Technische Hochschule Ingolstadt

Lösung Probeklausur Architektur und Entwurfsmuster

2017

Ausgangssituation

Als Mitarbeiter eines Projektteams werden Sie mit der Entwicklung eines Systems für die Administration einer Hochschule betraut.

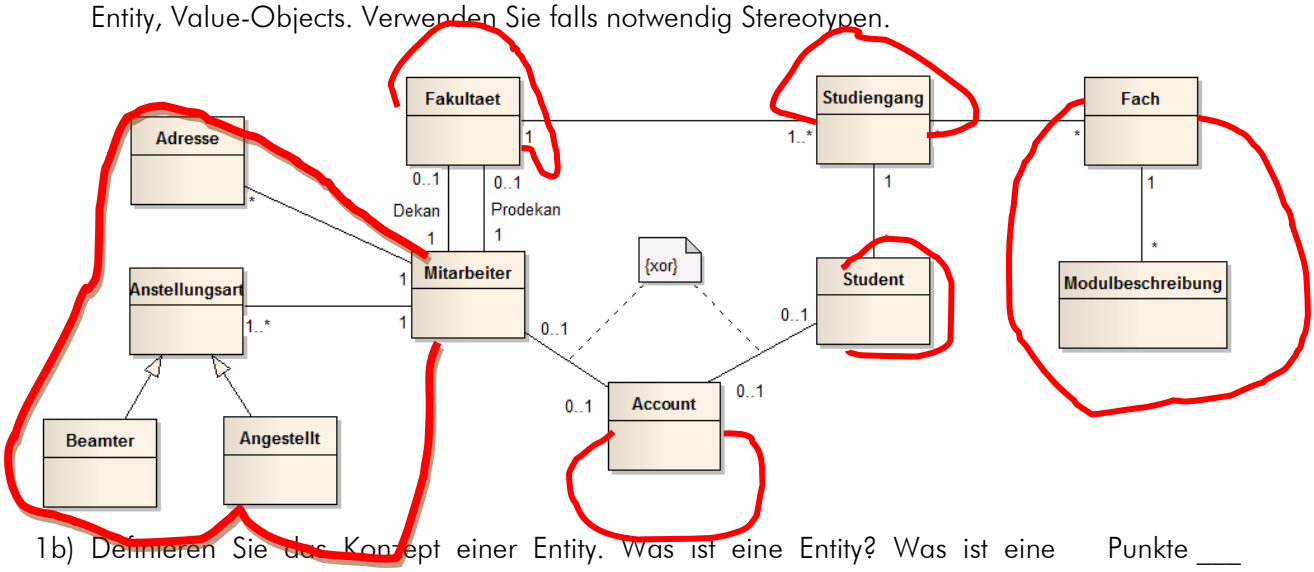
Produktvision:

- Das System soll in der Lage sein, eine Hochschule mit beliebig vielen Fakultäten und Abteilungen zu verwalten
- Über das System soll Studenten die Möglichkeit geboten werden, möglichst viel über eine moderne Online-Plattform zu erledigen. Hierunter fällt z.B. die elektronische Prüfungsanmeldung und die Noteneinsicht
- Das System soll einen hohen Sicherheitsstandard erfüllen. Jeder Benutzer, der mit dem System arbeitet, unterliegt bestimmten Restriktionen, welche durch ein Berechtigungssystem definiert werden
- Damit ein Benutzer eine Funktion ausführen kann, muss er sich am System anmelden

 Erzielte Punkte:_____

1. Domain-driven Design

1a) Im Rahmen des Designs wenden Sie nun die Methode „Domain-driven Design" auf das nachstehende Produktmodell an. Analysieren Sie das nachfolgende Diagramm und kennzeichnen Sie alle Klassen in diesem Diagramm. Stellen Sie folgende Kernelemente innerhalb des Diagramms heraus: Aggregate, Entity, Root-Entity, Value-Objects. Verwenden Sie falls notwendig Stereotypen. Punkte ___



1b) Definieren Sie das Konzept einer Entity. Was ist eine Entity? Was ist eine globale bzw. lokale Identität? Geben sie jeweils ein Beispiel für die Identität an. Punkte ___

- Ein Objekt welches nicht ausschließlich durch seine Attribute beschrieben wird sondern primär durch seine Identität wird Entity genannt
- locale Identität welche nur innerhalb des Aggregats eindeutig sein muss (Position einer Rechnungsposition)
- Globale Identiät muss global eindeutig sein. (Die rechnungsnr selbst)

