MODELLIERUNG UND PRÜFUNG VON CONSTRAINTS IN DYNAMISCHEN DIGITALEN BAUWERKSMODELLEN

André Borrmann
andre.borrmann@informatik.uni-weimar.de
Professur Informations- und Wissensverarbeitung
Fakultät Bauingenieurwesen
Bauhaus-Universität Weimar
Coudray-Straße 13b, 99421 Weimar

Betreuer der Arbeit: Dipl.-Ing. Thomas Hauschild, Prof. Dr.-Ing. habil. Reinhard Hübler
Art der Arbeit: Diplomarbeit im Rahmen des SFB 524
Fachbereich GI: Informationstechnik und technische Nutzung der Informatik

Einleitung

Die Bauplanung ist eine komplexe, verantwortungsvolle Aufgabe. Ihre Lösung ist durch eine hohe Zahl von Beteiligten aus unterschiedlichen Fachdisziplinen geprägt. Die große Menge gegenseitiger Abhängigkeiten zwischen den einzelnen Planungsleistungen zwingt die Beteiligten zu einer intensiven Kooperation, für deren adäquate Unterstützung die Gewährleistung eines reibungslosen Austauschs von planungsrelevanten Daten notwendig ist. Der von Hübler et al. verfolgte Ansatz zur Integration der bereits gut durch den Einsatz von Rechnern unterstützten Teilprozesse sieht die Verwaltung aller anfallenden Planungsdaten eines Vorhabens in einem gemeinsamen, verteilten Informationssystem vor. [1]

Ein geeignetes Informationssystem muss ein hohes Maß an Parallelität von Planungsaktivitäten gewährleisten und gleichzeitig die Konsistenz der Planungsdaten garantieren können. Während die Parallelisierung zu einer potentiellen Verkürzung der Planungszeiträume führt, können durch effektive Konsistenzsicherungsmaßnahmen auf semantisch hohem Niveau Planungsfehler und damit verbundene Folgekosten vermieden werden.

Für die rechnerinterne Abbildung des Planungsgegenstandes hat sich ein Vorgehen nach dem objektorientierten Paradigma durchgesetzt, dessen Ergebnis ein objektorientiertes Bauwerksmodell ist. Die Ausprägungen der Klassen eines solchen Modells beinhalten die Daten eines konkret existierenden bzw. zu planenden Bauwerks. Um den verschiedenen Sichten der beteiligten Fachplaner gerecht zu werden und gleichzeitig die Komplexität des Datenmodells zu begrenzen, hat sich eine Partitionierung des Bauwerksmodells als sinnvoll erwiesen. Die Partialmodelle repräsentieren die Taxonomie einer einzelnen Domäne und werden daher auch Domänenmodelle genannt.

Zwei wesentliche Aspekte des Erzeugens und der Verwendung von Planungsdaten machen eine Dynamisierung der verwendeten Domänenmodelle notwendig: Zum einen impliziert der Unikatcharakter der zu errichtenden oder zu revitalisierenden Bauwerke, dass ein für alle Bauvorhaben gültiges Produktmodell nicht zu identifizieren ist. Zum anderen resultieren aus dem langen Lebenszyklus des Bauwerks nicht vorhersehbare Anforderungen an die Struktur der zu verwaltenden Daten: Das System muss es daher erlauben, dass zur Laufzeit bereits vorhandenen Klassen des Domänenmodells Attribute hinzugefügt und Subklassen von ihnen gebildet werden können.

Zur Verwaltung dynamischer Bauwerksmodelle und deren Ausprägungen kommen dynamische Modellverwaltungssysteme (MVS) zum Einsatz. Das im Rahmen des SFB 524 prototypisch implementierte MVS basiert auf einer aktuellen Version der AKO-Schnittstelle, mit deren Hilfe

zum einen objektorientierte Domänenmodelle erstellt und zur Laufzeit manipuliert werden können und zum anderen Zugriff auf Instanzen eines Domänenmodells möglich ist. [2]

Das MVS agiert innerhalb der verteilten Umgebung eines Fachplanerbereiches als Server, die dort eingesetzten Fachapplikationen als dessen Clients. [3]

Constraints in dynamischen digitalen Bauwerksmodellen

Constraints im Kontext der Bauwerksmodellierung sind Zwangs- bzw. Nebenbedingungen, die für Instanzen der Klassen eines Domänenmodells gelten. Bei der Entwicklung von herkömmlichen Expertensystemen wird das Fachwissen durch Wissensingenieure erfasst und in die Wissensbasis eingespeist. Bei dem hier propagierten Informationssystems soll dagegen der Anwender seine Wissensbasis ohne das Mitwirken von Wissensingenieuren selbst erweitern können. Die Constraints werden vom Fachplaner mit Hilfe der Bezeichner des Domänenmodells beschrieben und können dem Domänenmodell dynamisch, d.h. zur Laufzeit des Informationssystems, hinzugefügt werden. Die Dynamik ist eine Notwendigkeit, die letztlich auch aus der Modifizierbarkeit des Domänenmodells resultiert.

