



Institut für komplexes
Datenmanagement AG

Am Niddatal 3
60488 Frankfurt am Main

Tel.: 069 40 56 8 - 0
Fax: 069 40 56 8 - 111
E-Mail: info@arago.de
URL: <http://www.arago.de>

Kürzere Time-to-Market von Vertriebsanwendungen

**Entwicklungsbeschleunigung durch die Trennung von
Frontend, Prozess, Funktion und Datenhaushalt**

Von Hans-Christian Boos, boos@arago.de

Die Aufgabenstellung von Vertriebs- und Marketingverantwortlichen für ihre internen und externen IT-Dienstleister ist schnell formuliert: In immer kürzerer Zeit sollen möglichst fehlerfreie Marktbearbeitungssysteme erstellt werden. Diese Anforderung entsteht, weil die Zyklen im Vertriebs- und Marketingmanagement immer kürzer werden und weil IT-Systeme heute bei der Erreichung der gesteckten Ziele kaum wegzudenken sind. Die bekannten Entwicklungsprozesse sind hierfür jedoch nur bedingt geeignet.



Wer schon einmal Aussagen gehört hat wie: „So was hatten wir doch schon mal. Es kann doch nicht so schwer sein, eine ähnliche Anwendung zu entwickeln.“ oder: „Diese kleine Änderung zwingt uns, alles Neu zu machen!“ hat vermutlich ein Problem in der IT-Architektur, die seinen Entwicklungsprojekten zugrunde liegt.

Dieses Problem ist gerade bei Applikationen zur Vertriebsunterstützung keine Seltenheit. Denn diese unterliegen aufgrund der sich ständig

**Schnellere Marktreife von Anwendungen
mit aragos 4-Schichtmodell**

wandelnden Marktanforderungen einer enormen Änderungsfrequenz. Von der Produktpräsentation über die Unterstützung bei der Beratung bis hin zur Abwicklung der eigentlichen Transaktion und den dahinter liegenden Prozessen ist die passende technische Vertriebsunterstützung heute ein wichtiger Baustein jeder Produktplatzierung und jeder Marketing- oder Vertriebskampagne. Die technischen Komponenten werden auf dem Weg zum Projekterfolg jedoch nicht als „Hauptakteur“ betrachtet sondern gelten nur als „Beiwerk“. Die als weniger wichtig eingestuft aber gleichwohl erfolgskritischen IT-Systeme rund um den Vertrieb stehen damit unter besonderem Erfolgs- und Zeitdruck. Probleme bei deren Implementierung und Inbetriebnahme kommen dabei sehr schnell an die Oberfläche.

Die oben beschriebenen Symptome können aus zweierlei Richtungen verursacht werden. Entweder werden IT-Systeme „quick and dirty“ entwickelt, so dass jede Änderung einer Neuentwicklung des Systems gleichkommt. Für alle Folgeaktionen sind damit Implementierungszeit und Kosten erheblich. Oder die Systeme werden mit der größtmöglichen „informatischen Schönheit“ entwickelt, damit sie flexibel und vollkommen unabhängig von allen Datenquellen und verwendeten Programmbausteinen sind. In diesem Fall kommt es allerdings meist zu unsäglichen Projektlaufzeiten.

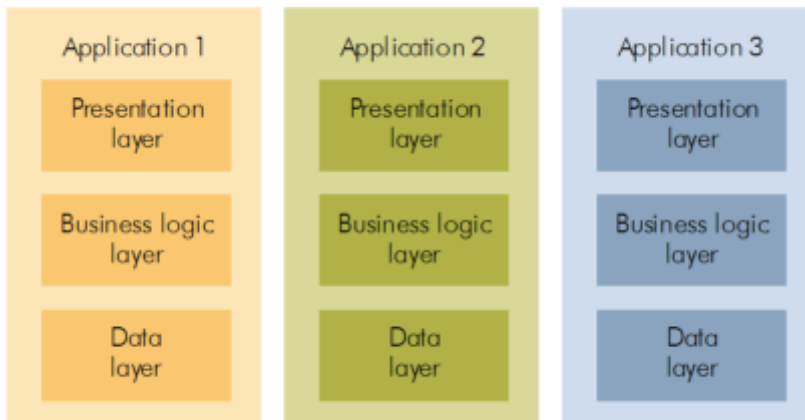
Keine der beiden Lösungen ist sinnvoll oder zufrieden stellend. IT sollte nicht gegen den Rat der Experten unsauber gecoded werden. Und genauso wenig kann IT „by the book“ in 7+x Iterationen entwickelt werden – denn das ist schlicht unbezahlbar und benötigt viel zu viel Zeit.