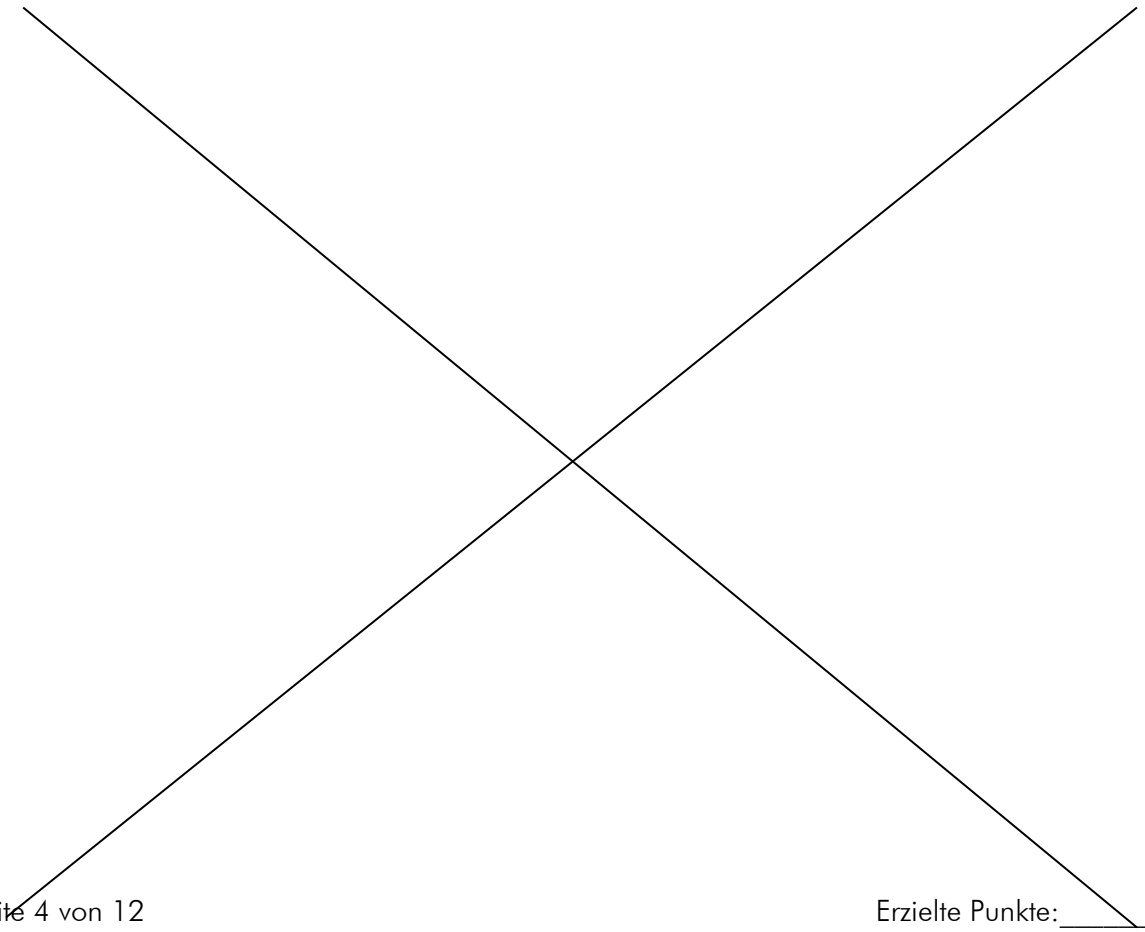
1c) Welche Besonderheiten sind bei der Implementierung von Assoziationen zwischen Aggregaten zu beachten? Gilt diese Einschränkung auch innerhalb eines Aggregates? Begründen Sie Ihre Aussage(n). Punkte ___

- Lose Kopplung zwischen Aggregaten da unerschiedliche Lebenzyklen der Objekte
- Innerhalb eines aggregates können direkte Objetkreferenzen benutzt werden da alle Objekte immer „gleichzeitig" existieren.
- Assoziationen zu anderen Aggregaten sollten immer auf die Root–Entity gehen

Erzielte Punkte:_____

1d) Viele Applikationen müssen auf Altsysteme zugreifen. Definieren Sie das Problem sowie die Lösung, welche durch eine „Anti-Corruption-Layer“ vorgeschlagen wird. Begründen Sie warum diese Lösung sinnvoll ist und warum Sie zu einer flexibleren Software führt. Punkte ___

