charakteristika berücksichtigt werden:

(1) Entwerfer kommunizieren über Repräsentationen des zu entwerfenden Gegenstandes (Skizzen, Modelle, Zeichnungen, Text, etc.). Diese Repräsentationen werden interpretiert, überprüft und kontinuierlich weiterentwickelt (Tellioglu, 2009).

(2) Um die Repräsentationen zu bearbeiten werden verschiedene Werkzeuge verwendet. Diese werden aufgrund ihrer jeweils begrenzten Funktionalitäten situationsbedingt gewechselt.

(3) Während dieses Bearbeitungsprozesses entstehen Varianten und Alternativen eines Entwurfs. Dabei ist zu beachten, dass zu einem bestimmten Zeitpunkt mehrere Varianten und Alternativen parallel existieren können.

(4) Der Zeitpunkt für die Speicherung eines bestimmten Entwurfsstandes ist nicht klar definierbar. Auch Zwischenstände können kreative Prozesse und Diskussionen in einer Gruppe anstoßen. Verwiesen sei hier auf die reflexive Praxis des Entwerfens (Schön, 1983) sowie die Beeinflussung des Denkens durch die Wahrnehmung (Arnheim, 1969).

(5) Es ist zu beachten, dass der Entwurfsprozess *"combines slow reflection with intense periods of very rapid mental activity as the designer tries to keep many things in mind at once."* (Lawson, 2005). Demzufolge gibt es auch bei kollaborativen Entwurfsprozessen sowohl asynchrone (mehrere Bearbeiter nacheinander) als auch synchrone (mehrere Bearbeiter gleichzeitig) Bearbeitungsphasen. Also Phasen, in denen der Einzelne besser ohne weitere Beteiligung arbeitet, als auch Phasen in denen Probleme gelöst werden müssen, welche intensiver Absprachen bedürfen. Zwischen beiden Formen wird in der Regel situationsbedingt gewechselt.

(6) Es ist offensichtlich, dass Teamwork im Design nicht bedeutet, dass jeder für alles zuständig ist. Dennoch ist zu beachten, dass jeder Einfluss auf die Arbeitsbereiche der Anderen ausübt. Da sich die Teilsysteme eines Entwurfes in vielen Bereichen überlagern, ist es nicht möglich, klar voneinander abtrennbare Zuständigkeitsbereiche für Teile eines Entwurfs zu definieren.

Im Folgenden soll der Stand der Forschung zur technischen Basis für räumlich verteilte Zusammenarbeit dargelegt werden. Die betrachteten Methoden und Systeme werden hinsichtlich der zuvor genannten Anforderungen bewertet.

2. Verteilte Systeme im Kontext von Entwurfsprozessen

Die kollaborative Arbeit an einem Projekt erfordert die Nutzung einheitlicher Datenstandards und effektiver Techniken des Datenmanagements. Der dateibasierte Austausch von Daten und die manuelle Verwaltung von Versionen und Varianten dominiert derzeit noch bei Planungsprojekten (Benning et al., 2010). Während die manuelle Verwaltung der Daten geprägt ist von Redundanzen und Fehlern, können Daten-Repositorien wie Subversion[1] - welches u.a. in Digital Project von Gehry Technologies Anwendung findet - oder Product Data Management (PDM) Systeme die Datenverwaltung effizienter gestalten.

Arbeiten unterschiedliche Applikationen gleichzeitig an einem Datenmodell, ergibt sich ein nicht linearer Workflow: Die Applikationen haben sowohl lesenden als auch schreibenden

[1] http://subversion.apache.org/

Zugriff, bearbeiten in der Regel aber immer nur einen fachspezifischen Teil des Modells. Diese Bearbeitung erfolgt zudem parallel, so dass die Notwendigkeit einer Datenkoordination besteht. Derzeit bieten Model-Server die beste Möglichkeit zur Unterstützung eines solchen parallelen Arbeitsprozesses. Diese arbeiten objektbasiert und erlauben die Abfrage, Übertragung und Aktualisierung spezifischer Objekte eines gemeinsam verwalteten Modells. Aktuelle Beispiele für Model-Server sind der Open Source BIM Server[2], der Jotne EPM EDMserver[3] oder der Eurostep BIM Collaboration Hub [4].

Als standardisiertes Objekt-Schema für Gebäudemodelle bilden die Industry Foundation Classes (IFC) die Grundlage solcher Model-Server (Eastman, Eastman, Teicholz, & Sacks, 2008). Eine aus einer BIM-Applikation erstellte IFC-Datei wird dabei in einen zentralen Server geladen und dort objektbezogen in einer relationalen Datenbank abgelegt. Aus dieser Datenbank können spezifische Modellinhalte wieder als IFC-Datei extrahiert werden, um diese in anderen Applikationen weiterbearbeiten zu können. Beim erneuten Upload in den Model Server organisiert dieser die Zusammenführung der Daten, um einen konsistenten Modellzustand herzustellen (Abb. 1). Das Zusammenführung von Teilmodellen im Model-Server erfordert ein Versions- und Änderungsmanagement auf Objekt Level (Nour et al., 2010). Das IFC Schema bietet hierfür den Mechanismus der IfcOwnerHistory, welcher das Erstellungsdatum, die Ursprungsapplikation sowie Änderungsdatum und Änderungsapplikation objektbezogen speichert (*IFC2x Edition 3 Technical Corrigendum 1*, 2007).

