

NEWS

HBS Faktura. Eine neue Auftragsbearbeitung von HBS Personal Computer. Testen Sie die Anwendung. Eine Demo-Version können Sie unter www.hbspc.ch/Faktura.html downloaden. Es ist keine Registrierung notwendig.

Application Management

Wollen Sie Ihre bestehenden Anwendungen erweitern? Planen Sie ein Refactoring einer bestehender Applikation oder brauchen Sie Unterstützung bei der Software Architektur? Wir helfen und unterstützen Sie im gesamten Application Management Prozess

HBS Personal Computer
Parkstrasse 16a
5012 Schönenwerd
Telefon: 062 295 66 66
Email: mail@hbs-web.ch



Diese Ausgabe
HBS .NET Framework P.1

N-Tier Applikationen mit dem HBS .NET Framework

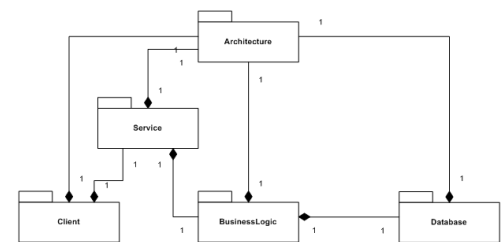
Ziel des HBS Frameworks ist es, dem Entwickler bei der Umsetzung von Client/Server Applikationen einerseits eine Basis bereitzustellen, mit welcher der Entwickler schnell verschiedenen Layers umsetzen kann. Andererseits dient das Framework als Design Guideline und stellt Interfaces und Patterns (GOF) zur Verfügung, sodass der Entwickler gezwungen wird, die vorgegebenen architektonischen Richtlinien einzuhalten damit Architektur Violations reduziert werden können.

Das HBS Framework unterstützt den Entwickler mit 5 Modulen:

1. Die Architektur, welche als zentrales Element von allen anderen Layern verwendet werden kann
2. Die Backend Anbindung für das Anbinden von SQL oder Access Datenbanken, oder aber die Anbindung von NHibernate
3. Den Business Logic Layer für die Aufbereitung der Daten für den Client
4. Das Webservice Modul, für die Anbindung von Webservices. Hier ist zu erwähnen, dass die Webservice im HBS Framework als reine Delegates fungieren.
5. Das Client Modul für das schnelle Umsetzen von Formularen und Erweiterungen/Hilfsmethoden zu Infragistics. Dieser Layer unterliegt dem MVC Konstrukt

Client Layer

Ziel des Client Moduls ist es, in Formularen oder anderen GUI Komponenten eine Business Logic Methode aufzurufen, welche synchron oder asynchron ablaufen kann. Dabei darf es keine Rolle spielen, ob die Business Logic lokal oder Remote (wie dies in einer richtigen Client Server Anwendung der Fall ist) vorhanden ist. Nach einem Call müssen die beteiligten GUI Komponenten Daten erhalten, welche sie dann entsprechend darstellen können. Die Kommunikation soll weiter auch zwischen den einzelnen Komponenten stattfinden können, sodass Komponente X mit einer Komponente Y kommunizieren kann. Die Kommunikation zwischen den einzelnen Komponenten soll dabei transparent sein, die Kopplung muss lose sein, sodass eine Verwendbarkeit von Komponenten einfach umzusetzen ist. Die GUI Komponenten dürfen keine Business Logic enthalten. Der Zugriff auf Daten muss einfach umzusetzen sein, wobei in den einzelnen Panels zwischen Remote und Lokale Architektur nicht unterschieden werden darf.



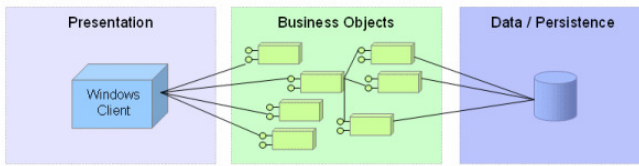
Module HBS Framework

Umsetzung Client Modul

Zur Umsetzung wurden mehrere Patterns von u.a. GOF berücksichtigt und mit einbezogen. Eine erste Entkopplung von GUI Komponenten erfolgt durch das MVC Pattern. Jedes Formular oder Controller ist ein Listener, sodass diese an einer zentralen Stelle, im Client Model, benachrichtigt werden können. Diese Listener sind somit im Model registriert. Das Client Model selbst ist ebenfalls ein Listener gegenüber der Proxy Klassen, sodass das Model über Beginn und Ende eines Business Logic Services informiert werden kann. So wird das Model von den Proxies informiert, falls eine Methode aufgerufen wird. Eine weitere Benachrichtigung erfolgt, falls die Methode abgearbeitet wurde. Diese Notifizierung erfolgt, indem das Resultat der abgearbeiteten Methode in einem Objekt an das Model zurückgesendet wird. Diese Notifizierung von den Proxies an das Client Model hat 2 Aufgaben:

1. Die Benachrichtigung an einen StatusListener. Dieser StatusListener kann somit einem Benutzer gerade bearbeitende Methoden (lokal oder auf einem Server darstellen). So wird bei Beginn einer Methode eine Statusbar aktiviert, beim Ende der Methode, die Statusbar wieder deaktiviert.
2. Eine Notifizierung der Proxies an das Model wird dazu benötigt, dass alle beim Model registrierten GUI Komponenten benachrichtigt werden können. Bei dieser Benachrichtigung wird das Resultat der Business Logic mitgegeben, sodass die entsprechenden Formulare oder Controller das Resultat entsprechend darstellen können.