- Definieren Sie eine „Isolation Layer“ welche dem Client externe Funktionalität innerhalb des eigenen Domänenmodells zur Verfügung stellt
- Die Layer kommuniziert mit den externen Systemen durch die existierenden Schnittstellen ohne bzw. mit geringer Modfikation.
- Intern ist die Isolation Layer ein Übersetzer in beide Richtungen
- Applikation bleibt unabhängig von den angebunden Systemen.
- Kann später durch neue Implementierungen ersetzt werden



2. Software Architektur & Design Prinzipien

2a) Erläutern Sie die Begriffe „Inversion of Control“ und „Dependency Injection“. Punkte __

- Inversion of Control: Die Komponenten wissen nichts über Ihre Umgebung. Die Umgebung steuert die Komponenten und deren Beziehungen zueinander

- Die Abhängigkeit zu den importierten Schnittstellen wird durch den Container(=Laufzeitumgebung) verwaltet.

- Wird eine Komponente erzeugt werden die Abhängigkeit von außen durch den Container injiziert.

2b) Erläutern Sie die Arbeitsweise eines Service-Locators. Welche zwei Realisierungs- varianten existieren? Wie beurteilen Sie den Service-Locator in Bezug auf die entstehende Kopplung? Punkte __

- Service-Locater ist ein Dictionary welches die Verweise auf die Implementierten Komponenten verwaltet.
- Ein Client frägt den Service-Locator nach der gewünschten Komponente an
- Dynamisch: Konfiguration per XML
- Statisch: Die Konfiguration des service-Locators wird im Quellcode festgelegt.
- Kopplung ist hoch, das das gesamte System Abhängigkeiten zu der Implementierung des Service-Locators hat.

2c) Erläutern Sie, warum man mit Hilfe der „Kapselung“ die Einhaltung von sog. „Invarianten“ leichter sicherstellen kann. Gibt es Sonderfälle die es erlauben das Prinzip Kapselung aufzuweichen? Was ist Ihr Rat als Architekt? Punkte __

- Kapselung sagt aus dass der Zugriff auf Variablen nur über Methoden erfolgt.
- Dadurch kann das Objekt immer selbst entscheiden was erlaubt ist und was nicht.
- D.h. durch die Kapselung kann das Objekt sicherstellen dass die Invarianten eingehalten werden.
- Einziges Argument gegen Kapselung: Performance durch sehr häufigen Zugriff auf die Attribute

2d) Definieren Sie das liskovsche Substitutionsprinzip. Skizzieren Sie ein Klassendiagramm und erläutern Sie daran was passiert, wenn gegen das Prinzip verstoßen wird. Begründen Sie, warum dieses Prinzip notwendig für einen sinnvollen Einsatz der Polymorphie ist. Punkte __

- Alle beweisbaren Eigenschaften der Oberklasse müssen auch für alle Unterklassen gelten.
- Wird gegen das Prinzip verstoßen ist die polymorphe Verwendung problematisch weil sich Unterklassen anders verhalten als die Oberklassen. Dadurch kann es zu instabilen Programmsystemen kommen.
- Notwendig: Durch die Einhaltung des Prinzips kann sich ein Client darauf verlassen dass jedes abgeleitete Objekt genauso benutzt werden kann wie die Oberklasse selber

2e) Definieren Sie das Interface-Segregation-Prinzip. Begründen Sie, warum der Einsatz dieses Prinzips eine leichtere Wartbarkeit, bzw. Erweiterbarkeit der Software garantiert. Punkte __

- Interface-Segregation-Prinzip: Ein Aufrufer sollte nur von Schnittstellen abhängig sein der er auch tatsächlich benötigt
  - Aufteilung eines großen Interfaces in kleinere Teile
  - Dadurch wird die Abhängigkeit von unnötigen Schnittstellen reduziert
  - Der Aufwand für die Implementierung einer Schnittstelle wird ebenfalls reduziert
  - Vorteil: Liefert für Interfaces eine Operationalisierung des Prinzips der hohen Kohäsion.
- Leichtere Wartbarkeit, Erweiterbarkeit: Änderungen die einen Aufrufer negativ betreffen werden reduziert. Dadurch können Erweiterungen leichter durchgeführt werden weil die Auswirkung der Änderung lokal begrenzt ist.