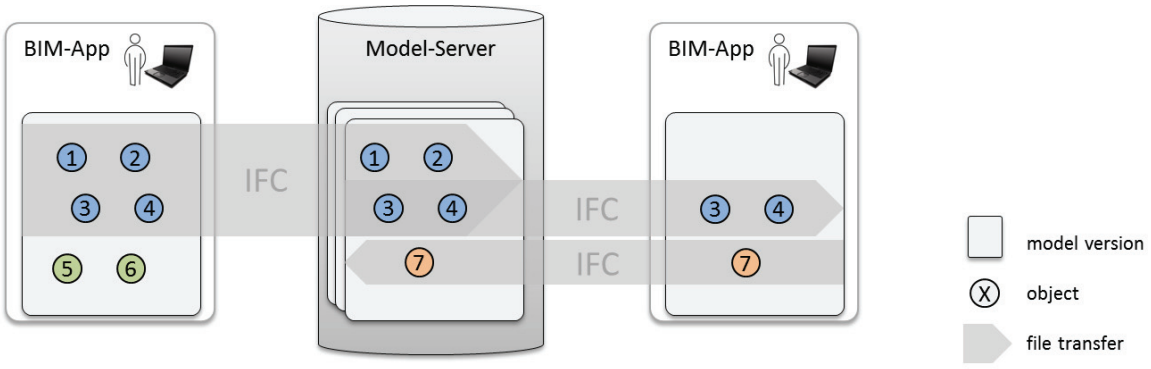

Abb. 1: Prinzip eines Model Server: Modellteile werden per IFC an den Server übergeben bzw. von diesem abgerufen. Übertragen werden nur die Modellteile, welche das Austauschformat unterstützt.

[2] http://code.google.com/p/bimserver/
[3] http://www.epmtech.jotne.com/
[4] http://www.eurostep.com/global/solutions/bim-collaboration-hub.aspx

Aufgrund dieser Funktionsweise erlauben Model Server lediglich eine asynchrone Zusammenarbeit verschiedener Applikationen. Die Notwendigkeit der Verwendung eines allgemeingültigen Austauschformates wie die IFC erfordert zudem immer eine Übersetzung der Daten aus dem applikationsspezifischen Gebäudemodell in das IFC-Format. Diese Übersetzung erweist sich als schwierig, da die Konsistenz der Daten oftmals nicht gewährleistet ist und sich bestimmte Modellinhalte in der IFC nicht abbilden lassen (bspw. Redlining Funktion und Kommentare/Anmerkungen).

Einige BIM-Applikationen unterstützen eigene Model-Server Funktionalitäten, indem direkt ein proprietäres Datenmodell verwaltet wird. Ein Beispiel hierfür ist die ArchiCad Teamworks Funktionalität auf Basis des Graphisoft BIM-Servers[5]. Teamwork Projekte werden hier zentral im BIM-Server vorgehalten. Gleichzeitig wird das Projekt auf dem Client Rechner lokal gespeichert, um ein schnelles Arbeiten zu gewährleisten. Werden lokal Änderungen am Projekt vorgenommen, müssen diese manuell zum Server gesendet bzw. von diesem abgerufen werden. Die zu bearbeitenden Projektteile werden jeweils vom Client reserviert und sind damit für alle anderen Teammitglieder gesperrt. Änderungen an einem Modell durch dritte werden jeweils erst nach manueller Aktualisierung des lokalen Modells für einen Bearbeiter sichtbar. Die Kommunikation der Teammitglieder untereinander erfolgt über ein integriertes Instant-Messaging-System. Das Arbeiten mit Projektvarianten und Alternativen wird durch den Graphisoft BIM-Server nicht unterstützt. Das Sperren und Reservieren von Projektbestandteilen forciert eine lineare Projektbearbeitung, die parallele Bearbeitung des gleichen Projektbereiches, und damit die Arbeit in verschiedenen Alternativen, ist nicht vorgesehen.