Das Dreistufenmodell (3-Tier)

Schlaue Köpfe haben deshalb schon vor vielen Jahren ein dreischichtiges Architekturmodell für die Gesamt-IT entwickelt. Dabei wird für jedes zu entwickelnde Programm eine strikte Trennung vorgeschrieben zwischen der Darstellungsebene, die der Benutzer zu sehen bekommt, der Programmlogik, die alle Verarbeitungsmechanismen vereint, und dem Datenhaushalt bzw. dem Zugriff aller notwendigen Daten. Diese Trennung wurde unter folgenden Kerngesichtspunkten vorgenommen:

1. Die Darstellung muss unabhängig sein, weil mehrere Oberflächen (z.B. OS/2 oder Windows) oder Oberflächen in mehreren Sprachen möglichst zeit- und kostensparend umgesetzt werden sollten, nämlich ohne alle Bausteine des Systems anzufassen.
2. Die Versorgung der Applikationen mit Daten (erster Ansatz zentrales Repository, nächster Ansatz Datawarehouses, dritter Ansatz EAI-Systeme) musste getrennt werden, weil hier die größten Abhängigkeiten eines IT-Systems zu anderen Systemen bestanden. Damit lagen hier auch die größten nicht zu beeinflussenden Änderungsfrequenzen und Qualitätsrisiken. Neben der Zeitersparnis bei der Anpassung auf neue Datenquellen oder Liefersysteme sollte die strikte Trennung des Datenhaushaltes auch die Wiederverwendung von Berechnungen oder aufwändig implementierten Plausibilitäten erleichtern.
3. Die eigentliche Programmlogik war nicht nur als der neben Darstellungsebene und Datenversorgung übrig gebliebener „Rest“ der

Anwendung eine eigene Schicht wert. Von der Logikschicht erhoffte man sich eine konsequent gute Entwicklung der eigentlichen IT-Funktionalität im Hinblick auf eine saubere Algorithmik für fehlerfreie sowie performante Programme und bezüglich einer hohen Wiederverwendbarkeit. Diese beiden Grundinteressen der Informatik sind mit der Abkopplung der Außenwelt - nämlich der Darstellung und der Datenversorgung - leichter umzusetzen.



Die Ebenen des Dreischichtmodells

Das dreischichtige Modell hat sich als sehr effektiv erwiesen und leistet auch heute noch gute Dienste. Es vermeidet das blinde „Drauf-Los-Codieren“ und sorgt für Nachvollziehbarkeit, weil es dem Prinzip der Arbeitsteilung folgt. Diese Teilung wurde dabei an leicht zu identifizierenden und logisch einleuchtenden Stellen vorgenommen. Hätte man grundsätzlich alle Anwendungen nach diesem Prinzip geteilt, wäre heute in vielen IT-Budgets erheblich mehr Raum für Innovationen und neue Projekte, als es aktuell der Fall ist.

Die Technologie bleibt nicht stehen

Seit der Einführung des Dreischichtmodells haben sich die technische Umgebung und die Welt der fachlichen Anforderungen enorm verändert. Die Datenmengen, mit denen Applikationen heute umgehen müssen, haben sich aus technischer Sicht exponentiell entwickelt. Dies ist neben dem Umstieg auf Web-Technologien oder der Einführung verschiedener Sicherheitszonen bei fast jeder Art von Anwendung vermutlich der Hauptgrund für die Entwicklung verteilter Systeme. Letzteres hat sich als die größte Änderung in der IT-Architektur herauskristallisiert.