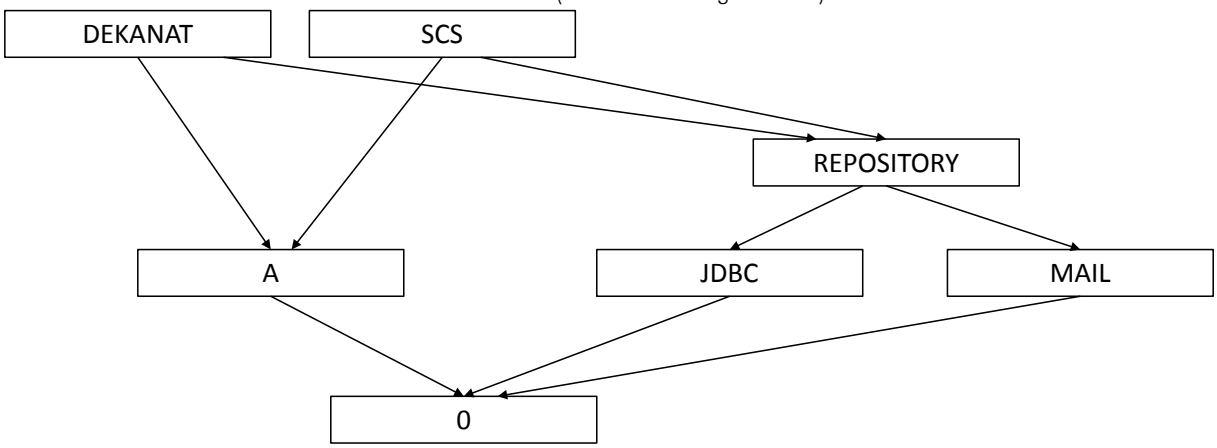
3.) Software Architektur – Die Struktursicht

Im Rahmen einer Architekturbewertung sollen Sie nun Komponenten, sowie deren Schnittstellen bewerten. Hierzu wird ihnen für das Repository des Hochschulsystems folgendes Quellcodefragment vorgelegt (Schnittstellenspezifikation):

```
public interface TechnicalLayer {
 // Studiengang laden
 Map<String,String> loadStudiengang( jdbc.Connection conn, String ident );
 // Studiengang in der Datenbank speichern
 void saveStudiengang(jdbc.Connection conn, Studiengang studiengang );
 // Mail versenden
 void sendMail( java.mail.Adress recipient, String betreff, String body );
}
```

3a) Skizzieren Sie den Software-Kategoriegraphen unter der Annahme, dass die Anwendungskomponente „ServiceCenterStudium“ (SCS) sowie die Anwendungskomponente „Dekanat“ auf das Repository zugreifen. Darüber hinaus gelten folgende Annahmen: Punkte ___

- Alle Datenbankspezifischen Klassen/Interfaces (jdbc) liegen in der Kategorie „JDBC“.
- Die Klassen des Produktmodells werden in der Kategorie „A“ (Anwendung) angesiedelt
- Collections liegen in Kategorie 0
- Das SCS kann automatisch Emails versenden (Email-API: Kategorie Mail)

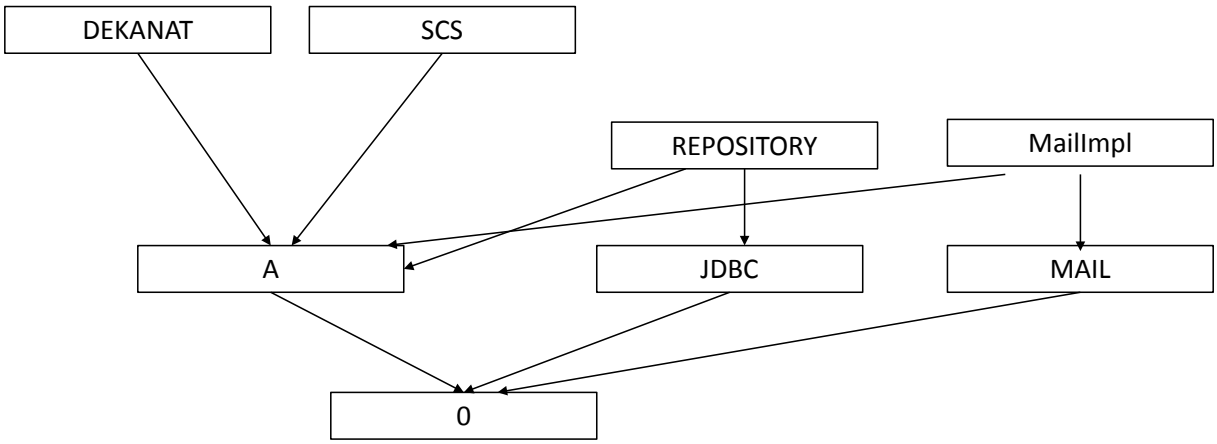


3b) Begründen Sie, warum man gemäß Quasar A und T Software immer trennen sollte. Wie löst Quasar dieses Problem, welche Art von Softwarekategorie wird hierzu verwendet und welche Einschränkung muss berücksichtigt werden? Punkte ___