Für die synchrone Projektbearbeitung bietet Autodesk seit einiger Zeit den freien Service AutoCad WS[6]. Dieser erlaubt die gleichzeitige Bearbeitung von CAD-Daten durch mehrere Nutzer via Internet. Die Dateien werden auf einen Webserver geladen und können über einen Webbrowser online in Echtzeit bearbeitet werden. Arbeiten mehrere Nutzer gleichzeitig an einer Datei sind Änderungen sofort sichtbar. Die synchrone Bearbeitung von Zeichnungselementen ist für jeden Nutzer möglich, ein Sperren oder Reservieren von Elementen ist nicht vorgesehen. Über eine Zeitleiste können vorhergehende Dateirevisionen abgerufen werden. Der Bearbeitungsprozess bleibt jedoch linear, das Anlegen von Varianten und Alternativen wird nicht unterstützt. Einen anderen Ansatz zur verteilten Echtzeitbearbeitung unter Einsatz verschiedener Applikationen bietet das Projekt Uni-Verse[7].

[5] http://www.graphisoftus.com/
[6] www.autocadws.com
[7] www.uni-verse.org (Das Projekt Uni-Verse gilt als abgeschlossen und wird derzeit nicht weiterentwickelt.)

Grundlage hierfür bildet "Verse", ein Netzwerkprotokoll zur Verteilung von 3D Daten zwischen verschiedenen Applikationen. Über Plug-Ins können Applikationen mit dem Verse-Server kommunizieren, welcher Modelländerungen an alle Clients in Echtzeit weiterreicht. Derzeit existieren Plug-Ins für 3ds Max, Blender und Gimp. Ein Versions- und Variantenmanagement wird durch das Protokoll nicht unterstützt.

3. Einschätzung der Systeme

Aus der Betrachtung bestehender Systeme und Methoden zur räumlich verteilten Kollaboration sind große Diskrepanzen zwischen den Anforderungen an ein ideales System (siehe Punkt 1) und dem derzeitigen Stand der Technik zu erkennen. Der dabei wohl auffälligste Punkt ist die Linearität der Systeme. So ist in keinem der Systeme eine tatsächlich parallele Bearbeitung eines gemeinsamen Modells möglich. Stattdessen wird die "Zusammenarbeit" vorgetäuscht, indem beispielsweise die zu bearbeitenden Modellbereiche reserviert werden müssen und damit den Handlungsraum der anderen Akteure einschränken. Diese stehen dann nur einem einzigen Bearbeiter zur Verfügung, bis dieser den entsprechenden Teilbereich wieder freigibt.

Wie in Punkt 1 dargestellt sind Entwurfsaufgaben in aller Regel sehr komplex, und die entsprechenden Arbeitsprozesse sind von grundsätzlich nichtlinearem Charakter. Das bedeutet beispielsweise, dass sich unterschiedliche Teilbereiche einer Entwurfsaufgabe überlagern und diverse Abhängigkeiten erzeugen, oder dass Detailprobleme teilweise Lösungsansätze auf anderen Abstraktionsebenen erfordern. Dieses komplexe Zusammenspiel kann nur durch eine effektive Zusammenarbeit bewältigt werden. Eine Einschränkung, wie die in den oben beschriebenen Systemen verwendete sequentielle Speicherung von Entwurfsänderungen (Transaktionen) führt jedoch unweigerlich zu einer Linearisierung des Bearbeitungsprozesses. Ein kreatives "Zusammen-arbeiten" weicht einem maschinellen "Nacheinander-arbeiten". Der kreative Workflow und die Beschäftigung mit den eigentlichen Entwurfsproblemen weicht den Fragen: Wer arbeitet zuerst? Wer kommt als nächstes? Und wer bearbeitet was?

Soll kollaborative Entwurfsarbeit tatsächlich durch digitale Entwurfssysteme unterstützt werden, sind neue nichtlineare Ansätze nötig, welche Entwurfsprozesse nicht als apriori organisierte, sondern als offene netzwerkartige Abläufe begreifen (Schneider, 2010). Für eine Umsetzung solcher Modelle liegt das Kernproblem jedoch auf einer tieferliegenden technischen Ebene: Dem Austausch und dem Management von Daten! Im Folgenden soll

hierfür ein technisches Framework vorgestellt werden, dass aufgrund seiner Konzeption eine größtmögliche Flexibilität bei der Nutzung als kollaboratives Entwurfssystem bietet.

4. Technisches Framework

Für die Softwareentwicklung nutzen wir das selbst entwickeltes Framework FREAC (Koenig, 2010). Den Kern von FREAC bildet ein zur Laufzeit dynamisch veränder- und erweiterbares Produktmodell. Produktmodelle stellen eine abstrakte Definition der physischen und funktionalen Aspekte eines Produktes dar (entsprechend Definition nach ISO 10303). Sie nutzen zu ihrer Modellierung das objektorientierte Paradigma, bestehen also aus Klassen und den von ihnen instanziierten Objekten. Ein dynamisches Modell erfordert, dass bestimmte Strukturen modifiziert bzw. erweitert werden müssen. Diese Modifikationen umfassen dabei u.a. das Hinzufügen und Ändern von Klassen, Attributen und Methoden sowie Objekten. In FREAC werden verschiedene Aspekte in Teilmodellen organisiert (z.B. Geometriemodell, Bauwerksmodell, Skizzenmodell), welche sich gegenseitig referenzieren können (Abb. 2). Neben der flexiblen Ergänzung des Kerns um neue Teilmodelle, ist auch die Anpassung und Erweiterung bestehender Teilmodelle zur Laufzeit möglich.