Technologisch gesehen ist eine solche „Arbeitsteilung“ auch auf Systemebene sinnvoll und hat viele positive Effekte. Für Entwicklung und Entwickler bedeutet sie jedoch eine Herausforderung. In einer verteilten Umgebung, deren Komponenten nicht nur von einer einzigen Anwendung genutzt werden, können die Applikationen schnell in Konkurrenz um Ressourcen treten oder sich sogar gegenseitig blockieren und behindern. In einer solchen Umgebung ist eine Applikation kein geschlossenes System mehr, sondern eine Ansammlung von Bausteinen, die mit mehr oder weniger stabilen

Verbindungen zusammengefügt wurde und vielen externen Einflüssen unterliegt.

Diese technischen Änderungen verursachen naturgemäß eine Verlängerung der Entwicklungszeit und ein erhöhtes Risiko. Das steht wiederum im diametralen Widerspruch zu den Änderungen auf fachlicher Seite. Gleichzeitig mit der Datenexplosion und der Parallelisierung der Infrastruktur haben sich die Lebenszyklen – insbesondere von Vertriebsunterstützungssystemen – verkürzt bzw. deren Änderungszyklen massiv erhöht. Erschwerend kommt hinzu, dass beide Entwicklungen mit einer Reduktion der verfügbaren Budgets und einem immer stärkeren Risikobewusstsein einhergehen.

Ein neues Modell ist also erforderlich, um die fachlichen und technischen Anforderungen bedienen zu können.

Die Anforderungen definieren das Modell

Vor einer Veränderung des dreischichtigen Modells ist im ersten Schritt eine Bestandsaufnahme auf Seiten der Anforderungen notwendig. Diese lässt sich

am besten in die drei Gruppen gliedern: 1. ökonomische Anforderungen der Fachabteilung (Auftraggeber), 2. ökonomische Anforderungen der Entwickler (Auftragnehmer) und 3. technologische Anforderungen moderner IT Architekturen. Eine vertriebsunterstützende IT ist aus den bereits beschriebenen Gründen ein Vorreiter bei der Definition der neuen Anforderungswelten.

Diesen Anforderungen aus unterschiedlichen Perspektiven ist gemeinsam, dass Wiederverwendung und Flexibilität die beiden wichtigsten Ziele sind. Beides ist erforderlich: Sowohl aus Sicht des Kunden, der geleitet ist von der Time-to-Market und von Kostenüberlegungen als auch aus Sicht der Lieferanten, dem die Herstellungszeiten am Herzen liegen. Und das gilt selbstredend auch für die technisch abstrakte Sicht, nach der ohnehin keine Funktion doppelt programmiert werden darf.

Das Vierschichtmodell

Die Kriterien „Wiederverwendung“, schnelle Erzeugung“ und „Flexibilität in den Frontend-Funktionen“ vor Augen, kann nun das klassische Dreischichtmodell unter die Lupe genommen werden.

- 1. Fachlich getriebene ökonomische Anforderungen:**
 - a. Stark verkürzte Entwicklungszeiten (kürzere Time-to-Market)**
 - b. Wiederverwendung bestehender Module für neue Prozesse bzw. leichte Abänderungen, um neue Prozesse kostengünstiger schaffen zu können**
 - c. Unabhängigkeit der Frontends von der eigentlichen Anwendungsentwicklung (Frontends werden immer mehr Teil der Marketingstrategie)**
 - d. Offene Schnittstellen zur Erleichterung von Kooperationen**
 - e. Minimierung von Risiken (Sicherheits- und Projektrisiken)**
- 2. Technisch getriebene ökonomische Anforderungen:**
 - a. Wiederverwendung einmal geschaffener Funktionen, um Entwicklungszeiten und Entwicklungskosten zu minimieren**
 - b. Einfacher Umgang mit verteilten Systemen**
 - c. Automatisches Verhindern von klassischen Sicherheits- und Performancefehlern.**
- 3. Technologische Anforderungen:**
 - a. Unterstützung verschiedener Technologien und Technologiestandards innerhalb einer Anwendung**
 - b. Einfacherer Umgang mit stark wachsenden Datenmengen**
 - c. Automatische Parallelisierung oder zumindest automatische Verwaltung von Parallelisierung**
 - d. Offene/standardisierte Schnittstellen zwischen einzelnen Programmbausteinen, Datenblöcken und anderen Komponenten einer Anwendung, um diese flexibel erweitern oder neu zusammensetzen zu können**