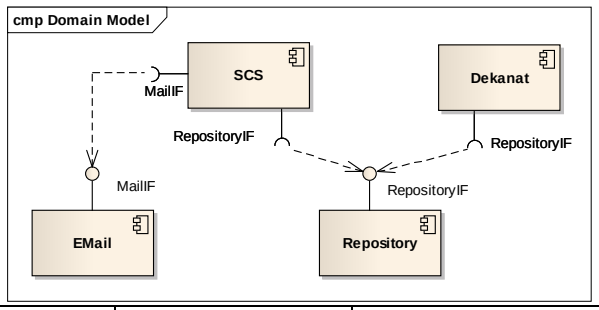
- Die technische Basis heutiger System verändert sich permanent/schnell. Deshalb ist die Trennung wichtig um Flexibilität und Erweiterbarkeit einfach zu gestalten

- R-Software: Representational Software. Reine Konverter-Software welche aus der Welt der Domänen in die Technologie übersetzt. Keine Logik in dem Konverter

3c) Zeichnen Sie nun einen verbesserten Kategoriegraphen. Zerlegen Sie das Interface „TechnicalLayer“ in verschiedene Interfaces, um das Interface-Segregation-Prinzip zu gewährleisten. Punkte ___



3d) Ordnen Sie die Artefakte des nebenstehenden Diagramms sinnvoll den Softwarekategorien aus Aufgabe 3d zu. Punkte ___



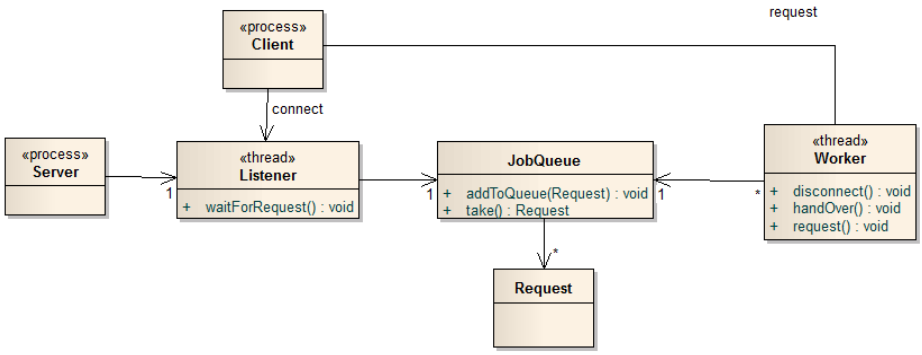
Artefakt	Typ	Kategorie
Implementierung SCS	Klasse(n)	SCS
Implementierung Dekanat	Klasse(n)	DEKANAT
Implementierung Email	Klasse(n)	MAILIMP
MailIF	Schnittstelle	A
RepositoryIF	Schnittstelle	A

3e) Die Kopplung eines Systems kann durch den Entwurf der Schnittstellen einer Komponente stark beeinflusst werden. Nennen Sie drei Möglichkeiten wie die Kopplung beim Schnittstellenentwurf beeinflusst wird. Geben Sie für jede Variante eine kurze Beschreibung sowie den Grad der Kopplung (Hoch, Mittel, Niedrig) an. Punkte ___

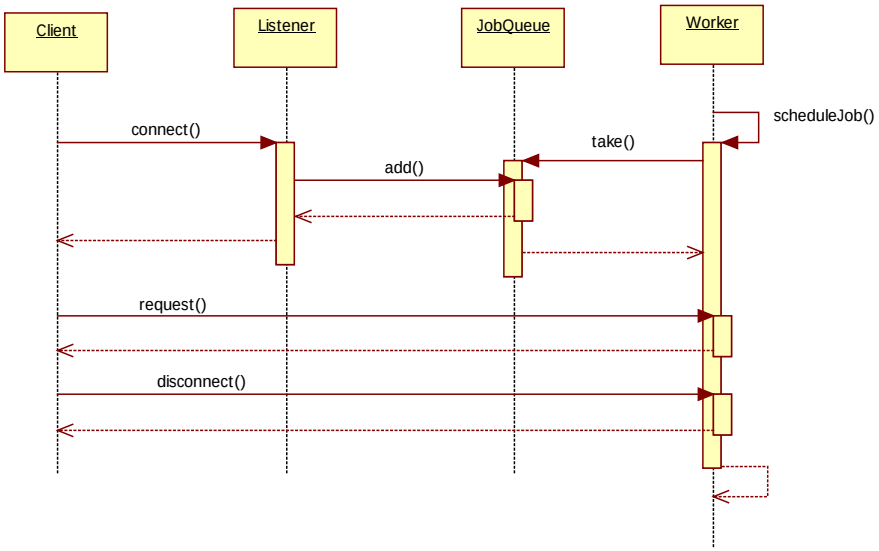
Variante	Beschreibung	Grad
Objekt	Direkte Übergabe der Domänenobjekte	Hoch
Schnittstelle	Verstecken der Objekte hinter Entitätsschnittstellen	Mittel
Transferobjekt	Erzeugen von Kopien der Entities und übergabe der TA-Objekte	Niedrig