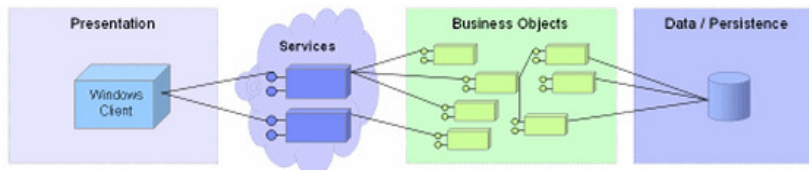
Anbindung Client ohne Webservice



Serviceorientierte Applikationsarchitektur ohne Webservice

Die Darstellung zeigt, die einzelnen Layers schematisch dargestellt. Die Präsentationsschicht beinhaltet alle Formulare, Controls, Panels, ClientModels, ClientServiceLocator und die Proxy-Klassen für den Client. Die Business Logic Schicht beinhaltet sämtliche Geschäftslogik, inklusive Aufbereitung der Transport Objekten (VO's) aus den Daten der DB. Die Persistenzschicht stellt die Schnittstelle zum bspw. SQL Server dar

Anbindung Client mit Webservice

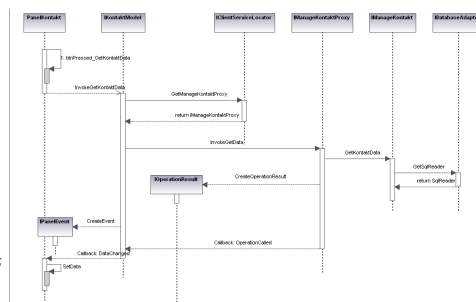


Serviceorientierte Applikationsarchitektur mit Webservice

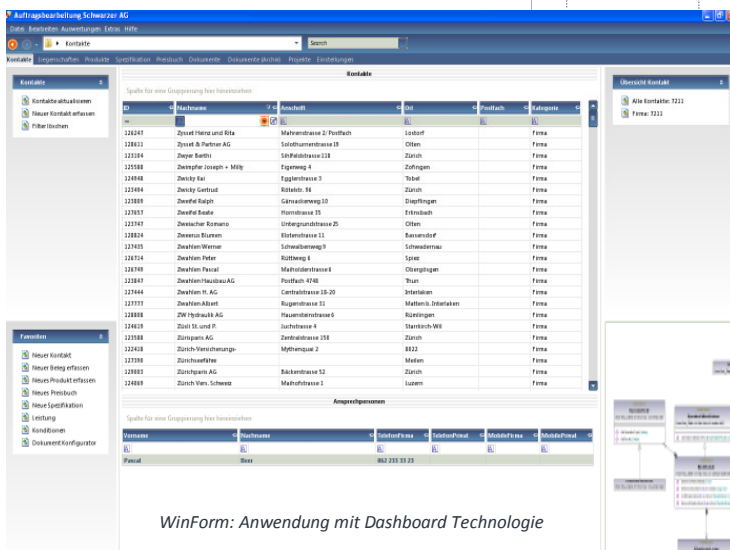
Die Darstellung zeigt, dass zwischen der Präsentation und der BL die Webservices dazwischen liegen. Die Services werden in einem Proxy-Layer gekapselt. Für die Präsentation Layer ist nicht ersichtlich, ob die BL durch die Services aufgerufen werden, oder diese direkt mit der BL kommunizieren. Die Entscheidung für die Delegation erfolgt in der Proxy Anbindung und ist auch dort gekapselt.

Integration HBS Framework in eine Applikation

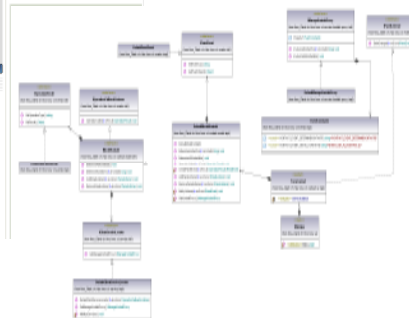
Die Integration des HBS Frameworks lässt sich in jede Anwendung einbinden, sei dies in eine N-Tier Architektur, oder aber auch in einen Fat Client. Der Entwickler kann in jedem Bereich vom HBS Framework profitieren. Durch die gegebenen Interfaces und Basisklassen werden dem Entwickler die einzelnen Layers vorgegeben, Architektonische Violations werden dadurch stark reduziert.



MSC: Asynchroner Service Call des Clients



WinForm: Anwendung mit Dashboard Technologie



Klassendiagramm: Client Modul (org. Package)

Das ultimative Toolset für die schnelle Erstellung der Präsentationsebene für Windows Forms, ASP.NET, WPS oder Silverlight. NetAdvantage for .NET bietet fortgeschrittene Tools zur Entwicklung der Präsentationsebene wie Grids, Charts, Toolbars, Menues, Listbars, Trees, Tabs, Explorer-Bars, Editors und viele mehr. Durch diese UI-Elemente können Sie Ihrer .NET Anwendung einfach den neuesten Microsoft Look&Feel inklusive Office 2007 geben.

Quelle: www.infragistics.com

NHibernate

Hibernate ist ein Open-Source-Persistenz- und ORM-Framework für Java. Für .NET ist eine portierte Version namens NHibernate verfügbar. Hibernates Hauptaufgabe ist das Object-Relational Mapping (O-R-Mapping, kurz ORM). So ermöglicht es, gewöhnliche Objekte mit Attributen und Methoden (im Java-Bereich POJOs genannt) in relationalen Datenbanken zu speichern und aus entsprechenden Datensätzen wiederum Objekte zu erzeugen. Beziehungen zwischen Objekten werden auf entsprechende Datenbank-Relationen abgebildet.

Quelle: www.wikipedia.org