Bauhaus-Universität Weimar
Lehrstuhl für Informatik im Bauwesen
Prof. Dr. K. Beucke

Integrationsmodelle

Das applikationszentrale und das objektzentrale Modell

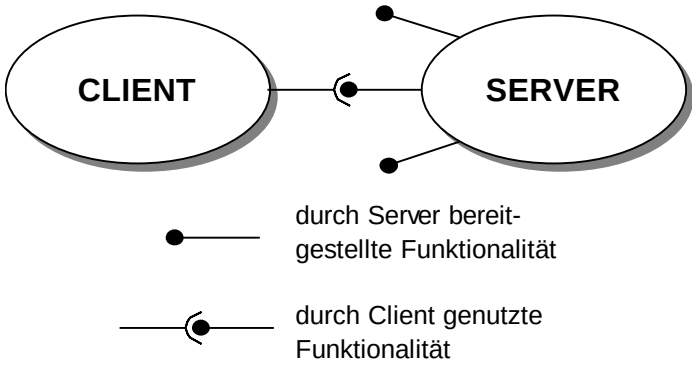
Lutz Röder, Karsten Koch Weimar, Oktober 1997

1. DAS APPLIKATIONSZENTRALE MODELL

Fast alle heute erhältlichen Applikationen besitzen Schnittstellen zum Austausch von Daten mit anderen Applikationen. Unter dem Begriff "Schnittstelle" versteht man die Standardisierung der Typen und der Struktur von Daten. Beispielsweise sind im Dateiformat DXF die Struktur (Header, Classes, Tables, Blocks, Entities) und die Datentypen (Line, Polyline, Ellipse) einer CAD-Zeichnung genau festgelegt. Durch konsequente Umsetzung einer Schnittstelle beim Exportieren und Importieren können Daten zwischen Applikationen ausgetauscht werden.

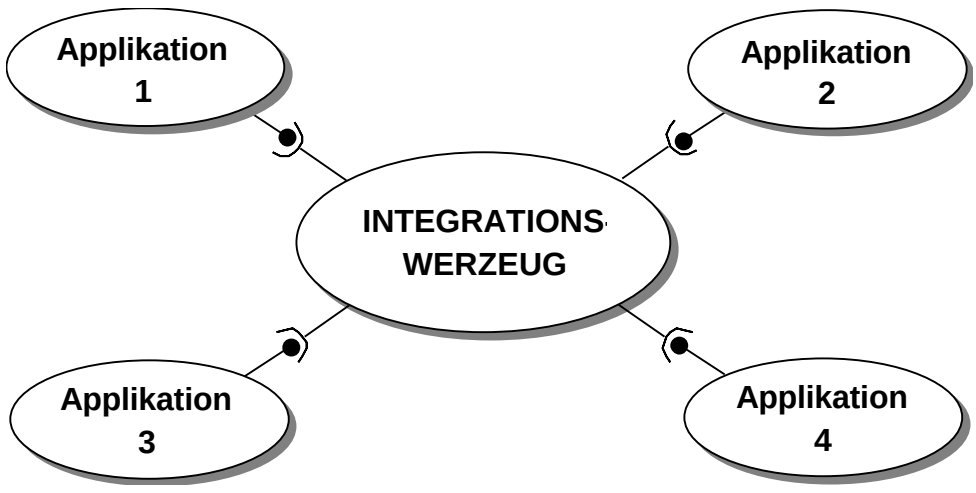
Die COM Technologie ermöglicht es, über Schnittstellen (Interfaces) auch Funktionalitäten quasi "auszutauschen". Applikationen können nicht nur Daten an andere Applikationen übergeben, sondern auch deren weitere Verarbeitung beeinflussen.