 Erzielte Punkte:_____

4.) Die Prozesssicht



4a) Erweitern Sie das nachfolgende Sequenzdiagramm so, dass der prinzipielle Ablauf des Job-Queue-Patterns ersichtlich wird. Erläutern Sie die Arbeitsweise des Patterns anhand des Sequenzdiagramms. Punkte ___



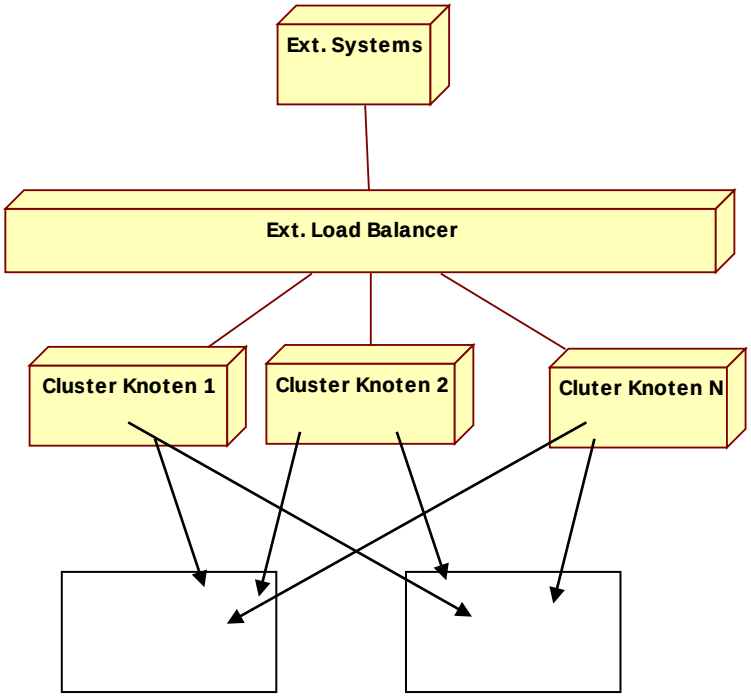
Eingehende Anfragen werden in eine JobQueue eingestellt
Die Worker holen wiederholte eingehende Requests aus der Job-Queue und arbeiten diese ab. Hat ein Worker den Request verarbeitete beginnt der Vorgang von vorne. Es wird ein neuer Job aus der JobQueue geholt

4b) Wie verhält sich dieses Pattern, wenn die Queue voll ist und keine freien Worker zur Verfügung stehen? Was kann man tun, um eine Überlast des Systems zu verhindern? Punkte ___

- Keine Freien Worker: Eingehende Requests müssen warten bis Ressourcen zur Verfügung stellen
- Wenn die Queue über Gebühr wächst dann kann der Hauptspeicher überlaufen
- Idee: Blockieren eingehender Requests bis der Füllstand der Queue wieder unter einen Maximallevel fällt
- Oder: Queue persistieren

5) Die physische Sicht

5a) Skizzieren Sie den physischen Aufbau eines active/active-Clusters (symmetrischer Cluster). Nehmen Sie in die Skizze folgende Komponenten mit auf: External Client, Loadbalancer, Application Server und Datenbank-Server. Zeichnen Sie auch redundante Komponenten ein. Punkte ___



5b) Beschreiben Sie die Funktionsweise eines active/active-Clusters. Punkte ___

Eingehende Anfrage werden durch einen Load-Balancer auf die verschiedenen Clusterknoten verteilt
Fällt eine Clusterknoten aus, übernehmen die restlichen Knoten die Arbeit für den ausgefallenen Knoten mit
Hierzu ist ein Cluster-Protokoll notwendig welches die Arbeit der Cluster-Knoten überwacht.
Wird ein Ausfall erkannt muss ein Clusterswitch durchgeführt werden

