

ClassiX[®]

ClassiX[®]

Das Konzept

Stand September 2007

Inhalt

1	Concept ClassiX®	3
1.1	Konzeption des Basis-Systems.....	4
	Baukastenprinzip	4
	Mehrschichtarchitektur	4
	Dynamisches Interface zu Geschäftsobjekten.....	5
	Maximale Unterstützung des Customizing.....	6
	Grundmechanismen betriebswirtschaftlicher Anwendungen.....	7
	Hohe Abstraktion der Datenspeicherung	8
	Reduzierte Netzbelastung.....	9
	Internationalisierung	9
	InstantView®-Design-Tool.....	9
	Ausblick	9
1.2	Konzeption des Unternehmensmodells CyberEnterprise®.....	10
	Prinzipien der Modellierung der Geschäftsobjekte.....	10
	Geschäftsobjekte als funktionale Bausteine	10
	Reale Objekte und Begriffe: die eigentlichen Geschäftsobjekte.....	11
	Dynamisierung des Modells durch Transaktionen	12
	Der Transaktionsmanager.....	12
2	Spezielle Objekte des CyberEnterprise®.....	14
2.1	Attribute.....	14
2.2	Strukturen.....	14
2.3	Allokationen.....	15
2.4	COM-Objekte.....	16
2.5	Drucken.....	17
2.6	Hilfe-System.....	18
2.7	Multimedia-Objekte, Spracheingabe.....	18
3	Index.....	19

1 Concept ClassiX®

Seit es EDV-Anwendungen gibt, werden die naheliegenden Ziele

maßgeschneiderte Anwendungslösungen

schnelle Anwendungsentwicklung

einfache Softwarewartung

verfolgt.

ClassiX® hat sich dieser Herausforderung neu gestellt.

Nach über 25 Jahren Erfahrung des EDV-Einsatzes im betrieblichen Umfeld ist ausreichend Erfahrung über die benötigte Funktionalität gesammelt worden. Es scheint an der Zeit, die traditionellen Wege der Anwendungsentwicklung neu zu überdenken. Das Concept ClassiX® konzentriert sich auf den Bereich betriebswirtschaftlicher Anwendungen, um mit innovativen Konzepten und dem Einsatz modernster Softwaretechnologie einen neuen Lösungsansatz zu bieten.

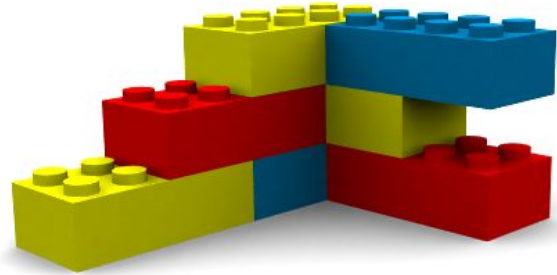
Die am Markt üblichen Software-Werkzeuge versuchen den gesamten Prozess allgemeiner Programmierung, vom Design bis hin zur Implementation, zu unterstützen. Der Ansatz von ClassiX® hingegen beginnt bereits mit vorgefertigten Komponenten. Projekte müssen nicht immer wieder von Null beginnen.

Wir betrachten zunächst die Konzeption des ClassiX®-Systems, gehen danach auf die Geschäftsobjekte des CyberEnterprise® ein und beschreiben dann die Eigenschaften und Leistungen des Systems.

1.1 Konzeption des Basis-Systems

Baukastenprinzip

Es gibt wiederverwendbare Komponenten¹ - auf verschiedenen Ebenen der Komplexität - mit nahezu unbegrenzter Kombinationsmöglichkeit.



ClassiX® ist durchgehend objektorientiert.

Reale und begriffliche (abrechnungstechnische) Gegebenheiten eines Unternehmens sind als (Geschäfts-)Objekte modelliert. Dies sind die elementare Bausteine, aus denen zusammen mit Komponenten für das User-Interface wiederverwendbare Teilstücke einer Anwendung aufgebaut werden (Modul). Eine Anwendung entsteht durch die Kombination dieser Module.

ClassiX® kann als objektorientiertes Framework verstanden werden, die Vielzahl bereits entwickelter und eingesetzter Module macht ClassiX® aber auch zu einer fertigen Anwendungslösung.

Der stufenweise Aufbau entspricht einer Softwarearchitektur mit mehreren, deutlich voneinander getrennten Ebenen:

Mehrschichtarchitektur

Die Abbildung zeigt die Schichten und führt die entsprechenden Bezeichnungen der ClassiX®-Terminologie ein (mittlere Spalte). Die rechte Spalte gibt Auskunft über die verwendete Programmiersprache / Basissoftware.

Terminologie			
Schicht	Module der Anwendung	AppsWarehouse	InstantView
	Interpreter	InstantView	C++
	Bausteine zur Visualisierung		
	Geschäftsobjekte	CyberEnterprise	C++
	Datenbank		ObjectStore

Hauptmerkmal ist die strikte Trennung zwischen Modell und Oberfläche, d. h. zwischen Geschäftsobjekten und Visualisierung. Letztere Aufgabe wird von der Softwareschicht InstantView® übernommen. Hier finden wir die Komponenten, welche die Interaktion mit dem Anwender organisieren. InstantView® zeigt sich nach außen als eine einfache Programmiersprache, mit der das User-Interface betriebswirtschaftlicher Anwendungen und das dynamische Zusammenwirken der Geschäftsobjekte beschrieben wird.

¹ Eine Grundvoraussetzung ist ein hohes Abstraktionsniveau der Komponenten, also genau das Gegenteil einer Sammlung von Programmcode für möglichst viele Probleme, der über Parameter gesteuert und mit Schaltern aktiviert/deaktiviert wird.

Das CyberEnterprise® ist eine Sammlung von C++-Klassen. Die Instanzen dieser Klassen – die Geschäftsobjekte – sind die Daten eines Unternehmens und werden in einer objektorientierten Datenbank (ObjectStore) gespeichert.