Abbildung 1



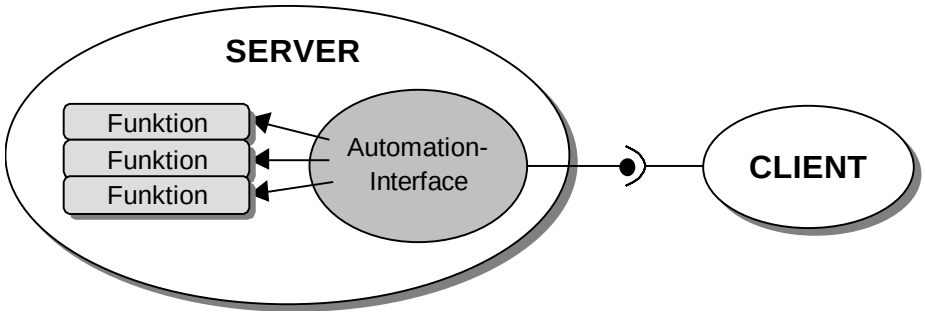
Um nun die Kommunikation zwischen allen beteiligten Applikationen zu gewährleisten, müssen alle eine Schnittstelle zur Automatisierung unterstützen. Die auf COM basierende Automation Technologie ermöglicht die Bereitstellung von Funktionalität über ein standardisiertes Interface. Eine Applikation, die mithilfe einer solchen Technologie Funktionalitäten über ein bestimmtes Interfaces bereitstellt, wird im folgenden Server genannt. Applikationen, die diese Funktionalitäten nutzen, heißen Client.

Abbildung 2



Beim applikationszentralen Modell wird einer Applikation eine zentrale Rolle zuteil: Sie dient dem Anwender als Integrationshilfe. Diese Applikation wird deshalb als Integrationswerkzeug bezeichnet. Sie verfügt über einen Mechanismus, über den sie auf die in einem Server implementierten Funktionalitäten zugreifen kann. Das Integrationswerkzeug kommt als Client zum Einsatz und die Applikationen als Server. Die Unterstützung des Automation-Interfaces wird vom Integrationswerkzeug nicht gefordert.

Abbildung 3



Die Aufgabe der Integration fällt im applikationszentralen Modell vollständig dem Anwender - im weiteren Integrator genannt - zu. Mit seinem Wissen werden die Beziehungen und Abhängigkeiten zwischen den Daten modelliert.

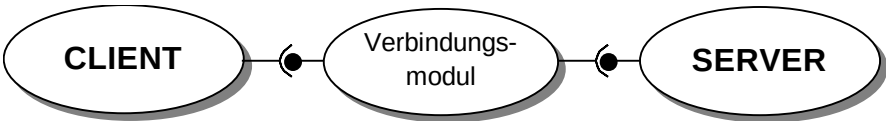
Abbildung 4

Mit dem Integrationswerkzeug steht ihm dazu ein Werkzeug zur Verfügung, das folgenden Ansprüchen genügt:

1. Das Integrationswerkzeug verfügt über Mechanismen, mit denen der Integrator in der Lage ist, Integrationsvorgänge zu automatisieren.
2. Es müssen Datentypen für applikationsübergreifende Daten vorhanden bzw. definierbar sein. Diese Daten müssen serialisierbar sein.
3. Der Integrator ist in der Lage, die auf die integrierten Server-Applikationen verteilten Daten vollständig zu serialisieren.

Werden Server, die gleiche Funktionalitäten bereitstellen ausgetauscht, so können Verbindungsmodule die bereits vorhandenen Intergrationsmechanismen an die Funktionen des neuen Servers anpassen. Diese Module wirken in die Richtung der Server-Applikation als Client, während sie selbst die alte Applikation ersetzen und deren Funktionen für das Integrationswerkzeug als Server bereitstellen.

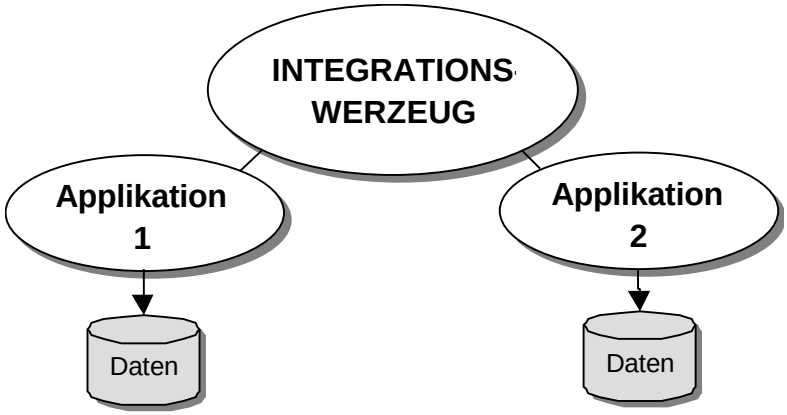
Abbildung 5



Um die Funktionalitäten einer Server-Applikation nutzen zu können, muß der Integrator im applikationszentralen Modell um Aufgabe und Syntax der Funktionen wissen. Das Vorhandensein der vollständigen Dokumentation der über das Automation-Interface bereitgestellten Funktionen ist somit eine weitere Anforderung an die Server-Applikation.