Die Schicht der Datenversorgung gewährleistet, dass Datenquellen unabhängig von den Anwendungen angepasst werden bzw. Datenstrukturen in den Anwendungen verändert werden können ohne gleich den unternehmensweiten Datenhaushalt auf den Kopf zu stellen. Sie bleibt in ihrem Design und ihrer Funktion unangetastet. Es ist allerdings anzumerken, dass die Schicht des Datenhaushalts keine Berechnungen oder Plausibilitäten, sondern lediglich den Zugriff auf die Datenquellen in lesender und schreibender Form gewährleisten soll. Bei sehr breiten Datenhaushalten – sprich einer großen Menge an Daten aus sehr vielen verschiedenen Quellen – kann es sinnvoll sein, die Datenhaushaltsschicht aufzuspalten: in einen Zugriffsteil, der mit der Struktur der Datenquellen verwoben ist, und in einen Aufbereitungsteil, der den Applikationen die passende Kompilation an Daten zur Verfügung stellt. Bei weniger breiten Datenhaushalten ist - unabhängig von der physikalischen Datenmenge - diese Aufspaltung zwar technisch gesehen sehr schön, aber pragmatisch einfach nicht erforderlich.

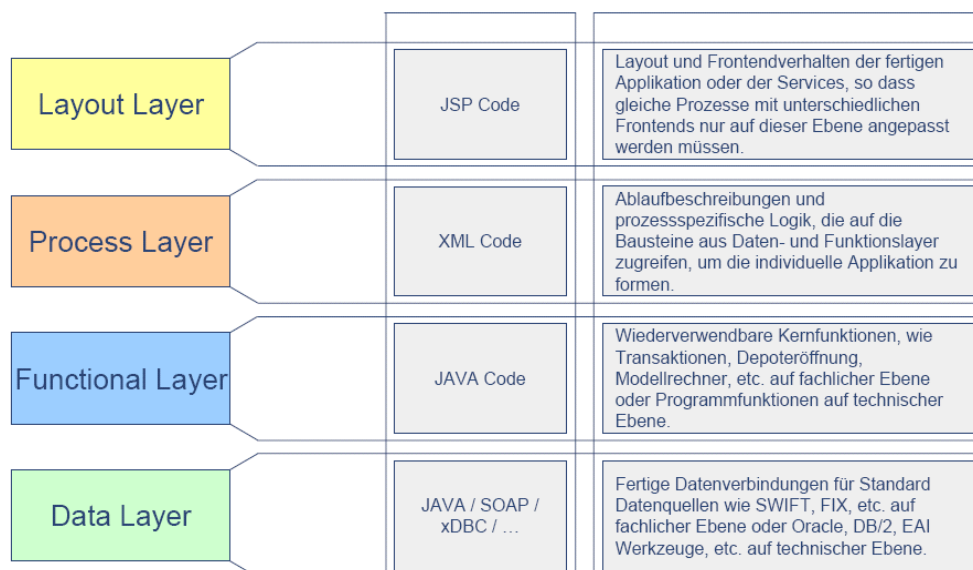
Betrachten wir nun die beiden weiteren Schichten „Darstellung“ und „Logik“ mit Blick auf die Wiederverwendbarkeit und damit unter dem Aspekt reduzierter Entwicklungszeit bzw. erweiterter Flexibilität. Zwischen diese beiden Schichten eine neue Stufe einzuziehen hat sich als Ansatz inzwischen bewährt. Diese neue Stufe nennt sich „Prozessschicht“.