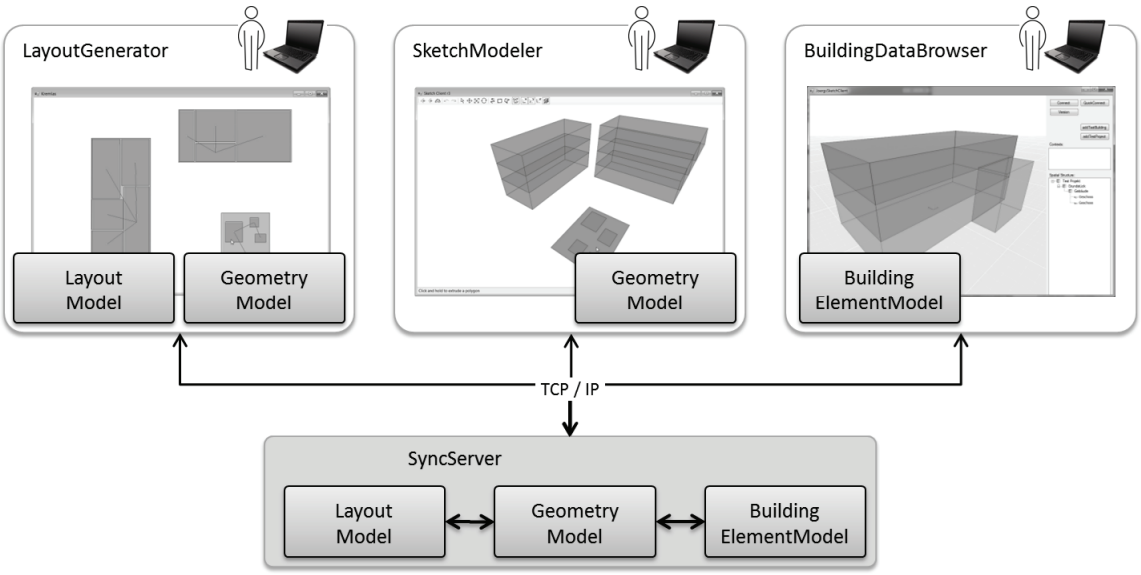

Abb. 2: Teilmodelle und Interaktion zwischen Clients und dem SyncServer.

Die Basis des Frameworks bildet die Bibliothek „DOMMC.Net“. Diese liefert alle Funktionalitäten zur versionierten, persistenten Speicherung und Verteilung der darauf aufbauenden Modelle. FREAC nutzt dabei Techniken wie Transaktionen und Versionierungen mit Online-Synchronisation, welche die Speicherung der Historie von Modellen sowie ihre quasi-parallele Bearbeitung erlauben. Ein FREAC-Modell wird auf einem lokalen Computer mittels Anwendungen für spezielle Aufgaben, den Clients, bearbeitet. Die Modelldaten liegen

auf einem zentralen Server (SyncServer), welcher Modelländerungen in Echtzeit auf alle verbundenen Clients überträgt sowie die Versionierung automatisch organisiert (Abb. 3). Anhand des Clientkonzepts wird eine reibungslose Vernetzung verschiedener Projekte ermöglicht, die, aufgrund der FREAC eigenen Dynamik, unabhängig voneinander entwickelt werden können.

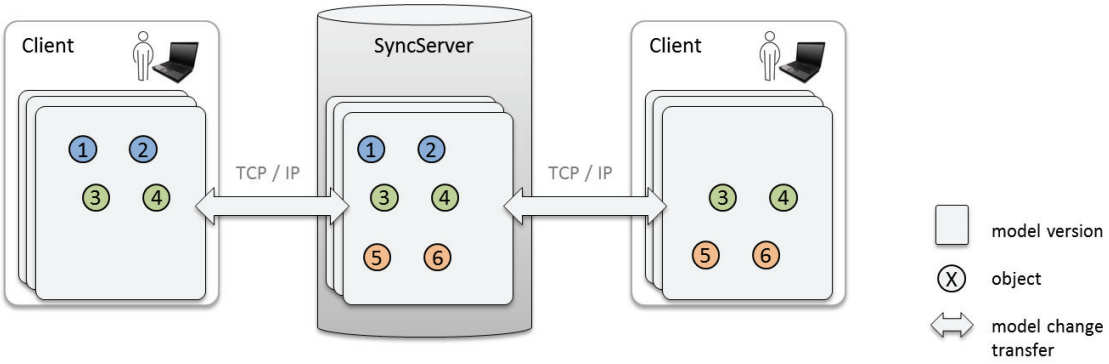

Abb. 3: Der SyncServer überträgt in Echtzeit die Modelländerungen auf alle verbundenen Clients. Der Zugriff auf Modellversionen ist in den Clients selbst möglich.