Die Datenhaltung erfolgt im applikationszentralen Modell verteilt. Jede Server-Applikation enthält nur Daten, die für die interne Datenverarbeitung relevant sind. Diese können entweder separat oder über das Integrationswerkzeug serialisiert werden. Für letzteres muß die Server-Applikation Serialisierungsfunktionen über das Automation-Interface bereitstellen.

Abbildung 6



Ein Vorteil des applikationszentralen Modells besteht in der semantikfreien Integration. Dadurch können Integrationsvorgänge sehr flexibel gestaltet werden. Erst durch das Wissen des Integrators können Applikationen und Daten verbunden werden. Struktur und Typ der Daten werden zur Laufzeit erst festgelegt. Damit schafft der Integrator selbst erst die Schnittstellen zum Datenaustausch im Rahmen des relevanten Projektes (Modells). Diese Schnittstellen bilden Zusammenhänge zwischen Daten und Funktionen der einzelnen Server-Applikationen ab. Durch die offene und flexible Funktionsweise der Intergrationsvorgänge können Änderungen der Schnittstellen in Abhängigkeit von Änderungen der Strukturen und Datentypen des Projektes durch den Integrator zur Laufzeit realisiert werden.

Eine weitere Möglichkeit zur Optimierung der Integrationsvorgänge besteht im vorlageorientierten Arbeiten. Bereits erarbeitete Schnittstellen und Integrationsvorgänge können hier unabhängig von Daten gesichert werden. Ähnliche Integrationsprobleme können später durch das Zurückgreifen auf diese bereits vorhandenen Mechanismen erheblich schneller gelöst werden. Da das Integrationswerkzeug eine beliebige Applikation sein kann, die die bereits genannten Anforderungen erfüllt, kann der Integrator auch mehrere Applikationen gleichzeitig als Integrationswerkzeug nutzen. Über ein Netzwerk können auch mehrere Integratoren verteilt zusammenarbeiten.

Es stellt sich natürlich die Frage nach der Realisierbarkeit einer solchen Integration. Sind die Funktionen der einzelnen Server-Applikationen ungenügend dokumentiert, zu umfangreich oder nicht ausreichend vorhanden, kann der für die Integration nötige Aufwand ein respektables Maß schnell überschreiten. Des weiteren sind Syntax und Aufgabe der bereitgestellten Funktionen abhängig vom Aufbau, der Datenhaltung und den verwendeten Algorithmen innerhalb der Server-Applikation. Somit liegt eine große Verantwortung beim Hersteller der jeweiligen Applikation. Ein weiteres Problem bereitet die Modell- oder Projektgröße. Bis zu welchem Umfang ist ein applikationszentrales Modell realisierbar? Wie kann ein Zusammenarbeiten von mehreren Intergratoren gewährleistet werden, wenn der Kommunikationsaufwand in akzeptablen Grenzen gehalten werden soll?

Diese Fragen gilt es noch zu untersuchen. Zusammenfassend läßt sich feststellen, daß das applikationszentrale Modell eine Möglichkeit zur flexiblen und semantikfreien Integration von Daten und den zu deren Verarbeitung nötigen Applikationen darstellt.

2. Das objektzentrale Modell

Objektorientierte Modellierung beinhaltet die Zusammenfassung von Daten, deren Struktur und Typen zu Objekten. Dabei bieten Objekte auch Funktionen an, mit denen die zugehörigen Daten - im folgenden auch Attribute genannt - objektspeziefisch verändert werden können. Diesen Modellierungsansatz greift das objektzentrale Modell auf.

