



Institut für komplexes
Datenmanagement AG

Eschersheimer Landstraße 526-532
60433 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 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 eingestuften aber gleichwohl erfolgskritischen IT-Systeme rund um den Vertrieb stehen damit unter besonderem Erfolgs- und Zeitdruck. Probleme bei deren Implementierung und Inbetriebnahme werden sofort zu Problemen im Kerngeschäft.

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 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.

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 zwischen der Darstellungsebene, die der Benutzer zu sehen bekommt, der Programmlogik, die alle Verarbeitungsmechanismen vereint, und dem Datenhaushalt, bzw. dem Zugriff aller notwendigen Daten vorgeschrieben. Diese Trennung wurde unter folgenden Kerngesichtspunkten vorgenommen:

1. Die Darstellung muss unabhängig sein, weil mehrere Oberflächen (z.B. 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 Algorithmmik für fehlerfreie sowie performante Programme und bezüglich einer hohen Wiederverwendbarkeit. Diese beiden Grundinteressen der Informatik sind mit einer klaren Schnittstelle zur Außenwelt - nämlich der Darstellung und der Datenversorgung - leichter umzusetzen.

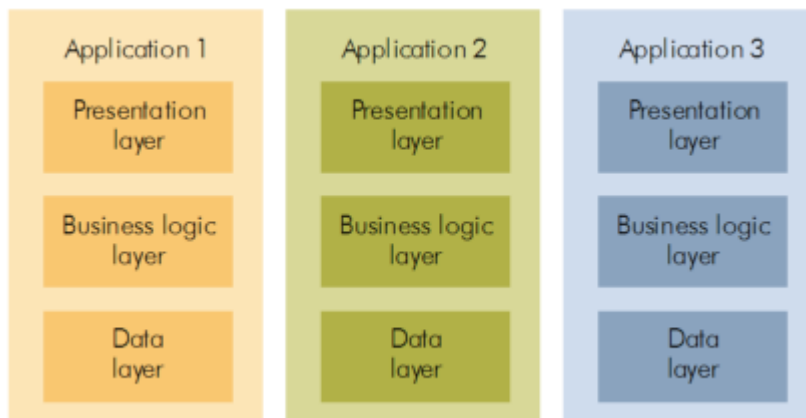


Abbildung: Die Ebenen des Dreischichtmodells

Das dreischichtige Modell hat sich als sehr effektiv erwiesen und leistet auch heute noch gute Dienste. Es vermeidet das blinde „Drauflos 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 dies aktuell der Fall ist.

Die Technologie bleibt nicht stehen

Seit der Einführung des Dreischichtmodells haben sich die technische Umgebung und die 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. Dass inzwischen fast allen Applikationen eine verteilte Infrastruktur zu Grunde liegt, 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 miteinander in Konkurrenz um Ressourcen treten oder sich sogar gegenseitig behindern, bzw. sogar blockieren. 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 Gegensatz 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 kann helfen, die fachlichen und technischen Anforderungen zu bedienen.

Die Anforderungen definieren das Modell

Vor einer Veränderung des dreischichtigen Modells ist im ersten Schritt eine Bestandsaufnahme 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. Aufgrund der Marktnähe ist die vertriebsunterstützende IT ein Vorreiter bei der Definition der neuen Anforderungswelten. Der Übersichtskasten zeigt mögliche Anforderungen dieser Gruppen.

Diesen Anforderungen aus unterschiedlichen Perspektiven sind Wiederverwendung und Flexibilität als die beiden wichtigsten Ziele gemeinsam. Beides ist erforderlich: Sowohl aus Sicht des Kunden, der von Time-to-Market und von Kostenüberlegungen getrieben wird, als auch aus Sicht der Lieferanten, denen die Herstellungszeiten am Herzen liegen. Die Anforderungen sind auch aus technischer Sicht gültig, nach der ohnehin keine Funktion doppelt programmiert werden darf.

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

Das Vierschichtmodell

Mit den Kriterien „Wiederverwendung“, „schnelle Entwicklung“ und „Flexibilität“ vor Augen, kann nun das klassische Dreischichtmodell unter die Lupe genommen werden.

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 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, aufzuspalten. Bei weniger breiten Datenhaushalten ist - unabhängig von der physikalischen Datenmenge - diese Aufspaltung zwar technisch gesehen sehr schön, aber 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 und die Funktionsschicht um alle die Elemente, die die einzelnen Funktionen zusammensetzen, bereinigt. Das Modell sieht also eine Funktionsschicht vor, die aus einzelnen Modulen besteht. Diese Funktionen müssen unabhängig von Arbeitsabläufen sein. Aus Sicht der Vertriebsapplikationen sind solche Funktionen zum Beispiel verschiedene Bausteine zur Selektion oder Filterung des Produktuniversums, 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 und 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 sich nur in kleinen Schritten an die Unternehmensprozesse anpassen lassen. Derartige Funktionsblöcke lassen sich anhand der Zusammenführung von IT-Infrastruktur und fachlichen Anforderungen an IT-Funktionen leicht identifizieren.