Dynamisches Interface zu Geschäftsobjekten

Die Software-Schicht InstantView® stellt mächtige, auf die Bedürfnisse betriebswirtschaftlicher Anwendungen zugeschnittene Bausteine für Darstellung und interaktive Bearbeitung der Daten zur Verfügung. Die entsprechenden C++-Klassen sind nicht direkt zugänglich, sondern werden über eine einfache Sprache erreicht, die der InstantView®-Interpreter ausführt. Dies erlaubt es, die Oberfläche zur Programmlaufzeit - z.B. direkt bei der Diskussion mit einem Anwender – zu modifizieren. Damit erhält ClassiX® Eigenschaften eines RAD-Werkzeugs.

Der Interpreter schützt das darunter liegende System, gewährleistet hohe Fehlertoleranz. Änderungen an den Anwendungen stellen ein nicht mehr so hohes Fehlerrisiko dar, und der geänderte InstantView®-Code kann bei laufendem System ausgetauscht werden.

InstantView®-Anweisungen für den Datentransfer zwischen (Geschäfts-)Objekten und der Oberfläche stützen sich auf die Definition einer Abbildung der Daten auf die Elemente der Windowoberfläche. Damit ist die Ein- und Ausgabe beliebiger Daten - *die in allen Anwendungen immer wiederkehrende Aufgabe* - besonders einfach und elegant zu beschreiben.

InstantView® ist objektorientiert mit all seinen Facetten, was sich auf die Module niederschlägt, wobei sich die Vererbung auch auf die Window-Ressourcen bezieht: Man kann die Windowoberfläche eines Moduls durch Ableitung modifizieren, Man vermeidet so das Kopieren von Codesegmenten und die damit verbundenen Probleme der Softwarewartung (siehe 2.10.).

InstantView®-Anweisungen haben natürlich vollen Zugriff auf alle Datenfelder der Geschäftsobjekte und können deren Methoden aufrufen.

InstantView® ist das dynamische Interface zu den Geschäftsobjekten, das

- die Visualisierung organisiert und
- das Zusammenwirken der Geschäftsobjekte steuert.

Der letzte Punkt beinhaltet, dass es eine Aufgabenteilung zwischen dem C++-Code in den Methoden der Objekte und mit InstantView® formulierten Algorithmen gibt, der die Anpassung an ständig variierende Anforderungen erleichtert. Dies ist das Thema des folgenden Abschnitts.

Maximale Unterstützung des Customizing

Aus Standard-Komponenten sollen individuelle Anwendungslösungen aufgebaut werden.

Diese wichtige Zielstellung beim Entwurf des ClassiX®-Systems hat Einfluss auf die Software-Architektur: Die Algorithmen und Programmteile, die stabiles Wissen einer Anwendungsdomäne repräsentieren, sollen von jenen Komponenten, die sich von Anwendung zu Anwendung unterscheiden und der Anpassung an sich ständig wandelnde Gegebenheiten unterliegen, soweit wie möglich getrennt werden.



Die C++-Klassen für die Geschäftsobjekte sind mit einem hohen Anspruch an Allgemeingültigkeit entworfen und enthalten nur die wesentlichen Datenfelder und Methoden. Sie sind trotzdem als Bausteine auch für sehr spezielle Anforderungen geeignet, da

- sie durch dynamische Datenfelder erweiterbar sind.
- mittels dynamischer Datenfelder beliebige Relationen zwischen Objekten aufgebaut werden können.
- die Berechnungsvorschriften und Auswahlregeln aus dem Programmcode in diverse Objekte verschoben sind.
- das Zusammenwirken der Objekte auf verschiedene Weise mit InstantView®-Anweisungen gesteuert werden kann.
- besondere Algorithmen auch mit InstantView® formuliert werden können.

Module sind in InstantView® geschriebene Programmteile. Sie entsprechen einer Teilaufgabe bzw. einem Arbeitsschritt der Anwendung und bestimmen, wie sich eine Anwendung dem Benutzer präsentiert. Die Module sind in einer umfangreichen Bibliothek, dem AppsWarehouse®, zusammengestellt. Somit stehen für gleiche Teilaufgaben diverse Module zur Verfügung.

Eine Anwendung wird erstellt, in dem

1. im Idealfall geeignete Module ausgewählt und zusammengestellt werden.
2. ein nicht ganz passendes Modul durch Vererbung erweitert wird.
3. InstantView®-Code von Grund auf neu geschrieben wird.
4. bei Bedarf das CyberEnterprise® durch neue C++-Klassen erweitert wird.

Diese Fälle zeigen die Erweiterbarkeit des ClassiX®-Systems auf verschiedenen Ebenen, mit von 1. bis 4. steigendem Aufwand. Wobei die Punkte 3 und 4 prinzipiell vermieden werden sollten, da sie einen hohen Aufwand darstellen.

Der Name „InstantView“ ist programmatisch gemeint: Ohne großen Aufwand können Mini-Applikationen für den Augenblick (ein bestimmter View auf die Daten) erzeugt werden, um operative Aufgaben schnell zu lösen. Am obersten Rand der Skala von ‚beständig‘ bis

‚flüchtig‘ steht InstantView® als Kommando-Interpreter. Aufgaben der Datenbankwartung und Datenpflege werden auf diese Weise gelöst.

Grundmechanismen betriebswirtschaftlicher Anwendungen

Das ClassiX® -System stellt eine Reihe von Diensten zur Verfügung, um die Modellierung betriebswirtschaftlicher Gegebenheiten zu erleichtern:

- **Einheitenarithmetik**

Für Berechnungen steht ein Objekt zur Verfügung, das mit einem Tupel aus Zahlenwert und Dimensionsangabe rechnet. Der Programmcode legt nicht fest, mit welchen Einheiten in einer Anwendung gerechnet wird. Diese Einheitenarithmetik ist eine Grundvoraussetzung für die im folgenden Punkt erwähnten Formel-Objekte. Der Anwender kann eigene Einheiten und Umrechnungsvorschriften definieren.

- **Berechnungsvorschriften, Bedingungstabellen**

Berechnungsvorschriften und Bedingungstabellen sollen kein Bestandteil des Programmcodes, sondern vom Anwender zu definierende Daten sein. Ein Formel-Objekt (CX_FORMULA) und ein darauf aufbauendes Objekt für Bedingungstabellen (CX_CONDITIONED_BAG) stehen dafür zur Verfügung.