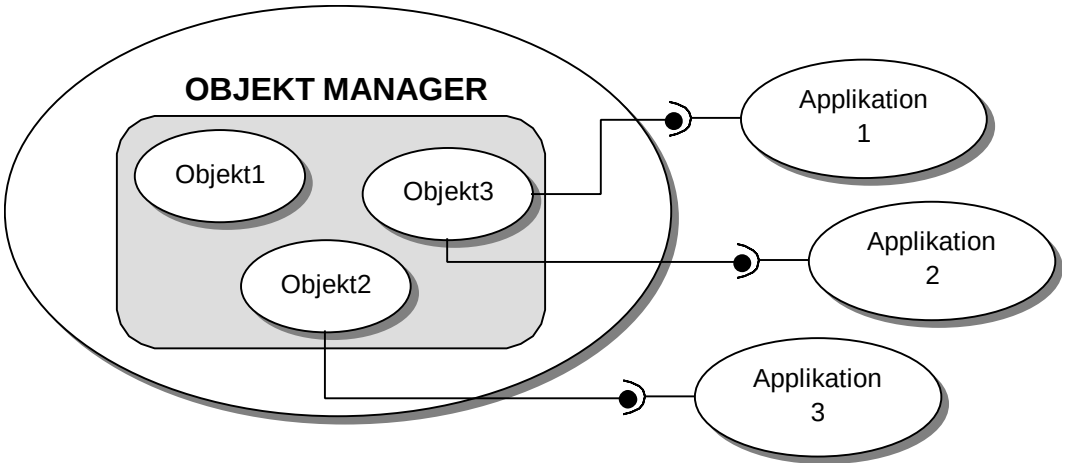
Objekte, die in diesem Modell zum Einsatz kommen, beinhalten

1. die jeweiligen Attribute, die zur Beschreibung des Objektes nötig sind,
2. Funktionen zur Bearbeitung und zum Austausch der objektspeziefischen Attribute, die durch einige standardisierte Interfaces zur Verfügung stehen,
3. standardisierte, vom Objekttyp größtenteils unabhängige Funktionen zur Visualisierung, Bearbeitung, Einbindung etc., die in verschiedenen ebenfalls standardisierten Interfaces zur Verfügung stehen.

Die Vorraussetzung für solche Interfaces ist COM oder eine gleichwertige Technologie. In diesem Zusammenhang sollen auch OLE und Active X erwähnt werden, die einen Standard für die unter 3. genannten Funktionen darstellen.

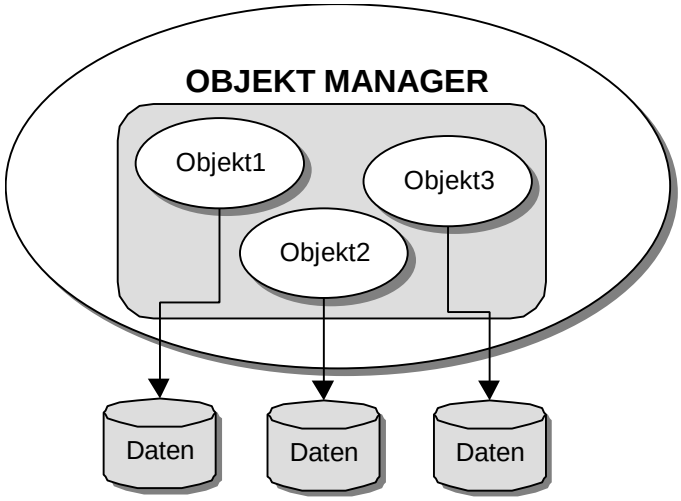
Ein entscheidendes Merkmal des objektzentralen Modells ist die Entkoppelung der Objekte von den Applikationen. Die Server-Applikationen bieten ihre spezialisierten Funktionen bzw. Operationen an, benötigen dazu jedoch die Objekte, auf die die Funktionen angewendet werden sollen. Die Objekte wiederum müssen diese Aktionen unterstützen. Um die Zusammenarbeit zwischen den Server-Applikationen und den Objekten zu gewährleisten, kommt eine spezielle Applikation zum Einsatz, die Objekt Manager genannt wird.

Abbildung 7



Der Objekt Manager bildet einen applikativen Rahmen um die Objekte. Der Anwender ruft die jeweiligen Server-Applikationen über die Objekte auf. Die im Objekt Manager festgelegten Strukturen sind variabel und nicht applikationsübergreifend. Sie dienen allein der Übersicht über das Modell. Die Datenhaltung erfolgt in einer Datenbank, in der die Objekte ohne eine Ordnung abgelegt sind.