Constraints bereichern das Digitale Bauwerksmodell um zusätzliche Semantik und können auf diese Weise dazu beitragen, dass das Informationssystem den hohen Anforderungen der kooperativen Bauplanung gerecht wird. Sie können dazu eingesetzt werden, einen Teil des Domänenwissens rechnerintern abzubilden. Bei derartigen Constraints handelt es sich um Regeln, die sich aus rechtlichen Normen ableiten oder den Stand der Kunst dieser Fachdisziplin wiederspiegeln und über viele Projekte hinweg Gültigkeit besitzen.

Daneben können Constraints auch zur Formulierung von Entwurfsbedingungen verwendet werden, die nur in einem konkreten Projekt gültig sind. Sie können beispielsweise die speziellen Wünsche des Bauherren oder die spezifischen fachlichen Anforderungen in einem Projekt formalisieren. Solche Constraints sind einer bedeutend höheren Dynamik ausgesetzt als die, die dem vergleichsweise statischen Domänenwissen zuzuordnen sind. Durch Auswertung projektspezifischer Constraints können frühzeitig potentielle Entwurfskonflikte erkannt werden. Außerdem kann mit ihrer Hilfe der Soll-Zustand der Planung beschrieben und durch ihre Auswertung mit dem aktuellen Ist-Zustand verglichen werden.

Im einfachsten Fall grenzt ein Constraint den für ein Attribut gültigen Wertebereich ein. Constraints können aber auch zur Abbildung von Abhängigkeiten zwischen Attributen dienen, was das Auftreten redundanter Attribute im Domänenmodell erlaubt. Sie können auf diese Weise dazu beitragen, die Aussagekraft des Domänenmodells zu erhöhen und seine Verwendbarkeit für die unterschiedlichen Applikationen einer Domäne sicherzustellen. Constraints in diesem Sinne können als Mittel zur Formulierung von Konsistenzbedingungen für Planungsdaten angesehen werden. Die Constraint-basierte Modellierung von Abhängigkeiten zwischen Attributen stellt darüber hinaus eine geeignet Grundlage für die Implementierung generischer Algorithmen zur Versionierung von Planungsdaten dar.

Um die notwendige Flexibilität bei der Planung zu gewährleisten, müssen temporäre Unstimmigkeiten zwischen den Planungsdaten und den formulierten Bedingungen vom Informationssystem toleriert werden. Wenn diese jedoch nicht erkannt, überwacht und zu gegebenen Zeitpunkt beseitigt werden, kann das zu schweren Planungsfehlern mit entsprechenden Folgekosten führen. Das kann durch die Formulierung entsprechender Nebenbedingungen und anschließender Prüfung der Instanzdaten auf Einhaltung der für sie gültigen Constraints vermieden werden. Dabei sollten der Zeitpunkt und der Umfang der Prüfungen vom Planer gewählt werden können.

Umsetzung eines Constraint-Moduls für Modellverwaltungssysteme

Für die Formulierung von Constraint-Ausdrücken wurde die Constraint-Sprache CML (Constraint Modeling Language) entwickelt. CML ist ein Dialekt der zum Industriestandard UML gehörenden Object Constraint Language (OCL), die für die formale Spezifikation von Softwaresystemen vorgesehen ist. Er ist auf die spezifischen Bedürfnisse für die Definition von Konsistenz- bzw. Entwurfsbedingungen in dynamischen Bauwerksmodellen zugeschnitten. [4]

Die Syntax von CML-Ausdrücken ist mit der von OCL-Ausdrücken identisch, CML besitzt jedoch einen gegenüber OCL leicht eingeschränkten Sprachumfang. Da beispielsweise deskriptive Bauwerksmodelle nicht über die Fähigkeit zur Abbildungen von Objektverhalten (Operationen bzw. Methoden) verfügen, können in CML solche Sprachkonstrukte der OCL entfallen, die der Formulierung von Vor- bzw. Nachbedingungen dienen. Alle CML-Constraints sind Invarianten im Sinne der OCL-Spezifikation.

Der Sprachumfang von CML bietet Möglichkeiten zum Zugriff auf facettierte Attribute, zur Navigation entlang von Assoziationsbeziehungen, für Verknüpfungen durch arithmetische und logische Operatoren, für Vergleichoperationen sowie konditionale Konstrukte. Der Zugriff auf eine Relation, über die mehr als eine Instanz assoziiert werden kann, resultiert in einer Kollektion von Objekten. Für Kollektionen gibt es eine Reihe von vordefinierten Operationen, die über die einzelnen Elemente iterieren. Beispielsweise testet die Operation *forAll()*, ob eine anzugebende Bedingung von allen Elementen eingehalten wird.