- **Beträge**

Wenn in den Geschäftsobjekten Beträge vorkommen (Preis, Menge, ...), soll nicht von vornherein festgelegt werden, ob es sich um einen festen Wert handelt, ob er zu berechnen oder ob er in einer Tabelle zu suchen ist. Um all diese und weitere Möglichkeiten offenzuhalten, existiert die Klasse CX_AMOUNT, die den Betrag als eine polymorphe Abstraktion behandelt.

- **freie Definition von Merkmalen**

Aufträge, Produkte usw. werden oft durch Merkmale beschrieben, die vom Anwender festgelegt werden müssen². Dynamische Datenfelder lösen dieses Problem. Mit den oben erwähnten Formel- und Conditioned-Bag-Objekten können Berechnungen und Auswahlalgorithmen von den Merkmalswerten abhängig gemacht werden.

- **Standardwerte**

Oft wird ein Wert nur dann explizit angegeben, wenn er von einer Standardvorgabe abweicht. Der Standardwert kann eine Eigenschaft des Datenfeldes sein; es kann aber auch sein, dass der Wert von einem logisch übergeordnetem Objekt zu übernehmen ist – eine Konstruktion, die sich über beliebig viele Hierarchiestufen erstrecken kann. ClassiX® unterstützt einen semantisch gesteuerten Zugriff auf Datenelemente.

- **Enumerationen**

Traditionelle Anwendungen arbeiten mit einer Vielzahl von Schlüsseln und Abkürzungen und muten dem Anwender zu, diese zu kennen. Im ClassiX® -System kann für solche Datenfelder eine Übersetzungstabelle registriert werden. Auf der Bedienoberfläche erscheint dann immer der Begriff in der aktuell eingestellten Sprache. Die Übersetzungstabellen werden vom Anwender gepflegt.

- **Aspekte**

Eine Relation zwischen zwei Objekten ist oft mit einer Bedeutung belegt, die nur dieser Beziehung eigen ist, nicht aber den beiden in Beziehung stehenden Objekten: Das referenzierte Objekt wird vom referenzierenden Objekt unter einem bestimmten Aspekt gesehen. ClassiX® stellt hierfür eine *Deskriptive Referenz* zur Verfügung.

² da sie zum Zeitpunkt der Programmentwicklung noch nicht bekannt sind

- **Werte überschreiben**
Wenn aus Stammdaten Bewegungsdaten erzeugt werden, kommt man oft nicht mit einer Referenz auf das Stamm-Objekt aus, da die neu erzeugten Daten in einigen Datenfeldern abweichende Werte haben. Mit einer *Überschreibenden Referenz* wird für solche Fälle eine Kopie vermieden. Mit diesem Konstrukt wird der Wert eines Datenfeldes eines Objekts davon abhängig, über welchen Navigationspfad man dieses Objekt erreicht.
- **Gültigkeit eines Objekts**
Bisweilen stehen für einen Sachverhalt verschiedene Objekte, von denen immer eines zu einem bestimmten Zeitpunkt oder unter einer bestimmten Bedingung gültig ist. Diese Objekte können zu einem Cluster zusammengefasst werden, und eine Referenz auf ein Objekt des Clusters wird automatisch zum gerade gültigen Objekt umgeleitet.
- **Objekte unter einer beliebigen Bezeichnung suchen**
Oft werden die gleichen Dinge in verschiedenen Bereichen eines Unternehmens mit unterschiedlichen Namen bezeichnet (z.B. technisch orientiert – verkaufsorientiert). Geschäftsobjekte können in einem Dictionary-Objekt unter beliebig vielen Bezeichnungen (mehrsprachig) eingetragen und wieder aufgefunden werden.
- **Nummernkreise**
Fast alle Belege eines Unternehmens werden aus organisatorischen Gründen fortlaufend nummeriert. Auch ist es sinnvoll, Stammdaten (z. B. Artikel) mit einer eindeutigen Kennzeichnung (Nummer) zu versehen. ClassiX® stellt Zähler-Objekte zur Verfügung, die über Bezeichner ansprechbar sind.
- **COM-Objekte**
In der ClassiX® -Datenbank können COM-Objekte gespeichert und mittels dynamischer Datenfelder beliebig mit Geschäftsobjekten verbunden werden. Alle Funktionen, die ein COM-Objekt über das DISP-Interface zur Verfügung stellt, können auch mit InstantView® aufgerufen werden. Damit ist beispielsweise eine nahtlose Integration aller Microsoft-Office Produkte gewährleistet.

Hohe Abstraktion der Datenspeicherung

Bei der Arbeit mit Geschäftsobjekten übernimmt ClassiX® das gesamte Speicherungs- und Transaktionsmanagement. Standardmäßig ist die objektorientierte Datenbank ObjectStore transparent eingebunden.

Die Datenbank kann logisch in Layer und Domains aufgeteilt werden, um z. B. Daten verschiedener Mandanten zu trennen oder um eine Datenmenge nach Anwendungsbereichen zu unterteilen (Trennung der Lagerverwaltung für Fertigungsteile, Rohstoffe, Büromaterial, usw.).

Objekte werden in der Datenbank zunächst logisch gelöscht, d.h. in einen 'Papierkorb' verschoben, aus dem heraus sie auch wieder aktiviert werden können.

Reduzierte Netzbelastung

ClassiX® stellt einen eigenen Server-Prozess zur Verfügung, um z. B. Queries auf dem Server ablaufen zu lassen. Somit wird nur noch das Ergebnis einer Query über das Netz zum Client transferiert.

Internationalisierung

Sowohl in den Windowobjekten der Oberfläche als auch in den Daten der Anwendung (den Geschäftsobjekten) können Texte in beliebig vielen Sprachen vorkommen. Dies gilt selbstverständlich auch für Fehlermeldungen und Hilfetexte. Diese können zur Laufzeit jederzeit gewechselt werden.