Abbildung 8



Die einzelnen Ordnungen oder Hierarchien sind applikationsspeziefische Daten und werden getrennt von jeder Applikation erstellt. Diese Forderung resultiert aus der Unvereinbarkeit aller nötigen Sichten in einer Modellhierarchie.

Es müssen also folgende Aufgaben mit dem Objekt Manager realisierbar sein:

1. Verwaltung der Objekte
2. Verknüpfung der Applikationen mit den Objekten
3. Verwaltung der in den Applikationen festgelegten Relationen zwischen den Objekten

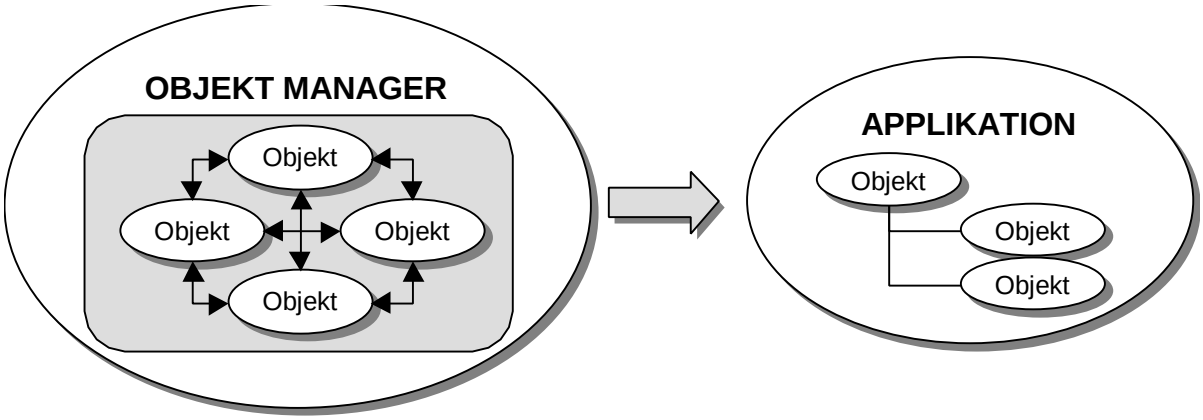
Dabei ist die Frage der Identifizierbarkeit von Objekten von entscheidender Bedeutung, da verschiedene Objekte vom gleichen Objekttyp verwaltet werden

müssen. Es wird also gefordert, daß jedes Objekt eindeutig identifizierbar sein muß. Die Identifizierung der Objekte im Objektmanager soll unabhängig von der verwendeten Datenbankart sein. Jedem Objekt wird ein eindeutiger Identifikator zugewiesen. Die Eindeutigkeit wird sowohl räumlich (nach Ort der Erzeugung), als auch zeitlich (nach Zeitpunkt der Erzeugung) verlangt. Dabei sind folgende Möglichkeiten näher zu untersuchen:

- Die Erzeugung der Identifikatioren erfolgt durch einen Algorithmus, der in implementierter Form vorliegt. Dieser verwendet die Gerätekennung, die Systemuhrzeit und einen zufälligen Sicherheitsfaktor für die Generierung der Identifikatoren.
- Der Anwender erzeugt den Identifikator für jedes Objekt selbst. Der Objektmanager verfügt über einen Mechanismus, mit dem die Eindeutigkeit eines jeden Identifikators überprüft werden kann. Die Änderung der Identifikatoren ist möglich.

Objektordnungen können von den Applikationen aus den Objekteigenschaften hergeleitet werden. Damit wird dem Anwender in jeder Umgebung ein übersichtliches Arbeiten möglich. Des weiteren ermöglicht eine solche Funktionalität die Suche nach bestimmten Objekten, die Erstellung von Listen bzw. das Analysieren des Modells und die Auswahl von Objekten nach verschiedenen Kriterien. Algorithmen, die zur Erstellung von Objektordnungen verwendet werden, müssen auf alle Attribute der Objekte über Interfaces zugreifen können.