Wie im technischen Teil dieses Artikels bereits erläutert, erlaubt die dynamische Modellstruktur des FREAC-Development jederzeit das Ergänzen und Anpassen von Teilmodellen ohne die Notwendigkeit der Recompilierung bereits bestehender Teilmodelle. Damit können die FREAC-Tools jederzeit um neue Softwareprototypen (Abb. 4) für neue Anwendungsfälle ergänzt werden. Neu entwickelte Clients können dabei sowohl bestehende, wie auch neue Teilmodelle nutzen.

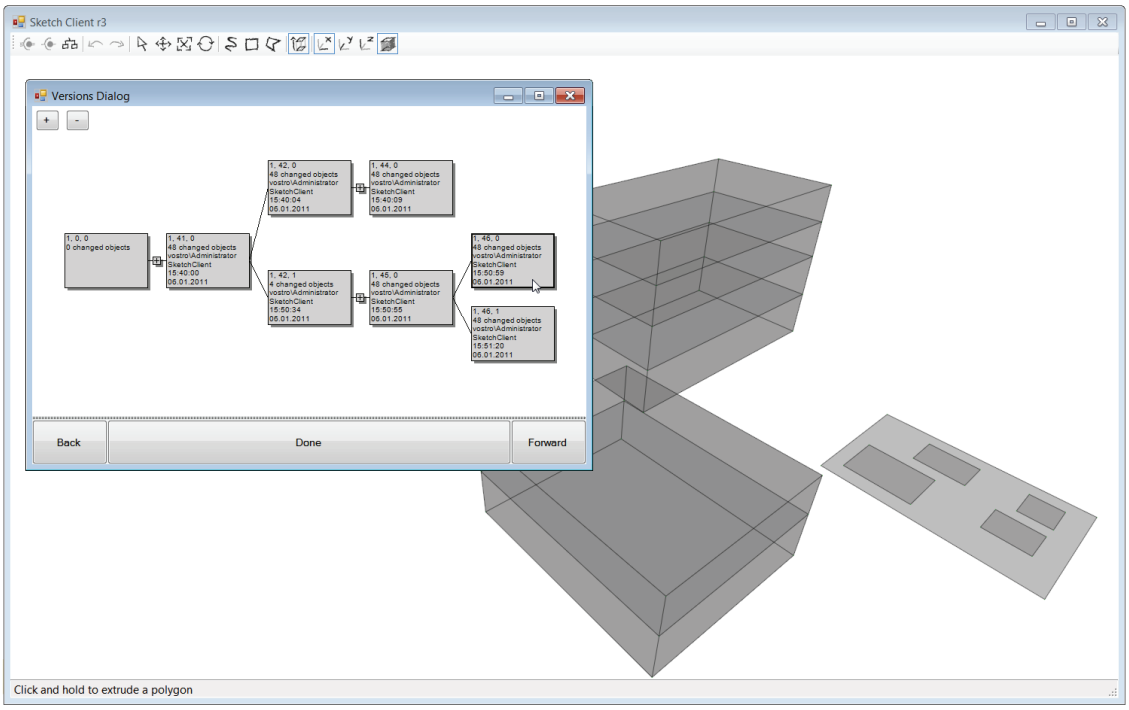

Abb. 4: Beispiel des Versionsdialogs anhand eines einfachen Modellier-Clients.

5. Ein universelles Versionierungskonzept

Die Verwendung und Schaffung von Werkzeugen für Entwurfsprozesse bedeutet immer auch einen Eingriff bzw. eine Manipulation der zu unterstützenden Prozesses selbst. Im Falle von Werkzeugen zur Unterstützung der Kollaboration mehrerer Beteiligter, stellen die implementierten Koordinationsmechanismen einen dieser prozessbeeinflussenden Faktoren dar. Damit diese Mechanismen, entsprechend der Charakteristika kreativer Prozesse, so flexibel wie möglich gestaltet werden können, liegt die Anforderung auf technischer Ebene darin, ein Versionierungskonzept zu schaffen, das so allgemeingültig wie möglich ist.

Innerhalb des FREAC Frameworks drücken sich Modelländerungen als Entstehung, Löschung oder Veränderung von Objekten aus. Eine Modelländerung muss den Modellinhalt immer von einem konsistenten Zustand in einen neuen konsistenten Zustand überführen. Diese Änderung ist quasi atomar und wird daher als Transaktion betrachtet. Bei dem von den Autoren genutzten Ansatz wird ein konsistenter Modellzustand als Version bezeichnet und mittels einer Transaktion zu einer neuen Version verändert (Abb. 5). Die Versionen, also der Zustand der Objekte eines Modells, werden persistent gespeichert, so dass die Entwicklungsgeschichte des Modells ebenfalls persistent gespeichert ist. Das Datenvolumen wird reduziert, indem nur Modelländerungen gespeichert werden.