Das AKO-Metamodell wurde um die Metaklasse *Constraint* erweitert. Ein Constraint-Objekt kapselt einen CML-Ausdruck und besitzt einen eindeutigen Namen. Es kann entweder mit einer einzelnen Instanz oder mit einer Klasse assoziiert werden, wobei das Constraint im zweiten Fall für alle Instanzen dieser Klasse gilt. Constraints, die Abhängigkeiten zwischen Attributen oder einen Teil des Domänenwissens abbilden, werden vorwiegend mit Klassen assoziiert. Constraints, die projektspezifisches Entwurfswissen repräsentieren, werden hingegen in der Regel mit einzelnen Instanzen assoziiert.

Constraint-Objekte werden vom MVS zusammen mit dem Domänenmodell und dessen Instanzen verwaltet. Der für die laufzeitdynamische Auswertung der CML-Ausdrücke zuständige OCL-Interpreter arbeitet ebenfalls im Adressraum des MVS-Servers. Dadurch kann die für die Prüfung großer Instanzmengen notwendige hohe Performanz erzielt werden.

Beim Zuweisen eines CML-Ausdrucks an ein Constraint-Objekt wird dieser syntaktisch und semantisch geprüft. Dabei wird die Existenz der im Ausdruck verwendeten Bezeichner für Attribute bzw. Relationen überprüft und festgestellt, ob die Typen der Operanden mit der jeweiligen Operation verträglich sind. Ist diese Überprüfung erfolgreich, wird der beim Parsen erzeugte Syntaxbaum zusammen mit dem Constraint-Objekt gespeichert. Dadurch ist beim Interpretieren kein erneutes Parsing notwendig, was wesentlich zur hohen Performanz des Constraint-Moduls beiträgt.

Die Auswertung eines Constraint-Ausdrucks für eine konkrete Instanz findet mit Hilfe des CML-Interpreters statt. Er ersetzt im CML-Ausdruck die Bezeichner von Attributen durch die konkrete Wertbelegung der Instanz und die Navigationsausdrücke durch Kollektionen von assoziierten Instanzen. Auf den daraus resultierenden Basistypen kann der Interpreter die im CML-Ausdruck benutzten Operationen ausführen. Ergebnis einer solchen Auswertung ist immer ein boolescher Wert, der besagt, ob die betreffende Instanz den im Constraint-Ausdruck formulierten Bedingungen genügt.

Die Prüfung der Instanzen auf Einhaltung der für sie gültigen Constraints findet zu einem vom Nutzer festzulegenden Zeitpunkt statt und kann von einem MVS-Client aus angestoßen werden. Sie kann für einzelne Instanzen oder entlang von Aggregationsrelationen für ganze Instanzpopulationen stattfinden. Die Constraint-Prüfung besitzt besondere Signifikanz beim Zusammen-

führen von Teildatenbeständen nach deren getrennter Offline-Bearbeitung und beim Erzeugen eines Revisionsstandes.

Beim Feststellen der Verletzung einer Entwurfs- oder Konsistenzverletzung macht das System detaillierte Angaben über die nicht eingehaltenen Constraints und der davon betroffenen Instanzen. Auf dieser Basis kann die Ursache der Inkonsistenz lokalisiert und manuell behoben werden.

Ausblick

Die Constraint-basierte Abbildung von Domänen- und Projektwissen ist eine gute Grundlage für die angestrebte Assistenz der Entwurfsumgebung beim Lösen von Entwurfskonflikten. Gerade angesichts der in der Bauplanung auftretenden Vielzahl komplex vernetzter Abhängigkeiten zwischen den Entwurfsobjekten ist eine Unterstützung des Planers bei der Suche nach einer optimalen Lösung wünschenswert. Für die Umsetzung einer solchen Funktionalität erscheint der Einsatz von Techniken des Constraint-basierten Lösens (engl. Constraint Based Reasoning) sinnvoll und muss eingehend untersucht werden.

Literatur

[1] Hübler, R.; Hauschild, Th.; Willenbacher, H.: Distributed Cooperative Building Models for Revivification of Buildings, International Association for Bridge and Structural Engineering (IABSE) Symposium 2002, Melbourne 2002

[2] Kolbe, P.; Ranglack, D.; Steinmann, F.: Eine Schnittstelle für dynamische Objektstrukturen für Entwurfsanwendungen. Internationale Konferenz für Anwendungen der Informatik und Mathematik in den Ingenieurwissenschaften IKM '97: Berichte, Bauhaus–Universität Weimar, 1997

[3] Hauschild, Th.; Hübler, R.: Distributed, Collaborative Management of Building Models for Revivication Projects. IX. International Conference on Computing in Civil and Building Engineering (ICCCBE), Taipeh, 2002

[4] The Object Management Group: The Unified Modeling Language Specification, Version 1.5. Chapter 6: The Object Constraint Language Specification. OMG Document 03-03-13, Framingham MA, 2003