5c) Nennen Sie zwei nicht-funktionale Anforderungen, welche durch eine Cluster-architektur positiv beeinflusst werden können. Geben Sie jeweils eine Begründung an. Punkte ___

- Skalierbarkeit: Eingehende Anfragen können parallel auf verschiedene Knoten verteilt werden
- Verfügbarkeit: Ausfall eines Knoten führt nicht zum Gesamtausfall des Systems

5d) Beurteilen Sie folgende Aussage: „Es müssen nicht alle Komponenten eines Clusters redundant ausgelegt sein“. Welches Problem ist darin enthalten? Punkte ___

- Die Verfügbarkeit ist nur gegeben wenn alle Teilkomponenten die gleich hohe Verfügbarkeit haben
- Sollte eine Komponenten mit geringerer Verfügbarkeit ausfallen ist das Gesamtsystem betroffen. Das ist nicht der Sinn eines Clusters

6 Design Pattern, Enterprise Patterns

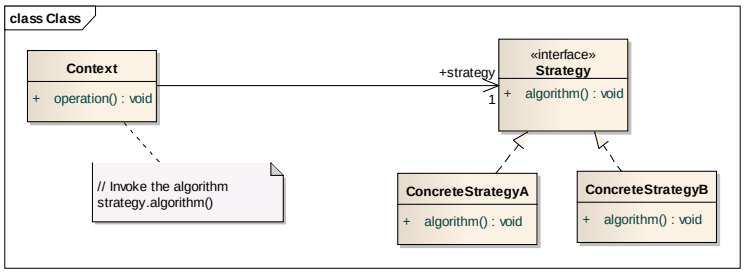
6a) Welches Grundproblem wird durch die Verwendung von Erzeugermustern adressiert? Geben Sie ein Quellcodefragment an, welches das Problem verdeutlicht. Begründen Sie, warum die Flexibilität einer Software durch die Anwendung dieser Kategorie von Mustern erhöht werden kann und welches Architekturprinzip dadurch positiv beeinflusst wird. Punkte ___

Quellcodefragment:

```
___Klasse a = new Klasse()_____________________________________________________
```

- Die zu erzeugende Klasse wird im Quellcode festgelegt
- Soll die Implementierung getauscht werden muss der Quellcode angepasst werden
- Erzeugerpattern trennen die Erzeugung der Objekte von deren Verwendung. Dadurch bleiben die Clients unabhängig. Software kann flexibel erweitert werden da Clients nicht verändert werden

6b) Erläutern Sie die Grundidee des Strategy-Patterns. Punkte __

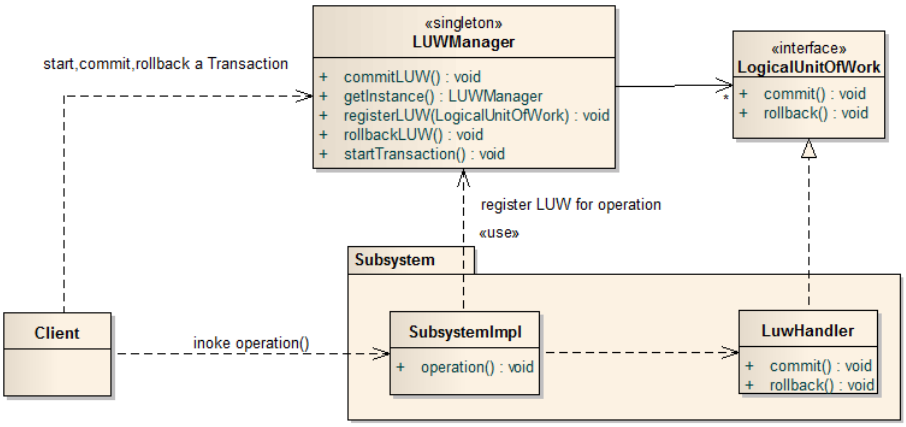


- Das Strategiemuster soll es erlauben dass ein ganzer Algorithmus in Form einer Kapsel ausgetauscht wird
- Eine Anwendung möchte verschiedene (Lösungs-)Strategien für ein Problem anbieten
- Eine allgemeine Operation benutzt verschieden Strategien um ein Problem zu lösen
- Die Strategie wird abstrakt definiert und kann in dem Algorithmus injiziert werden
-

6c) Wie kann man die Idee des Strategy-Patterns bei der Verwendung von Collections sinnvoll anwenden? Erläutern Sie die Idee und geben Sie ein Beispiel an. Punkte __

Sortieren von Collections mit Hilfe von Comperator-Interface (= Strategie)
Ein allgemeiner Sortieralgorithmus. Die Vergleichsoperation wird als Strategie ausgelagert und kann von außen Injiziert werden. D.h. der gleiche Algorithmus kann durch die Strategie parameterisisert werden.

