Software Engineering Besprechung Übung3: Softwarearchitektur

Softwarearchitektur

Teaching Team

Paul Muntean
muntean@ifi.uzh.ch

Martina Rakaric
martina.rakaric@gmail.com

Abgabe

Abgabe	OLAT
Erlaubte Datentypen	PDF Datei oder .zip-File falls Source Code enthalten
Beschriftung PDF	Ex[**n**] [**NameA_Matrikelnummer**].pdf, wobei [**n**] die Nummer der Übung ist. Die PDF Datei sollte ausserdem ebenfalls Ihren Namen und Matrikelnummer beinhalten.

Aufgabe 3.1
a): Prozessarchitektur

Aufgabe 3.1
a): Prozessarchitektur

Teilaufgabe 3.1 A

Wie viele parallel laufende Prozesse brauchen Sie in der Implementierung Ihrer App? Begründen Sie Ihre Entscheidung. Wenn Sie mehr als einen Prozess brauchen, geben sie für jeden Prozess an, was seine Hauptaufgabe ist.

Nicht Korekte Loesung (Entnohmen von Studenten Loesungen)

"Die Applikation soll die Daten des ausgewählten Datensatzes interaktiv visualisieren" und "die Applikation soll die Daten wahlweise tabellarisch oder geographisch visualisieren" müssen parallel laufen.

Hier war es nicht klar das es um parallel ablaufende “Threads” geht!

Muster Loesungen (Entnohmen von Studenten Loesungen)

Beispiel 1

Wir planen in unserer Applikation **1 Prozess** zu verwenden.

Da wir nur eine manuelle Aktualisierung der Daten benötigen, kann das Programm die Aufgaben sequentiell abarbeiten.

Mehrere gleichzeitige Requests aus dem Web-Frontend werden über parallele Threads gelöst, welche jedoch im selben Prozess arbeiten.

Beispiel 2

Die Applikation benötigt zwei parallel laufende Prozesse:

Einer der beiden Prozesse stellt die Nutzeroberfläche bereit. Dazu gehören alle grafischen für den Endnutzer zugänglichen Darstellungen, welche als Schnittstelle zwischen dem Endnutzer und der Software dienen.

Der andere Prozess widmet sich allen im Hintergrund ablaufenden Prozessen. Der Prozess wickelt alle Anwendungen ab, die auf dem Server laufen und Daten verwalten.

Aufgabe 3.1
B): Identifikation von Komponenten 1

Teilaufgabe 3.1 B

Identifizieren und benennen Sie die Komponenten, aus denen Ihre App bestehen soll. Charakterisieren Sie jede Komponente in 1-2 Sätzen.

Nicht Korekte Loesung (Entnohmen von Studenten Loesungen)

- A web browser (or client);
  Client handles the presentation logic, which controls the way in which users interact with the application.
- A web application server;
  Web application server manages the business logic. It also manages requests from a variety of remote clients.
- A database server.
  Application relies on a database server, which provides the data for it.

- Wir werden keine Datenbank haben
- Geben Sie andere Komponenten Beispiele

Muster Loesungen(Entnohmen von Studenten Loesungen)

Darstellungskomponente (View): Zuständig für die Darstellung und das laden der Datensätze als Text oder Grafik.

Filterkomponente (Filter): Zuständig für die Segmentierung, Filterung und Facetierung des Datensatzes. Liefert den gewünschten gefilterten Datensatz zur weiteren Verarbeitung zurück.

Speicherkomponente (Export): Zuständig für die Speicherung der benutzerdefinierten Visualisierung aufgrund der gewünschten Filterkriterien. Liefert die gewünschte Visualisierung im gewünschten Export-Format (z.B. als Bild).