Das Konzept der Locales wurde wesentlich erweitert. Jeder Anwendungs-Client kann entsprechend regional geltender Eigenheiten eingestellt werden: Landeswährung, Sprache, Zeitzone, Maßeinheiten, Zahlen- und Datumsformate, Feiertage, (Schul-)Ferien, usw. Die Regionalisierung kann beliebig tief geschachtelt werden (Land, Bundesstaat, Stadt, Unternehmen, Werk, ...).

InstantView®-Design-Tool

Zu InstantView® gehört ein interaktives Design-Tool (Workbench), das den InstantView®-Code für eine Oberflächenbeschreibung automatisch generiert.

Dieses Werkzeug ist selbst mit InstantView® geschrieben: InstantView® bearbeitet hier statt der Geschäftsobjekte die Windowobjekte der Oberfläche.

InstantView®-Code kann alternierend mit der Workbench oder mit einem Editor bearbeitet werden. Der Codegenerator des Design-Tools modifiziert bereits existierenden Code, wobei die mit dem Editor getroffene Formatierung und Kommentare erhalten bleiben.

Da die Workbench selbst aus InstantView®-Code besteht, kann sie als leicht zu modifizierender Baukasten eines Design-Tools gesehen werden. Eine reduzierte Version – MiniWorkbench – kann in eine Anwendung integriert werden, um schnelle Änderungen zu unterstützen.

Ausblick

Die Architektur des ClassiX® -Systems mit scharf gegeneinander abgegrenzten Schichten dient nicht zuletzt dazu, die Integration neuer Technologien und die Anpassung an sich ständig weiterentwickelnde Hard- und Softwareumgebungen zu ermöglichen.

Um den Weg für E-Commerce-Anwendungen zu ebnet, wird

- für InstantView® eine Scripting-Engine (InstantWeb) entwickelt, mit der InstantView® als Scriptsprache für den Internet-Explorer zur Verfügung stehen wird.
- es möglich sein, Anwendungen mit Java zu erstellen. Es wird Windowobjekte als Java-Klassen geben, die einen großen Teil der im InstantView®-Runtime-System vorhandenen Funktionalität benutzen - u. a. die Bindung der Daten der Geschäftsobjekte an die Elemente der Oberfläche.

1.2 Konzeption des Unternehmensmodells CyberEnterprise®

Das ClassiX® Basissystem mit InstantView® stellt einen mächtigen Apparat zur Verfügung, um einen Benutzerdialog schnell aufzubauen. So sollen die Geschäftsobjekte eines hoch abstrakten Unternehmensmodells - die Bausteine des CyberEnterprise® - die effiziente Programmierung komplexer betriebswirtschaftlicher Anwendungen ermöglichen. Immer wiederkehrende Datenstrukturen und Berechnungsalgorithmen müssen so aufbereitet werden, um Bausteine aller betrieblichen Anwendungen sein zu können:

Prinzipien der Modellierung der Geschäftsobjekte

- breite Funktionalität (intelligente Objekte)
- hohe Abstraktion (Wiederverwendbarkeit der Objekte)
- große Übersichtlichkeit (nicht zu viele Objekttypen)
- gute Lesbarkeit (leicht verständliches Modell)
- hohe Flexibilität (beliebige Kombination der Objekte)
- geringer Speicherbedarf (schlanke, performante Objekte)
- leichte Erweiterbarkeit (klar strukturiertes Modell)

Geschäftsobjekte als funktionale Bausteine

Das wichtigste Prinzip eines Komponentensystems ist die hohe Abstraktion seiner Elemente. Geschäftliche Daten und Abläufe müssen auf wenige, unterschiedliche Strukturen reduziert werden. Die beliebige Kombinier- und Erweiterbarkeit dieser Bausteine bilden dann die Vielfalt unternehmerischer Informationen und Prozesse dar.

Innerhalb des CyberEnterprise® gibt es drei Ebenen von Objekten verschiedener Komplexität:

1. **Basis-Objekte** kapseln Informationen über Datum, Zeit und Werte; es sind Objekte nur begrenzter Funktionalität.
2. **Informationsobjekte** verarbeiten Attribute oder Sachmerkmale von Artikeln, Adressen, Zu- oder Abschläge, Allokationen (Stücklisten oder Arbeitspläne), Kalendarien, Transaktionsobjekte, Monitore (Konten und multidimensionale Datenwürfel), usw. Hier findet man aus funktioneller Sicht die meisten Bausteine.
3. **Geschäftsobjekte** bedienen sich dann nur noch dieser Basis- und Informationsobjekte, um ihre eigenen Daten und Methoden zu verwalten. Diese Geschäftsobjekte sind am ehesten vergleichbar mit den Entitäten traditioneller, betriebswirtschaftlicher Software.

Hinzu kommen

- Werkzeugklassen, für Druckausgaben, für den Import und Export von Daten und
- Klassen für COM-Objekte.

Jedes Element (C++ Klasse) des CyberEnterprise® definiert darüber hinaus sein Verhalten für

- die arithmetischen Operatoren (+, -, *, /).
- die Vergleichsoperatoren.

- die Konvertierung in eine Zeichenkette.
- den Import einer Zeichenkette.
- den Zugriff auf Datenfelder, Bindung der Datenfelder an Variable.
- die Erweiterung um weitere Datenfelder und Referenzen zu anderen Objekten.
- den Aufruf von Methoden,

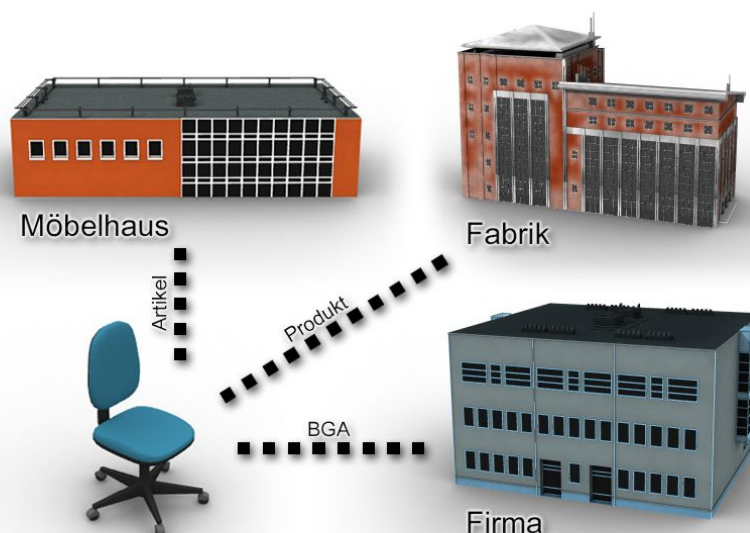
Natürlich kann man nicht alle Objekte addieren - schon gar nicht mit beliebigen anderen Objekten. Deshalb erbt jede Klasse zunächst als Reaktion:
Signalisieren einer Ausnahmebedingung - d. h. die Operation ist nicht unterstützt.