Mit Einführung der Prozessschicht wird die Funktion der Darstellungsschicht um alle logischen Elemente bereinigt und die Funktionsschicht um alle die Elemente, die einzelne Funktionen zusammensetzen. Das Modell sieht dann über dem Datenhaushalt eine Funktionsschicht vor, die aus einzelnen Modulen besteht, welche laufend auf dem definierten Datenhaushalt ausgeführt werden. Diese Funktionen müssen unabhängig von Arbeitsabläufen sein. Aus Sicht der Vertriebsapplikationen sind solche Funktionen zum Beispiel verschiedene Bausteine zur Selektion oder Filterung des Produktbestandes, Bausteine zur Darstellung von Produkten, Bausteine um die Verbindung von Produktdaten zu redaktionellen Inhalten herzustellen, Simulationsbausteine und schließlich Bausteine, die eine Transaktion durchführen.

Diese Funktionsschicht ist die Schicht der grundsätzlichen Wiederverwendung. Sie beinhaltet nur Einzelfunktionen aus dem technischen Repertoire des Unternehmens und Einzelfunktionen aus dem Kerngeschäft des Unternehmens. Damit wird eine Mehrfachverwendung dieser Funktionsbausteine garantiert. Prozesse und Abläufe eines Unternehmens verändern sich laufend, während die fachlichen Kernfunktionen oft einen äußerst langen Lebenszyklus haben. Auch die technischen Kernfunktionen haben einen extrem langen Lebenszyklus, da sie niemals komplett abgeschafft oder verändert werden und sich nur in kleinen Schritten an die Unternehmensprozesse anpassen lassen. Derartige Funktionsblöcke lassen sich anhand der Fakten, die sich aus der Zusammenführung von IT-Infrastruktur und der Anforderungen an IT-Funktionen ergeben, leicht identifizieren.

Über der Funktionsschicht liegt die Prozessschicht. In dieser Schicht werden die einzelnen Funktionen zu geschäftlichen Abläufen zusammengeführt. Hier wird der Verkaufsprozess, eine neue Marketingkampagne oder die gesetzlich vorgeschriebene Änderung am Transaktionsprozess abgebildet. Diese Prozesse ändern sich relativ schnell, da sie von den gesetzlichen Rahmenbedingungen, von der Akzeptanz der Benutzer, von der Anforderung der Partner etc. abhängig sind. Die Auskoppelung dieser Schicht schafft deshalb eine wesentlich einfachere Zusammensetzbarkeit der Funktionen aus der Funktionsschicht. Ein solcher Entwicklungsgrundsatz nimmt auf diese Weise einen noch engen Bezug auf das Paradigma der Wiederverwendung.

Die Prozessschicht beinhaltet prozessspezifische Verbindungen zur Datenschicht und kann damit auch für einzelne Prozessschritte die erforderlichen Plausibilisierungen vornehmen oder die nur für einen bestimmten Prozess benötigten Datensets speichern. Auch in der Prozessschicht ist eine Wiederverwendung im Sinne des Aufrufs von Unterprozessen – wie man dies aus der betriebswirtschaftlichen Prozessmodellierung oder der Grundlehre der Informatik kennt – möglich. Mit dieser Schicht kann die technisch-fachliche Abstimmung erleichtert werden, da Prozesse übereinstimmend begriffen werden können und damit eine gute Schnittstelle bilden.



Die Ebenen des Vierschichtmodells

Aus Sicht der vertriebsunterstützenden Softwareentwicklung lassen sich auf der Ebene der Funktionsschicht aus den gleichen Funktionen immer wieder neue Prozesse für bestimmte Vertriebspartner, Kanäle, Kampagnen oder einfach nur neue Marketingaktionen zusammensetzen, ohne dass jede Funktion immer wieder neu erstellt werden muss. Durch die Kopplungsfähigkeit der Prozesse lassen sich leicht Verbindungen im Sinne des Cross- oder Upsellings zwischen

einmal implementierten Prozessen herstellen oder bereits vorhandene Prozesse als Untergruppen in neuen Methoden oder Komplettanwendungen aufrufen.