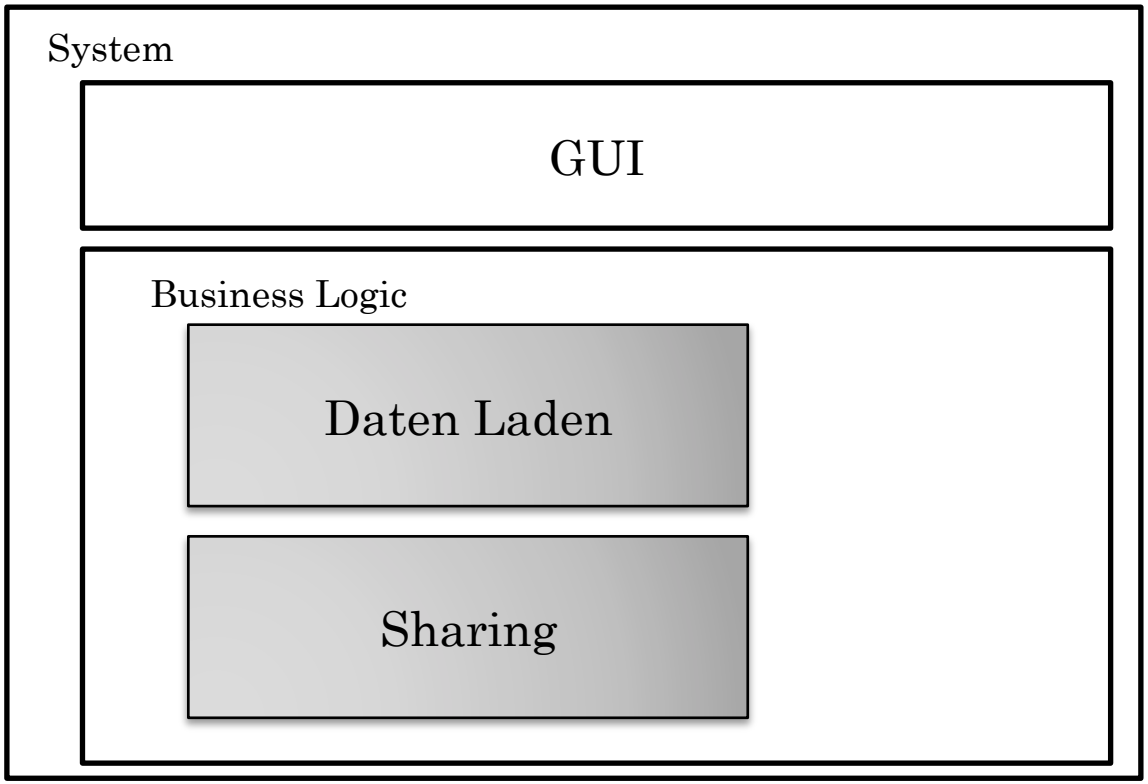
Kommentarkomponente (Comment): Zuständig für das Auslesen und Speichern von Kommentaren und Bildern auf Visualisierungen.