Reale Objekte und Begriffe: die eigentlichen Geschäftsobjekte

Das statische Abbild von Unternehmen wird beschrieben durch die realen Objekte: Partner, Sachen und Geld. Diese Objekte repräsentieren Daten-Entitäten, die für alle am Wirtschaftsprozess teilnehmenden Parteien gleich erscheinen und sind.

Die aus Sicht des zu modellierenden Unternehmens (Mandant) relevanten Informationen und Eigenschaften realer Objekte werden mittels Begriffe abgebildet, die in direkter Beziehung zu einem realen Objekt stehen: z. B. beschreibt der Begriff "Kunde" die Rolle eines Partners, der vom Mandanten Waren oder Dienstleistungen gegen Entgelt bezieht. Die beschreibende Eigenschaft "Kunde" enthält nur die Daten, die für diese Geschäftsbeziehung notwendig sind. Dem gleichen Partner können weitere Begriffe (oder Eigenschaftsobjekte) zugeordnet werden, wie z. B. der Begriff "Lieferant". Auch können gleiche Begriffe mehrfach einem realen Objekt zugeordnet werden (z. B. verschiedene "Kunde" Eigenschaftsobjekte mit unterschiedlichen Konditionen für verschiedene Produktgruppen oder Projekte).

Einem realen Objekt werden - meist aus rein abrechnungstechnischer Absicht - gleichzeitig verschiedene Begriffe zugeordnet, wie hier am Beispiel eines realen Objektes "Stuhl" gezeigt:



Darüber hinaus repräsentieren Begriffe auch alle organisatorischen Einheiten, wie Kostenstelle, Auftrag oder Bilanzposition. Begriffe oder Abrechnungsobjekte können als semantische Aspektklassen zu den realen Geschäftsobjekten aufgefasst werden.

Dynamisierung des Modells durch Transaktionen

Jede Tätigkeit in einem Unternehmen, die - aus Gründen der Organisation und einer effizienteren Gestaltung - mittels der EDV abgebildet werden soll, wird durch Transaktionsobjekte beschrieben. Meist werden derartige Vorgänge von Belegen begleitet, wie Auftragsbestätigungen, Lieferscheine, Rechnungen, Materialentnahmescheine, Buchungsbelege im Rechnungswesen oder ausgehende Bestellungen.

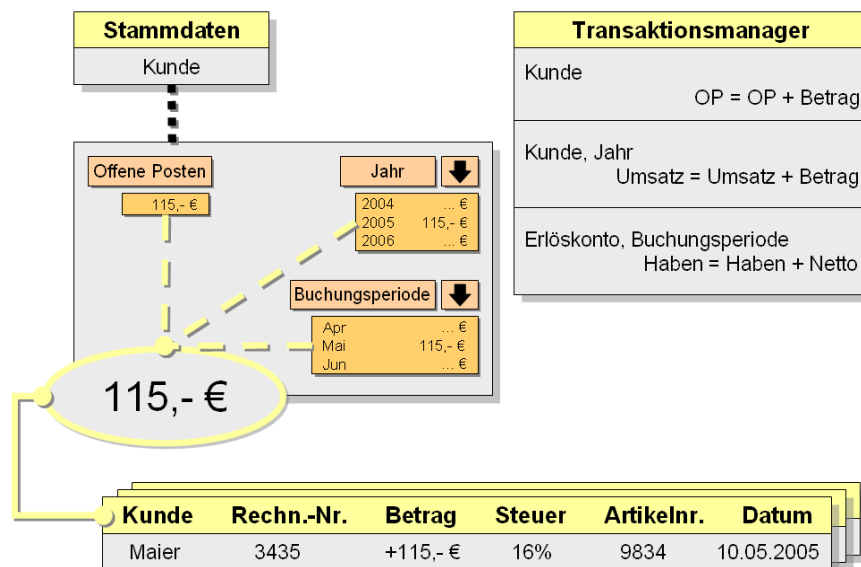
Diese Vorgänge oder Transaktionen führen immer zu einer (Zustands-)Änderung von Geschäftsobjekten. Eine Auftragsbestätigung erhöht den Auftragsbestand, vermindert eventuell den verfügbaren Bestand eines bestellten Lagerartikels. Auch das Editieren einzelner Datenfelder (Ändern einer Artikelbezeichnung) kann mittels Transaktionsobjekte ausgeführt werden, um eine vollständige Historie der Geschäftsprozesse (in diesem Beispiel: Pflege von Artikeldaten) zu erhalten.

Meist interessiert neben den Daten jeder einzelnen Transaktion (Datum der Auftragsbestätigung, Höhe der Rechnung, Entnahmemenge eines Artikels) auch die Kumulierung einzelner Werte, und das mit bestimmten Dimensionen (z. B. Auftragseingang pro Monat und Verkaufsgebiet). Man spricht hier von einer mehr- oder multi-dimensionalen Datensicht: Der Wert "Auftragseingang" soll jeweils in der Kombination aus Periode und Verkaufsgebiet gespeichert werden, also z.B. 412.350,-€ Auftragseingang im Monat Juli 2006 im Verkaufsgebiet Nord.

Multidimensionale Daten werden in sogenannten Datenwürfeln oder Zellen gespeichert, in unserem Beispiel der Auftragseingang mit den beiden Dimensionen Periode und Verkaufsgebiet haben. Ein derartiger Datenpool eignet sich in hervorragender Weise für OLAP (on-line analytical processing).