Über der Funktionsschicht liegt die Prozessschicht. In dieser Schicht werden die einzelnen Funktionen zu geschäftlichen Abläufen zusammengeführt. Hier wird zum Beispiel der Verkaufsprozess, eine neue Marketingkampagne oder die gesetzlich vorgeschriebene Änderung am Transaktionsprozess abgebildet. Diese Prozesse ändern sich relativ schnell, da sie von der Akzeptanz der Benutzer, von der Anforderung der Partner, von den gesetzlichen Rahmenbedingungen etc. abhängig sind. Mit der Auskoppelung dieser Schicht können fachliche Änderungen und neue Prozesse leicht und ohne Neuentwicklung von Funktionen aus den Modulen der Funktionsschichten zusammengesetzt werden.

Dieser Entwicklungsgrundsatz fördert also die Wiederverwendung der Funktionen und beschleunigt gleichzeitig die Bereitstellung neuer fachlich relevanter Prozesse.

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

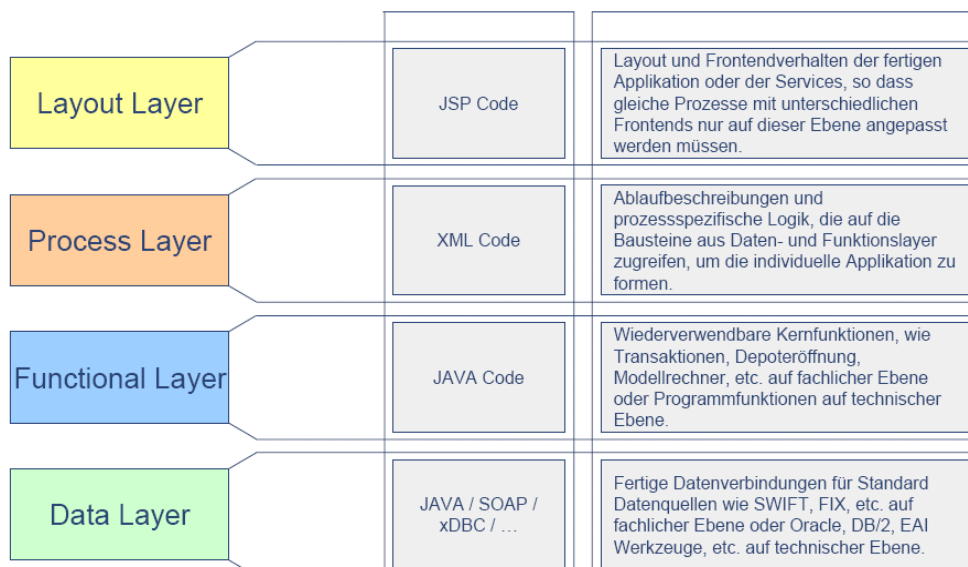


Abbildung: Die Ebenen des Vierschichtmodells

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.

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 (maschineller Benutzer) umsetzt. Aus den Datenschnittstellen der Prozessschicht werden Daten in die Darstellungsschicht übertragen und in dieser abgebildet.

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 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.

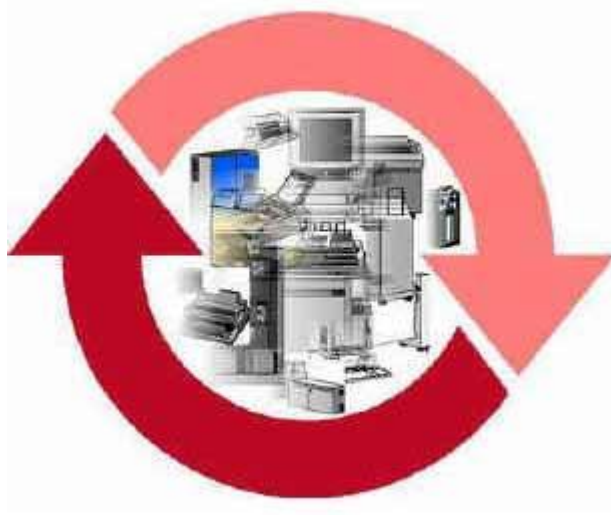


Abbildung: Maximale Wiederverwendung und -verwertung

Ein spezieller Transaktionsprozess – zum Beispiel für die Vertriebsaktion eines bestimmten Produktes – kann damit ganz leicht erzeugt werden. Beschränkt sich der eigentliche Transaktionsprozess nur auf die Eingabe, welches Produkt zu welchen Konditionen gebucht werden soll, fallen nur Änderungen in der Darstellungsschicht an. Diese übergibt bereits vordefinierte Eingaben an den in der Prozessschicht befindlichen Transaktionsprozess. Ähnliches gilt für die Kombination von Kampagnenanwendungen oder für die Integration von einzelnen Anwendungsprozessen zu neuen Portalen. Dabei 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

Martin Friedrich

Vorstand Marketing und Vertrieb

Institut für komplexes Datenmanagement AG

Eschersheimer Landstraße 526-532

60433 Frankfurt am Main

Tel.: 069 40 56 8 - 210

Fax: 069 40 56 8 - 111

E-Mail: friedrich@arago.de

URL: <http://www.arago.de>