Aufgabe 3.1
C): Vorläufiges Komponentendiagramm

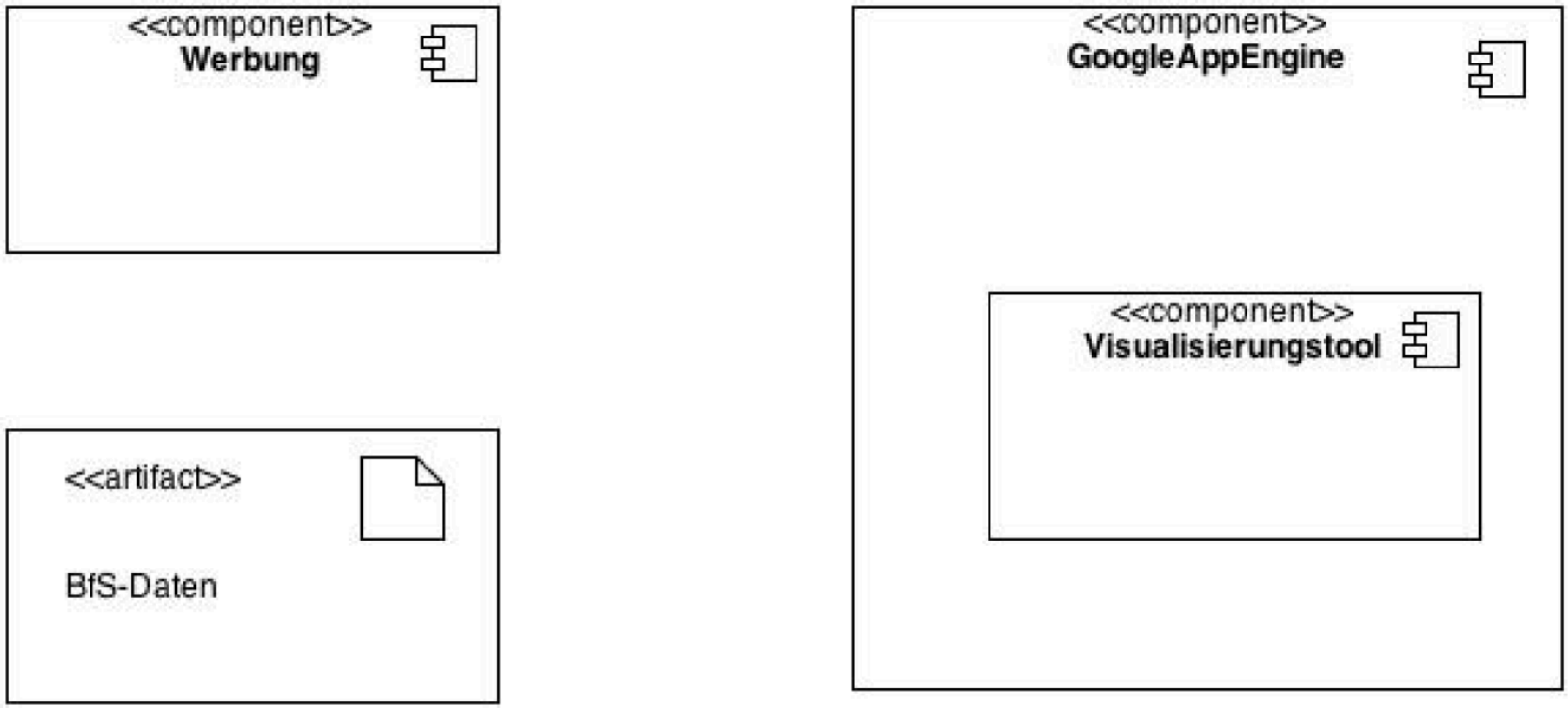
In welchen Kategorien koennen die Komponenten eingeteil werden? View, etc.

Teilaufgabe 3.1 C

Zeichnen Sie ein vorläufiges UML-Komponentendiagramm.