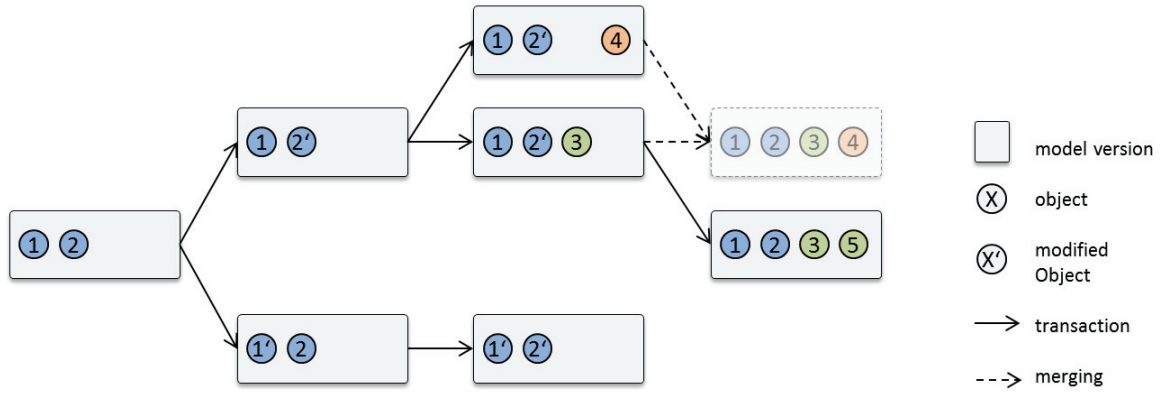

Abb. 5: Prinzip der Versionierung durch Transaktionen.

Eine Version kann hierbei beliebig viele direkte Vorgänger- und Nachfolgeversionen besitzen. Besitzt eine Version mehrere direkte Nachfolgeversionen, so stellen diese alternative Modellveränderungen bzw. -zweige dar, während die Zusammenführung mehrerer direkter Vorgängerversionen zu einer Version eine Zusammenführung parallel verlaufener Modelländerungen (Abb. 5), ein Merging, darstellen. Merging beinhaltet eine Vielzahl von Problemen, welche unter Punkt 5 beschrieben werden. Um Merging bei synchroner Zusammenarbeit möglichst zu vermeiden, wird die Parallelisierung der Bearbeitung permanent pseu-

doparallel auf Basis kurzer Transaktionen ausgeführt, vergleichbar dem Multitaskingansatz von Einprozessorsystemen. Als kurze Transaktionen werden dabei Transaktionen bezeichnet, deren Ausführungszeiten für den Nutzer als nicht verzögernd wahrgenommen werden. Die Synchronisation der parallel arbeitenden Clients erfolgt dabei nach einem einfachen Tokenansatz. Ansätze langer Transaktionen sollen in späteren Arbeitsschritten untersucht und gegebenenfalls ergänzt werden.

Um Netzlasten zu reduzieren und die Antwortzeiten während der Objektzugriffe zu erhöhen, werden bewusst keine entfernten Methodenaufrufe verwendet, sondern die Objektinhalte einmal nach Abschluss einer Transaktion auf allen Clients synchronisiert. Methodenaufrufe erfolgen so immer auf dem lokalen Rechner. Dieser Ansatz ist dann optimal, wenn während der Transaktionen nur eine geringe Anzahl von Objekten verändert wird. Entscheidend ist dabei verständlicherweise nur die Anzahl der Objekte, nicht die Vielzahl der Veränderungen an einem einzelnen Objekt. Ergänzend zur persistenten Speicherung der Objektinhalte innerhalb der Versionen können diesen auch Zusatzinformationen wie beispielsweise der Bearbeiter, Bearbeitungszeit und durchgeführte Operation zugeordnet werden. Letztere spielen insbesondere für Merging eine große Rolle.

6. Automatisches Merging

Merging ist vor allem in der Softwareentwicklung ein bekannter Begriff. Hierbei werden Quellcodefiles parallel von unterschiedlichen Entwicklern modifiziert und die Änderungen dann wieder zusammengeführt. Dabei werden der pessimistische und der optimistische Ansatz unterschieden. Bei dem pessimistischen Ansatz werden jeweils bestimmte Quellcodefiles einem Entwickler zur Bearbeitung zugewiesen, für alle anderen Entwickler ist eine Veränderung gesperrt. Bei einem optimistischen Ansatz wird davon ausgegangen, dass die Veränderungen der Entwickler in der Regel nicht kollidieren. Sollte doch eine Kollision eingetreten sein, so wird zunächst ein automatisches Zusammenführen (Merging) der Veränderungen versucht, gelingt dies nicht, so muss die Zusammenführung manuell erfolgen.