Der Transaktionsmanager

Welche Daten welcher Geschäftsobjekte verändert werden sollen oder welche Werte mit welchen Dimensionen gespeichert werden sollen, wird durch Einträge in einen Transaktionsmanager gesteuert. Jeder Vorgang besitzt seinen eigenen Transaktionsmanager.



Im gezeigten Beispiel handelt es sich um eine Transaktion "Rechnung", für die ein Transaktionsmanager folgende Verarbeitung vorsieht:

1. Addiere den Rechnungsbetrag zum OP-Konto des Kunden (Datenwürfel mit nur einer Dimension: Kunde)
2. Addiere den Rechnungsbetrag in einen Datenwürfel (Umsatzkonto) mit den Dimensionen Kunde und Jahr (Umsatzstatistik pro Kunde und Jahr)
3. Addiere den Netto-Rechnungsbetrag in einen Datenwürfel (FiBu-Konto) mit den Dimensionen Erlöskonto und Buchungsperiode

Die so automatisch entstehenden Datenwürfel - eindimensional (wenn die Periode nicht mitgezählt wird!) meist als Konten bezeichnet - registrieren die Mengen- und Wertflüsse, die durch Transaktionen erzeugt werden. Alle Dimensionen der gewünschten Datenwürfel (z. B. das Erlöskonto) müssen eindeutig aus den Daten innerhalb der Transaktion ermittelt werden können (z. B. aus dem Artikel).

Konsolidierungspfade innerhalb von Dimensionen (Umsatzstatistik auch per Kundengruppe und Jahr) liegen im Ermessen derer, die einen Vorgang mit dem Transaktionsmanager beschreiben und werden nicht automatisch generiert. Auch wenn höher aggregierte Datenwürfel nicht gleich bei jeder Transaktion fortgeschrieben werden, so lassen sich diese Daten jederzeit durch Kumulierung ermitteln. Vollständiges drill-down oder roll-up wird gewährleistet.

2 Spezielle Objekte des CyberEnterprise®

2.1 Attribute

In ClassiX® können beliebige Datenfelder dynamisch definiert und in jedem Geschäftsobjekt zusätzlich gespeichert werden (dynamische Datenfelder oder *slots*). Eine Definition dieser Datenfelder nimmt üblicherweise der Systemadministrator vor, in Verbindung mit Erweiterung der Anwendungen oder Bildschirmmasken, aus denen heraus diese neuen Datenfelder gepflegt werden können.

Dynamische Datenfelder werden wie normale, in den Klassen fest definierte Variablen verwendet und definieren nur den Datentyp. Die Bedeutung des Datenfeldes ergibt sich für den Anwender nur aus dem Prompt auf dem Bildschirm oder dem Titel auf einem Ausdruck.

Ein Objekt der Klasse `CX_SLOT_ATTRIBUTE` kapselt ein dynamisches Datenfeld und ermöglicht es somit dem Anwender weitere Informationen diesem slot hinzuzufügen: z. B. einen mehrsprachigen Namen, einen Standardwert oder auch einen Gültigkeitsbereich. Auch diese Attribut-Objekte können Geschäftsobjekten beliebig häufig hinzugefügt werden.

So kann z. B. ein Artikel zusätzlich mit den Attributen bzw. Sachmerkmalen für Farbe, Durchmesser oder Gewicht versehen werden. Dabei kann eine Standardfarbe vorgesehen werden oder ein Maximalgewicht definiert sein. Diese Angaben sind auch ohne Änderung oder Erweiterung von Anwendungen oder Bildschirmmasken möglich.

Die Verwaltung mehrerer `CX_ATTRIBUTE` Objekte übernimmt die Klasse `CX_ATTRIBUTE_SET`, die quasi eine Erweiterung eines Geschäftsobjektes darstellt. Ein solches `CX_ATTRIBUTE_SET` Objekt kann auch über ein `CX_CONDITIONED_BAG` erreicht werden, falls in Abhängigkeit bestimmter Bedingungen verschiedene Attribut-Werträume modelliert werden sollen.

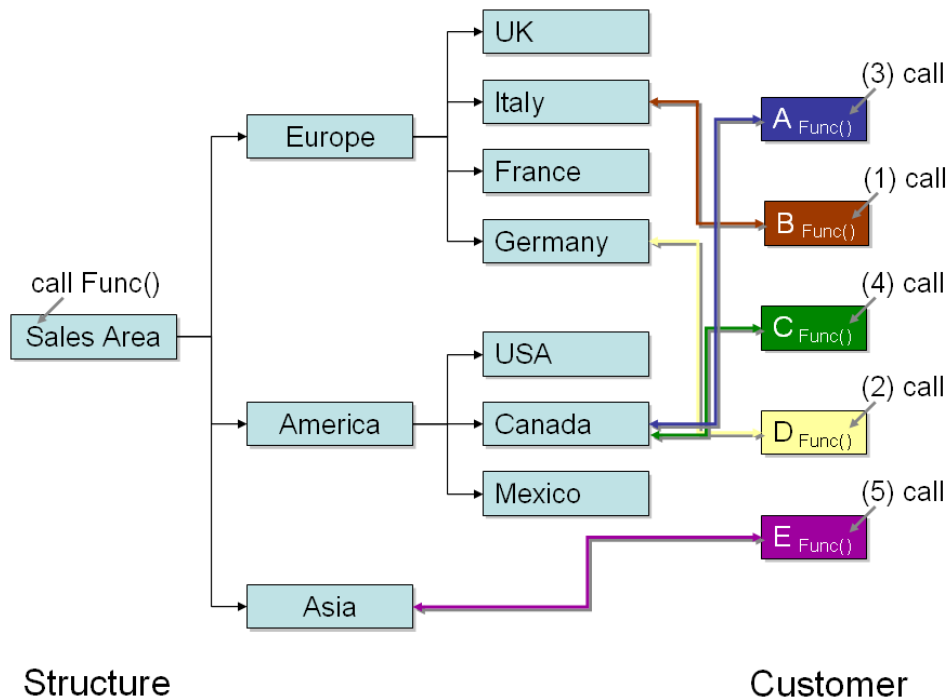
Die Klasse `CX_FORMULA_ATTRIBUTE` ermittelt den Wert eines dynamischen Datenfeldes immer nur durch Berechnung mit einer Formel, z. B. bei dem Attribut "Fläche" aus den Attributen "Breite" und "Länge". Diese Berechnung findet immer aktuell bei Abfrage des Wertes dieses Attributes statt.