In der Prozessschicht liegt auch die Schnittstelle zur Service Oriented Architecture (SOA), denn in dieser Schicht werden die Prozesse definiert und die einzelnen Services oder ganze Servicegruppen abgebildet und implementiert, wenn sich diese zu einem Prozessablauf zusammenführen lassen.

Zu guter Letzt bleibt der Blick auf die Darstellungsschicht. Diese dient nun tatsächlich nur noch zur Abhandlung der Interaktion mit dem Benutzer. Der Benutzer kann dabei menschlich oder maschinell sein. Auf einen bestehenden Prozess kann ein neuer Mandant aufgesetzt werden, der ein verändertes Layout oder eine andere Sprache (menschliche Benutzer) nach außen gibt, oder der einfach eine andere Servicekommunikation umsetzt. Aus den Datenschnittstellen der Prozessschicht werden Daten in die Darstellungsschicht übertragen und in dieser abgebildet. Hier werden auch die Daten von den Benutzern entgegengenommen und als Eingangswerte für die einzelnen Prozessschritte aufbereitet.

Die Vorteile der vier Schichten zahlen sich schnell aus

Bei einer auf Basis dieses vierschichtigen Modells aufgesetzten Plattform ist eine maximale Wiederverwendung von Komponenten möglich. Erstellungskosten, Pflegeaufwand und die für die Erstellung benötigte Zeit werden dabei minimiert. Gleichzeitig bedeutet diese Minimierung eine Flexibilisierung der Entwicklung, da aus fachlicher Sicht einfache Änderungen auch technisch wesentlich schneller und sogar fehlerfreier umgesetzt werden können.



Maximale Wiederverwendung und -verwertung

Ein spezieller Transaktionsprozess – zum Beispiel für die Vertriebsaktion eines bestimmten Produktes – kann damit ganz leicht erzeugt werden. Beschränkt sich der eigentliche Transaktionsprozess z.B. nur auf die Eingabe, welches Produkt zu welchen Konditionen gebucht werden soll, handelt es sich lediglich um eine Änderung der Darstellungsschicht. Diese übergibt bereits vordefinierte Eingaben an den in der Prozessschicht befindlichen Transaktionsprozess. Mit dem Benutzer werden damit wesentlich weniger Interaktionen durchgeführt.

Ähnliches gilt für die Kombination von Kampagnenanwendungen oder für die Integration von einzelnen Anwendungsprozessen zu neuen Portalen. Hier werden nur vorhandene Prozesse oder Darstellungen zu neuen Gesamtanwendungen zusammengefügt.

Aber auch im Falle des Austauschs einzelner Funktionen, zum Beispiel auf Grund gesetzlicher Änderungen, müssen nur die Funktionsblöcke in der Funktionsschicht und eventuell deren Schnittstellen in der Prozessschicht angepasst werden – die Prozesse bleiben in ihrem Ablauf vollkommen unberührt.

Aus technischer Sicht lassen sich in den Prozessen und Funktionen auf einfache Weise technische Standards implementieren, die dann immer wieder verwendet werden – quasi automatisch. So müssen zum Beispiel generelle Sicherheitsmaßnahmen, wie der Schutz von Web-Services vor XSS-Attacken, nur einmal implementiert werden, um automatisch in alle Prozesse und Anwendungen eingebunden zu sein.

Ansprechpartner

Asswin Zabel
Leiter Marketing und Kommunikation

Am Niddatal 3
60488 Frankfurt am Main

Tel.: 0 69 / 4 05 68 - 1 05
Fax: 0 69 / 4 05 68 - 1 11
Mobil: 01 77 / 4 17 37 69
Email: zabel@arago.de