Der genannte Ansatz wird nun auf die Ebene des Objektorientierten Paradigma (OOP) übertragen, anstatt Files werden hier Objekte verändert. Im Gegensatz zu Quellcodefiles lassen sich Objektinhalte (im Sinne der OOP) praktisch nicht manuell zusammenführen. Eine automatische Zusammenführung ist jedoch keinesfalls trivial. Allein aus den Änderungen der Objektinhalte der zusammenzuführenden Objekte einen zusammengeführten Objektinhalt ableiten zu können erweist sich oft als schwierig bis unmöglich. Vielmehr müssen die Algorithmen darauf aufbauen, welche Operationen in den Vorversionen durchgeführt

wurden und wie der Anwender Widersprüche zwischen den Operationen auflösen lassen möchte. Es genügt also nicht die reine Speicherung der Objektinhalte, sondern auch die durchgeführten Operationen bei den Transaktionen müssen nachvollziehbar sein.

7. Aushandlungssache

Das automatische Zusammenführen verschiedener Versionen eines Entwurfs ist ein technisch höchst anspruchsvolles Problem. Ansätze zur Lösung dieses Problems wurden im vorigen Abschnitt diskutiert. Nichtsdestotrotz muss man sich bewusst sein, dass Merging nicht allein ein technisches, sondern vielmehr ein konzeptionelles Problem ist. Dieses ist in der Natur der Entwurfsprobleme an sich begründet. Ein Charakteristikum von bösartigen Problemen (Wicked Problems), zu denen nach Rittel und Webber (1973) Entwurfsprobleme gehören, ist dass die Definition eines Problems die Art der Lösung bereits mitbestimmt. Diese Definition ist jedoch abhängig vom Wissen, vom Erkenntnisstand und von den Erfahrungen des Entwerfenden. Sie ist also in gewissem Sinne willkürlich! Entwurfsentscheidungen sind demzufolge unweigerlich von subjektivem Charakter. Einzig durch die Kommunikation und Interaktion mehrerer Beteiligter können solche Entscheidungen objektiviert werden. Objektiviert meint dabei jedoch nicht wahrheitsgetreuer, sondern kann aufgrund der angesprochenen Problematik lediglich einen Konsens unter den Beteiligten erreichen, der weniger subjektiv als eine Einzelentscheidung ist.

Kommen wir zurück zum Zusammenführen von Entwurfsvarianten, dann wird deutlich, dass dessen Automatisierung ganz grundlegende Fragestellungen im Weg stehen. Diese werden am einfachen Beispiel der Positionierung eines Fensters in einer Fassade offensichtlich: Schiebt einer der Beteiligten ein Fenster nach links, um die Ausgewogenheit in einer Fassade zu erhöhen, und ein anderer Beteiligter schiebt dasselbe Fenster etwas nach rechts, um die Lichtverteilung im Innenraum zu optimieren, dann ist es offensichtlich, dass die Lösung dieses Konflikts nicht darin bestehen kann, das Fenster in der Mitte der beiden Operationen zu positionieren. Ein Merging dieser beiden Entwurfsvarianten ist grundsätzlich nicht möglich.

Wenn in solchen Fällen Einigung erfolgen soll, dann kann das nur über einen Mensch zu Mensch Aushandlungsprozess geschehen. Das einzige was das Versionierungssystem dabei tun kann, ist darauf hinzuweisen, dass es Konflikte gibt, um den Aushandlungsbedarf zu signalisieren. Wie ein solches Benachrichtigungssytem gestaltet werden muss, ist in zukünftigen Forschungsarbeiten zu klären. Mögliche Anhaltspunkte hierfür liefert das im Bereich des CSCW bekannte Konzept der Boundary Objects (Star, 1989). In unserem Kontext wür-

de das digitale Modell solch ein Boundary Object darstellen, dass während des Entwurfsprozesses verformt, interpretiert, weiterentwickelt wird. Dieses Boundary Object besteht neben seiner Materialisierung durch sichtbare Artefakte (Modelle, Zeichnungen, Skizzen, Text), auch aus allen im Prozess entstandenen Varianten und Alternativen (Versionen). Jeder Zustand eines Modells besitzt also Vorgänger und Nachfolger. Es enthält also Informationen darüber, wie es entstanden ist, und wie es weiterentwickelt wurde. Diese Informationen können wichtige Erkenntnisse über das zu entwerfende Objekt liefern und den Aushandlungsprozess (Diskussionen, Gruppenbildungen, Interaktion, Ideenfindung) vorantreiben.

8. Zusammenfassung und Ausblick

Versionierung ist ein zentrales Thema im computergestützten kollaborativen Entwurfsprozess. Damit diese jedoch nicht nur schmückendes Beiwerk in einem CAAD-System ist, sondern auch aktiv beim Entwerfen genutzt werden kann, ist es wichtig, bestimmte Kriterien (Charakteristika des Entwerfens) bei der Versionierung zu berücksichtigen. Diese betreffen die Arbeit an einem gemeinsam genutzten Modell, die Koppelung verschiedener (auch fachfremder) Werkzeuge (Applikationen) mit dem Modell, die Möglichkeit einer asynchronen sowie einer synchronen Kollaboration, sowie eine nicht-lineare Aufzeichnung des gesamten Entwicklungsprozesses.