Hat ein dynamisches Datenfeld Werte, die von bestimmten Bedingungen abhängig sind, kann ein Objekt der Klasse `CX_CONDITIONED_ATTRIBUTE` eingesetzt werden. Diese Bedingungen können Werte anderer Attribute sein. (Beispiel: Wenn Montagepunkt = rechts oder Montagepunkt = beidseitig, dann Länge_rechts = Länge; wenn Montagepunkt = links oder Montagepunkt = beidseitig, dann Länge_links = Länge).

2.2 Strukturen

Klassifikationsmerkmale (Teileart, Artikel- oder Kundengruppen) werden als eigenständige Strukturobjekte hinterlegt und sind nicht mehr Sortierkriterium innerhalb der Objekte. Alle Geschäftsobjekte können beliebig (häufig) zugeordnet werden, wobei durch überschreibende oder beschreibende Referenzen eine Identität geschaffen werden kann.

Strukturobjekte ihrerseits können hierarchisch zugeordnet werden.



Die Funktionalität der Geschäftsobjekte kann hierarchisch durch eine gesamte Struktur hindurch additiv aufgerufen werden (z.B. Umsätze sortiert nach Kundengruppen).

2.3 Allokationen

Allokationen sind zu verstehen als (Hilfsobjekte für) mengenmäßige Verteilungs- und/oder Zuordnungsstrukturen.

Ein Objekt der Klasse `CX_SINGLE_ALLOCATION` besteht im Kern aus dem Tupel aus Referenz zu einem Objekt und einer Menge: Das referenzierte Objekt wird in einer bestimmten Menge "verwendet". Die Menge ist vom Typ `CX_AMOUNT` (s. o.). Das referenzierte Objekt kann selbst eine Allokation zugeordnet haben, wodurch hierarchische Allokationsbäume entstehen können.

Wird ein Objekt vom Typ `CX_SINGLE_ALLOCATION` von einem anderen Objekt aus referenziert, dann stellt es sich diesem als "Ressource" dar.

Die Klasse `CX_CONDITIONED_ALLOCATION` erlaubt es, bedingte Allokationen darzustellen (Variantenverarbeitung).

Objekte vom Type `CX_SET_ALLOCATION` verwalten Mengen von Allokationen und repräsentieren damit Stücklisten, Arbeitspläne, aber auch Kostenumlagen. Beim Auflösen solcher Allokationen werden die Mengen an den Knoten mittels von `CX_PRODUCT_AMOUNT` Objekten dargestellt. Die Änderung einer Menge an irgendeinem Knoten bewirkt die sofortige Korrektur aller Mengen tiefer liegender Allokationen.

2.4 COM-Objekte

Die Klasse CX_COM_OBJECT ermöglicht es, beliebige COM-Objekte in der Datenbank zu speichern (sie besitzt eine eigene Implementierung des Interfaces ILockBytes). Alle Funktionen des DISP-Interface können – wie jede andere Funktion eines Geschäftsobjektes – auch mit InstantView® aufgerufen werden.

Geschäftsobjekt CX_PERSON

Peter Panther

Maximale Unterstützung des Customizing

Aus Standard-Komponenten sollen individuelle Anwendungslösungen eine wichtige Zielstellung beim Entwurf des ClassiX-Systems sein. Die Architektur: Die Algorithmen und Programmteile, die stabiles Verhalten repräsentieren, sollen von jenen Komponenten, die sich von Änderungen unterscheiden und der Anpassung an sich ständig wandelnde Gegebenheiten unterliegen, soweit wie möglich getrennt werden.

variabel, ständige Anpassung an neue Gegebenheiten

Kommandos, operative Eingänge, temporäre Anwendungen, Applets (Applets)

algorithmische von InstantView

Geschäftsobjekt Cyber-Enterprise das Betriebssystem

unveränderlich

Die C++-Klassen für die Geschäftsobjekte sind mit einem hohen Anspruch an Alle...

Save

Name	Änderung
Panther Peter männlich	5.2.2000 21.0
Lou Anreas-Salomé weiblich	5.2.2000 20.4

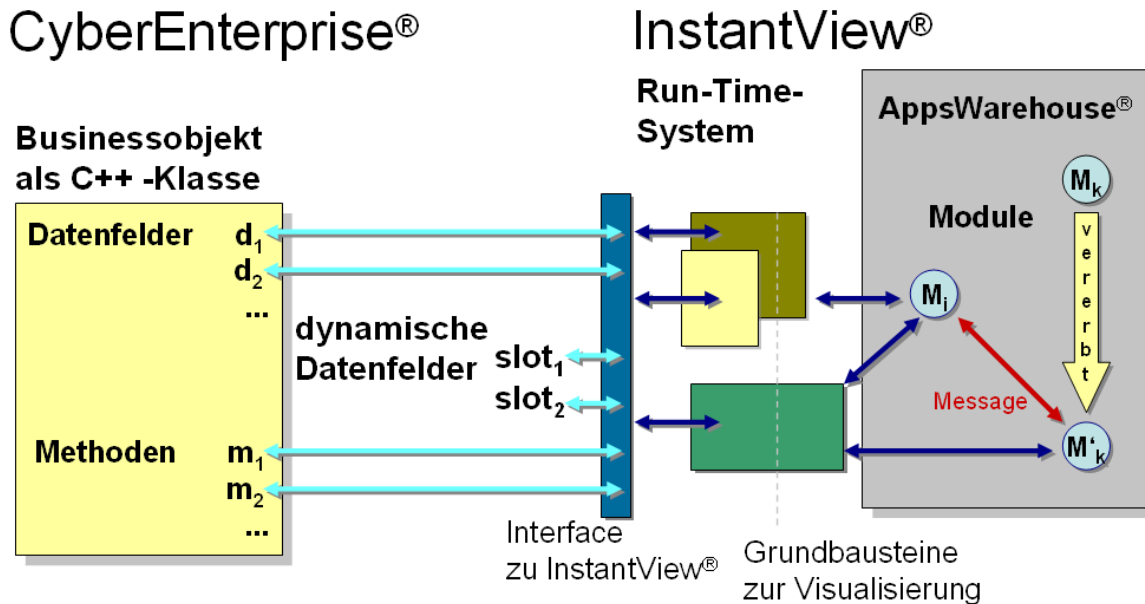
Methods

- int GetGridSpaceBetweenHorizontalLines(void)
- void PutGridSpaceBetweenHorizontalLines(int)
- int GetGridSpaceBetweenVerticalLines(void)
- void PutGridSpaceBetweenVerticalLines(int)
- void * GetGridOriginFromMargin(void)
- void PutGridOriginFromMargin(void *)
- void * GetKerningByAlgorithm(void)
- void PutKerningByAlgorithm(void *)
- void * GetJustificationMode(void)
- void PutJustificationMode(void *)
- void * GetFarEastLineBreakLevel(void)
- void PutFarEastLineBreakLevel(void *)
- char * GetNoLineBreakBefore(void)
- void PutNoLineBreakBefore(char *)
- char * GetNoLineBreakAfter(void)
- void PutNoLineBreakAfter(char *)
- void * GetTrackRevisions(void)
- void PutTrackRevisions(void *)
- void * GetPrintRevisions(void)
- void PutPrintRevisions(void *)
- void * GetShowRevisions(void)

Funktionen

Von CX_COM_OBJECT abgeleitete Klassen für bestimmte COM-Objekte – z.B. Microsoft-Word-Dokumente – benutzen das C++-Interface des speziellen COM-Objekts und können deshalb bestimmte Operationen mit höherer Performance durchführen.

Zum Schluss ein Blick auf das Zusammenspiel zwischen InstantView® und den Objekten des CyberEnterprise®:



zur Abbildung, von links nach rechts:

Die C++-Klassen der Geschäftsobjekte benötigen ein spezielles Interface, das Datenfelder und Methoden für InstantView® exportiert (der entsprechende C++-Code wird mit einem speziellen Tool generiert)

Das Laufzeit-System der Softwareschicht InstantView® enthält C++-Klassen, die u.a.

- das User-Interface organisieren (Grundbausteine zur Visualisierung).
- Operationen mit Objekten und Objektmengen (Collections) erlauben.
- den Objektcharakter der Applets implementieren (Kapselung durch Messages, Vererbung, Polymorphie).
- Dienste der Datenbank zugänglich machen.

Diese Dienste werden durch den in Module organisierten InstantView®-Sourcecode aufgerufen. Module kommunizieren ausschließlich durch Messages (und sind damit weitgehend entkoppelt) untereinander. Ein Modul kann problemlos durch ein Modul ähnlicher Funktionalität ersetzt werden.

2.5 Drucken

InstantView® unterstützt das Erzeugen von Drucklisten. Ausgabertexte, OLE-Objekte, Bitmaps und andere graphische Elemente wie Linien und Rechtecke können jederzeit an beliebige Positionen in der aktuellen Seite oder innerhalb des gesamten Dokuments platziert werden (Seitenkonzept).

Teile der Window-Oberfläche können in die Druckliste übernommen werden; bei einer Objektbox schließt dies auch die gerade nicht sichtbaren Zeilen ein.

Ist eine Druckliste breiter als der bei Windows bzw. OS/2 angemeldete Drucker, so wird die Liste auf mehrere Blätter verteilt.

2.6 Hilfe-System

Sowohl für die Online-Hilfe des Systems als auch für die der Anwendungen werden HTML-Files verwendet. Die einfache Verbindung zwischen InstantView®-Source-Code und HTML-Files versetzt Anwender in die Lage, die Hilfstexte mit eigenen Anmerkungen zu erweitern. Dabei kann jedem Window-Objekt eine eigene Hilfe zugeordnet werden.

2.7 Multimedia-Objekte, Spracheingabe

Die InstantView®-Anweisungen für die Dateneingabe *DrainWindow* und Datenausgabe *FillWindow* unterstützen auch Bitmaps (Graphiken) als Datenfelder der Geschäftsobjekte. Graphiken können aber auch als Hintergrund eines jeden Fensters oder als Symbole für einen Button benutzt werden.

In jedem geeignetem Oberflächen-Objekt kann ein Video-File abgespielt werden und es gibt Anweisung zum Abspielen von Wave-Files (Audio).

Eingabefelder können für Spracheingabe aktiviert werden, d. h. das Eingabefeld, das den Focus besitzt, würde sowohl Tastatureingaben als auch in das Mikrofon gesprochenen Text akzeptieren. Man kann Kommandos vereinbaren, d. h. es werden Tupel aus einem Wort und einer Message (siehe 2.4.) angemeldet. Wenn ein als Kommando angemeldetes Wort erkannt wird, wird es nicht Bestandteil des eingegebenen Textes sondern die mit dem Wort verbundene Message wird ausgelöst.

3 Index

Anweisung.....		CX_AMOUNT.....	7
DrainWindow.....	18	CX_CONIDTIONED_BAG.....	7
FillWindow.....	18	CX_DESCRIPTIVE_REF.....	7
Applet.....	4	CX_DICTIONARY.....	8
Aspekt.....	7	CX_FORMULA.....	7
C++.....	5, 6	CX_OVERWRITING_REF.....	8
Concept.....	3	CX_STRUCTURE.....	15
CyberEnterprise.....	3, 6, 10	CX_VALUE.....	7
Dynamische Datenfelder.....	6, 7, 14	Message.....	18
Enumeration.....	7	multilingual.....	8, 9
Gültigkeit eines Objekts.....	8	ObjectStore.....	5, 8
InstantView.....	4	Transaktion.....	12
InstantView-Interpreter.....	5, 7	Vererbung bei Applets.....	5
Internationalisierung.....	9	Workbench.....	9
Klasse.....			