Abbildung 9

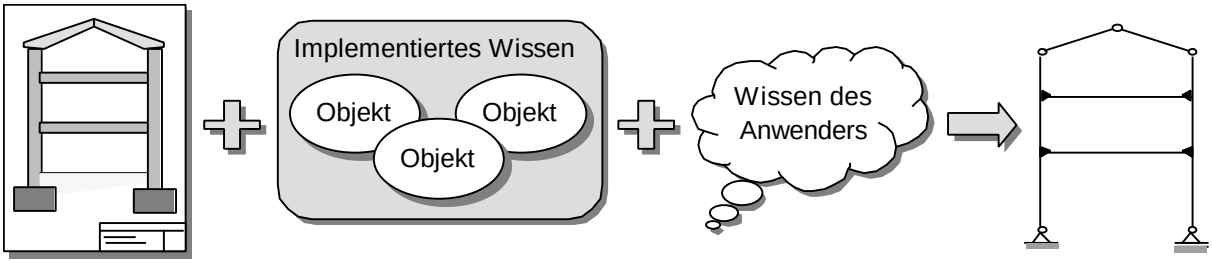


Ein Problem stellt der Vergleich der Attribute mehrerer Objekte unterschiedlichen Objekttyps dar, wenn davon ausgegangen wird, daß diese nicht in standardisierter Gestalt vorliegen. Da dies aber für allgemeine Algorithmen zur Erstellung von Objektordnungen notwendig wäre, muß davon ausgegangen werden, daß auf Objekttypen spezialisierte Algorithmen verschachtelt verwendet zur Verwendung kommen. Die Gestaltung der Algorithmen ist abhängig von der Art und den Gesetzen innerhalb der angestrebten Odnung. Zu untersuchen sind Eigenschaften wie Asymetrie, Linkseindeutigkeit und Transivität.

Das objektzentrale Modell stellt an die Applikationen weit komplexere Anforderungen als das applikationszentrale Modell. Durch die weitestgehend objektorientierte Realisierung besitzen die Gesetze und Mechanismen, die das Zusammenwirken von Applikationen und Objekten beschreiben eine sehr große Bedeutung. Die Entwicklung von Applikationen für das objektzentrale Modell wird nur möglich sein,

wenn diese Gesetze bzw. Mechanismen geklärt sind und die benötigten Objekte existieren. Der Schwerpunkt weiterführender Untersuchungen sollte sich auf die Gestaltung der Objekte konzentrieren.

Abbildung 10



3. VERGLEICH DER MODELLE

	applikationszentral	**objektzentral**
semantikfreie Intergration	möglich	nicht möglich
Intergration	nur im Rahmen von Vorlagen automatisierbar	geschieht einmalig
Integrations-aufgabe liegt bei	Anwender (Integrator)	Entwickler
Austauschbarkeit von Applikationen	Nur durch Verbindungsmodul	ja
Komplexität der Interfaces	niedrig	hoch
Standardisierung der Interfaces	begrenzt nötig	allgemein und für jedes Applikationsfachgebiet separat nötig
Objektordnung	durch Integration gegeben, statisch	kann nach bestimmten Kriterien generiert werden, flexibel
Anforderungen an den Anwender	hoch, Integrator = spezialisierter Anwender	niedrig
Modellbildung	Nur mit hohem Aufwand möglich, unflexibel	mit geringem Aufwand möglich, flexibel
Serialisierung	in jeder Applikation separat, Automatisierung möglich	getrennt nach objektspezifischen und applikationsspezifischen Daten

4. AUSBLICK

Das applikationszentrale und das objektzentrale Modell sind noch unzureichend formuliert. Für einen aussagekräftigen Vergleich müssen bei beiden Modellen die Bereiche Interfaces, Identifikation und Modellordnung noch genauer untersucht werden. Dabei sollte Wert auf die Einbeziehung bereits vorhandener Mechanismen gelegt werden, um die Realisierbarkeit der Modelle zu erleichtern.

Bei der Untersuchung der Standardisierungsmöglichkeiten können beispielsweise bereits existierende Interfacedefinitionen im Hinblick auf ihre Verwendbarkeit untersucht werden.

Für die Identifikation ist zu klären, durch wen Identifikatoren vergeben werden sollen und welche Faktoren bei der Generierung einbezogen werden.

Die Erzeugung von Modell- und Objektordnungen erfordert eine genaue Untersuchung der Eigenschaften von Objektbeziehungen. Es ist zu klären, wie diese eindeutig gefunden und abgebildet werden können.