Das vorgestellte Versionierungskonzept zeigt eine auf technischer Ebene allgemeingültige Lösung zur Versionierung von digitalen Objekten (Entwurfsartefakten). Die Allgemeingültigkeit des Konzepts liegt in der Vielfalt der auftretenden Fälle begründet und bildet eine flexible Basis für weitere Arbeiten. Neben dem allgemeingültigen Versionierungskonzept auf unterster Datenebene wurden Probleme des Mergings vorgestellt, sowie deren Lösung durch einen hybriden Ansatz aus objekt- und operationsbasierter Versionierung diskutiert. Letztlich wurde auf Probleme des Mergings auf konzeptioneller Ebene hingewiesen, und erläutert, dass diese nur durch menschliche Aushandlungsprozesse gelöst werden können.

Zukünftige Entwicklungen beschäftigen sich mit der Unterstützung dieses Aushandlungsprozesses durch Koordinationsmechanismen, welche sich dem Versionierungskonzept bedienen. Dazu gehören Fragen zur Navigation: Wie kommt man von einer Version zur nächsten? Lassen sich Ähnlichkeiten in verschiedenen Alternativen feststellen? Wie können diese zur Strukturierung benutzt werden? Wichtig sind zudem Fragen zur Interaktion: Wie gestaltet sich der Informationsaustausch in diesem komplexen Versionsnetzwerk? Wie kann man Elemente von einer Version in eine andere transportieren? Schließlich sind Fragen zur Ko-

ordination der behandelten Prozesse zu beantworten: Lassen sich aus der Prozessverfolgung Entwurfsabsichten ableiten? Wie lassen sich verschiedene Versionen bewerten bzw. vergleichen?

Referenzen:

Arnheim, R. (1969). *Visual thinking*. Berkeley: University of California Press.

Benning, P., Dumoulin, C., Dehlin, S., Tulke, J., Åberg, P., Ryd, N., et al. (2010). *Collaboration Processes - A State of the Art*.

Eastman, C., Eastman, C., Teicholz, P., & Sacks, R. (2008). *BIM handbook: a guide to building information modeling for owners, managers, designers, engineers, and contractors*: Wiley.

IFC2x Edition 3 Technical Corrigendum 1. (2007). buildingSMARTInternational.

Koenig, R., Thurow T., Braunes J., Tonn C., Donath D., & Schneider S. (2010). *FREAC: A Technical Introduction to a Framework for Enhancing Research in Architectural Design and Communication.* Paper presented at the Future Cities - 28th eCAADe Conference, Zürich.

Latour, B., & Yaneva, A. (2008). "Give me a gun and I will make all buildings move": An ANT's view of Architecture' In R. Geiser (Ed.), *Explorations in Architecture: teaching, Design, Research* (pp. 80 - 89). Basel: Birkhäuser.

Lawson, B. (2005). Oracles, draughtsman and agents: the nature of knowledge and creativity in design and the role of IT. *Automation in Construction, 14*(3), 383-391.

Maher, M., Bilda, Z., & GÜL, L. (2006). Impact of Collaborative Virtual Environments on Design Behaviour. In J. S. Gero (Ed.), *Design Computing and Cognition '06* (pp. 305-321-321): Springer Netherlands.

Nour, M., Férino, M., Dehlin, S., Tulke, J., Pfitzner, M., Houttu, J.-M., et al. (2010). *Overview of Information Management Applications, Including Object-Based Version Management*.

Rittel, H., & Webber, M. (1973). Dilemmas in a general theory of planning. *Policy Sciences, 4*(2), 155-169.

Schmidt, K., & Wagner, I. (2004). Ordering Systems: Coordinative Practices and Artifacts in Architectural Design and Planning. *Comput. Supported Coop. Work, 13*(5-6), 349-408.

Schneider, S., Richter N., Petzold F., Koenig R. (2010). *A Matter of Negotiation: Managing Uncertainties in an Open Design Process.* Paper presented at the Design Research Society - International Conference on Design & Complexity, Montreal.

Schön, D. (1983). *The reflective practitioner: how professionals think in action*: Basic Books.

Sebastian, R. (2007). *Managing Collaborative Design*: Eburon Academic Publishers.

Star, S. L. (1989). The structure of ill-structured solutions: boundary objects and heterogeneous distributed problem solving *Distributed Artificial Intelligence (Vol. 2)* (pp. 37-54): Morgan Kaufmann Publishers Inc.

Tellioglu, H. (2009). *Keeping artifacts alive: towards a knowledge management system*. Paper presented at the Proceedings of the International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing.