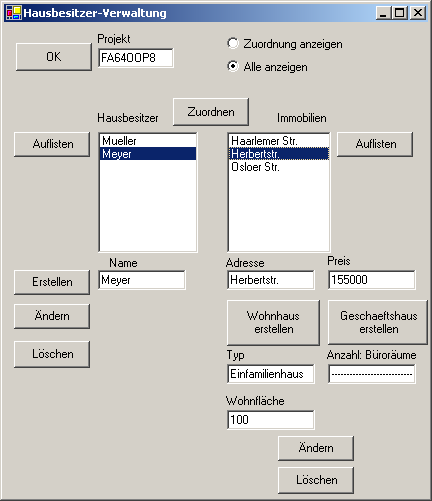
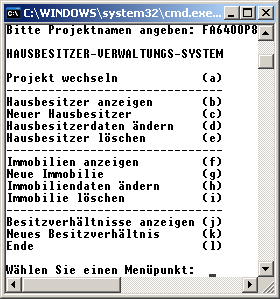
**Software-Projekt**

# Zielsetzungen

Es soll ein Software-System als 3-Schichten-Architektur entwickelt werden, welches auf einer Datenhaltung aufsetzt und ein beliebiges Anwendungsproblem (als Prototyp) löst.



Benutzungsoberfläche

##### <<interface>> IFachkonzept

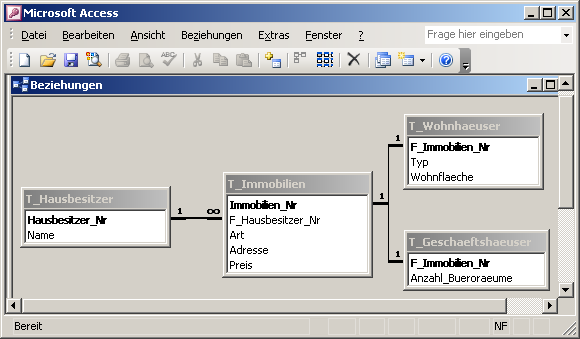
**Fachkonzept-Schicht**

##### Fachkonzept 2

##### Fachkonzept 1

##### <<interface>> IDatenhaltung

**Datenhaltungs-Schicht**



<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<Immobilien>

<Immobilie Art=“Wohnhaus“>

<Adresse>Herbertstr.</Adresse>

<Preis>155000</Preis>

<Typ>Einfamilienhaus</Typ>

<Wohnflaeche>100</Wohnflaeche>

<Besitzer>Meyer</Besitzer>

</Immobilie>

</Immobilien>

\*Quellen: http://www.stefan-baur.de/downloads/Lastenheft.pdf, geprüft 27.08.2012

http://de.wikipedia.org/wiki/Lastenheft, geprüft 27.08.2012

# Produkteinsatz

Die einzelnen Anwendungen sollen von qualifiziertem Personal bedient werden können.

# Produktübersicht, Produktfunktionen und Produktleistungen

## **Die Benutzungsoberfläche**

Es sollen zwei verschiedene Benutzungsoberflächen (GUI und TUI) entwickelt werden, die einen vergleichbaren Funktionsumfang zur Verfügung stellen und jeweils eine Referenz auf die Fachkonzeptschicht besitzen.

## Die Fachkonzeptschicht

Es reicht eine Fachkonzeptschicht. Diese sollte aber eine Klassenimplementierung enthalten, die sich an einem Entwurfsmuster orientiert (s. u.).

## Die Datenhaltung

Es sollen zwei unterschiedliche Datenhaltungen vorgesehen werden, eine Pflichtdatenbank (relationale Datenbank) und eine Wahldatenhaltung (XML-Datei, Textdatei etc.). Das Anwendungs-problem muss mindestens eine Assoziation (1:n oder m:n) enthalten, so dass die relationale Datenbank mindestens aus 2 Tabellen besteht. Verkürzer benötigen nur eine Datenhaltung.

## Das Design-Pattern

Das Design-Pattern orientiert sich am ausgewählten Anwendungsproblem und wird in der Regel ein erzeugendes Muster sein, z. B. Builder, Prototyp, Abstract Factory oder Factory Method. Singleton ist zu einfach.

## „Auswechseln“ der Schichten

In der Main-Methode könnte über die Zusammensetzung der Schichten entschieden werden, z. B. durch:

**GUI eineGUI = new GUI(new Fachkonzept(new Datenbank()));**

oder

**TUI eineTUI = new TUI(new Fachkonzept(new XML()));**

# Qualitätsanforderungen, Abnahmeanforderungen und Testdaten

Am „Abnahmetag“ (im vorletzten Turnus) soll demonstriert werden, dass alle 4 Möglichkeiten lauffähig sind. Lese- und Schreibzugriffe müssen funktionieren, das verwendete Design-Pattern erläutert und im Quelltext nachgewiesen werden. Testdaten werden in Absprache mit dem Auftraggeber ([ehlert@oszimt.de](mailto:ehlert@oszimt.de)) eingegeben. An diesem Tag wird auch die „Dokumentation“ abgegeben. Hinweise zu dieser Dokumentation (gleichzeitig zweite Klassenarbeit) erfolgen im Laufe des Halbjahres.

# Ergänzungen bzw. Hinweise

* Es werden auf freiwilliger Basis Dreiergruppen gebildet.
* Jede Gruppe bekommt zwei Referatsthemen, Verkürzergruppen nur ein Thema.
* Ein funktionierender Prototyp ist besser als ein ambitionierter „Rohrkrepierer“!
* Programmiersprachen können beliebig gewählt werden.
* Ein (kurzes!) „Pflichtenheft“ muss die Team- bzw. Arbeitsaufteilung, die Gruppennr., eine GUI- und Konsolen-Skizze, mögliche Testszenarien und Meilensteine beinhalten und die Problemstellung und die geplanten Datenhaltungen nennen. Ein ER-Modell weist die gewünschte Assoziation nach. Das „Pflichtenheft“ ist am Ende des zweiten Turnus unaufgefordert dem Auftraggeber (Lehrer) zu zeigen.
* Es gibt eine Teamzensur. Die Teams bestimmen dann die individuellen Zensuren,

dabei hat der Teamsprecher das „letzte Wort“. Gelingt nicht eine einvernehmliche

Lösung wird der Auftraggeber als Mediator hinzugezogen.