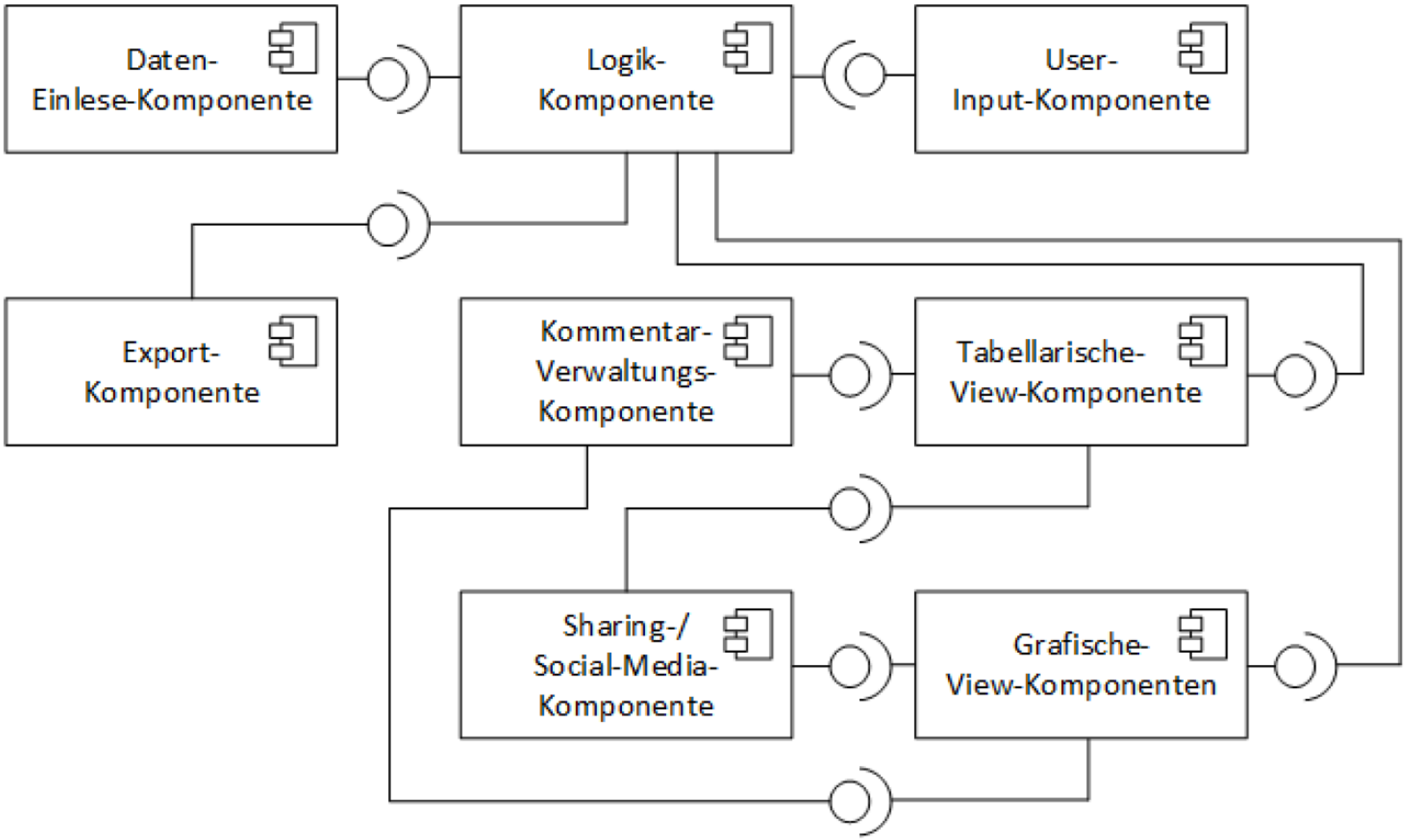


Nicht Korekte Loesung (Entnohmen von Studenten Loesungen)



Die Beziehungen zwischen den Komponneten fehlen!

Muster Loesung (Entnohmen von Studenten Loesungen)



3.2 Klassen und Schnittstellen Beispiel

Teilaufgabe 3.2

In dieser Teilaufgabe geht es darum, dass Sie die wichtigsten Objekte, Attribute (in UML auch features genannt) und Operationen Ihrer Architektur identifizieren. Diese abstrahieren Sie zu Klassen in UML. Charakterisieren Sie jede Klasse mit einem Satz. Es bewährt sich, die Klassen zunächst tabellarisch zu beschreiben (siehe Beispiel unten).

Name	Attribute	Operationen
«interface» EreignisLog Stellt ein Logbuch für Ereignisse bereit		+eintragErzeugen (neuerText: String, neueSorte: Ereignisart) +eintragLöschen (autorisiert: Rechte) +eintragFinden (von: Zeitstempel, bis: Zeitstempel): Logeintrag +nächstenFinden (dieses: Logeintrag): Logeintrag

Werbesseungs wuerdig Loesung (Entnohmen von Studenten Loesungen)