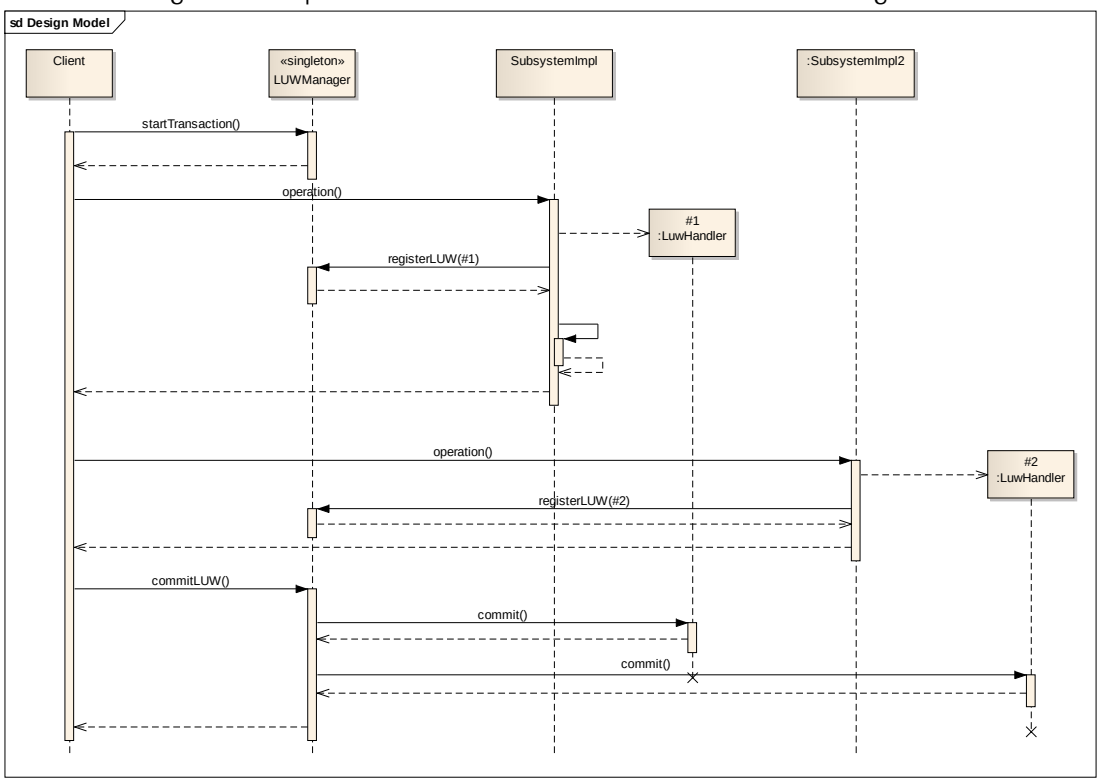
6 Design Pattern, Enterprise Patterns



6d) Welches Problem wird durch das Pattern „Logical Unit of Work“ gelöst? Welche Lösungsidee liegt diesem Pattern zugrunde? Punkte __

- Wird ein Client-Request ausgeführt werden intern verschiedenste Datenstrukturen verändert (=Zustandsänderung)
- Wie kann die Konsistenz erhalten werden, wenn der Aufruf aufgrund eines Fehlers nicht vollständig durchgeführt werden kann?
- Wie erfolgt ein Rollback auf den beteiligten Datenstrukturen?
- An dem LUW-Manager können LUW-handler registriert werden welche am Ende der Gesamttransaktion aufgerufen werden. Abhängig vom Zustand der Transkation können die Handler die korrekten Zustand der Datenstrukturen wieder herstellen

6e) Erweitern Sie das nachfolgende Sequenzdiagramm so, dass der prinzipielle Ablauf des Patterns „Logical Unit of Work“ sichtbar wird. Fügen Sie falls notwendig neue Objektinstanzen bzw. Methodenaufrufe in das Diagramm ein. Punkte __



Erzielte Punkte:_____