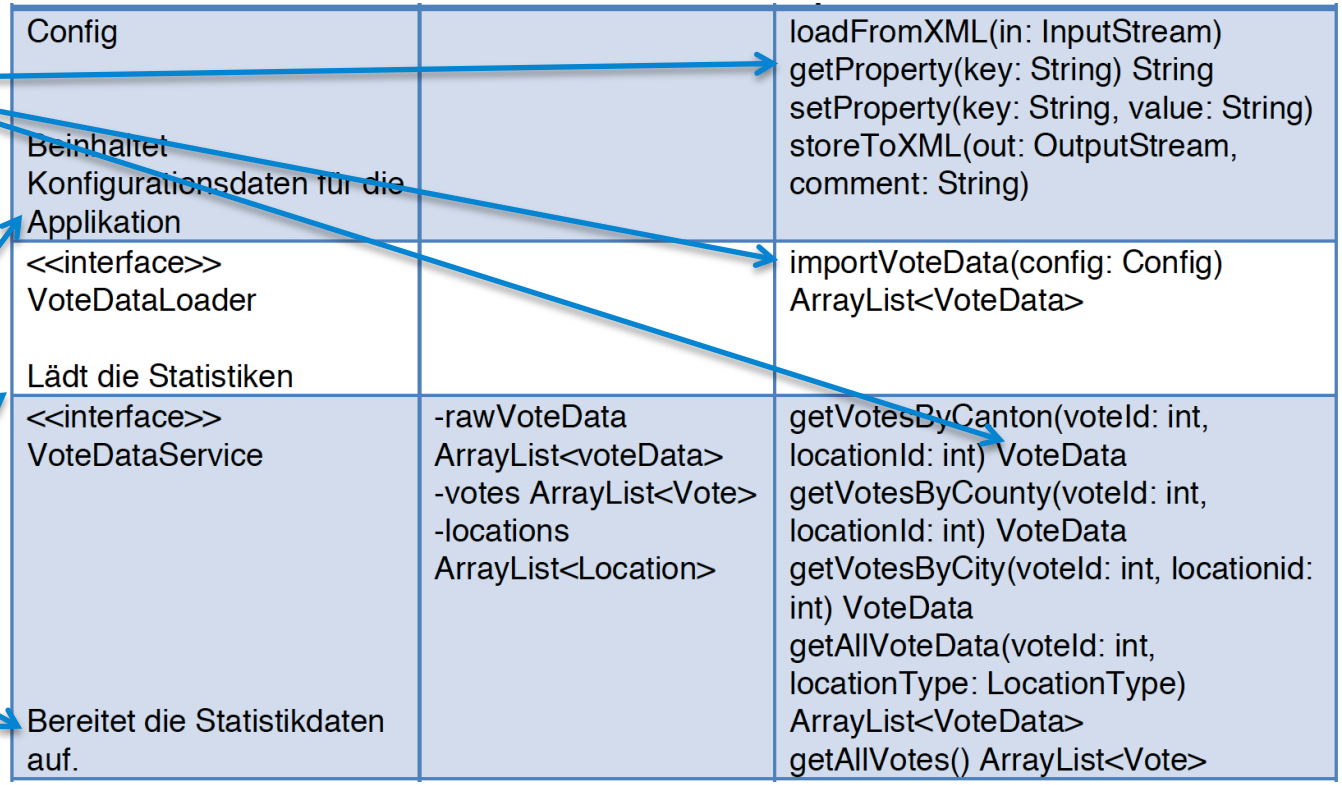
Operationne haben keine Pameter und keine “Return value”!

Jede Klasse, Interface is gut beschrieben.

Datenlader Lädt daten periodisch? (thread simuliert) herunter und stellt diese bei bedarf bereit	- daten[]: Array[Data]	+ ladeDaten()
{abstract} Visualisierung Basisklasse für die Visualisierungen		+ zoomIn() + zoomOut() + wechsleAuswertung (auswertung:Integer)
GeographischeVisualisierung Stellt die gewählten Auswertungen graphisch dar.	+ istAktiv: Boolean	+ override zoomIn() + override zoomOut() + override wechsleAuswertung (auswertung:Integer)
TabellarischeVisualisierung Stellt die gewählten Auswertungen tabellarisch dar.	+ istAktiv: Boolean	+ override zoomIn() + override zoomOut() + override wechsleAuswertung (auswertung:Integer)
Kommentar Klasse, welche Textkommentar und Bild enthalten kann	+ kommentar: String + bild: Image + x: Integer + y: Integer	+ Kommentar(text:String, bild:Image)

Muster Loesungen (Entnohmen von Studenten Loesungen)

Muster Loesung, Parameter und return value fehlen nicht.



Achtung: Jede Klasse, Interface is beschrieben.

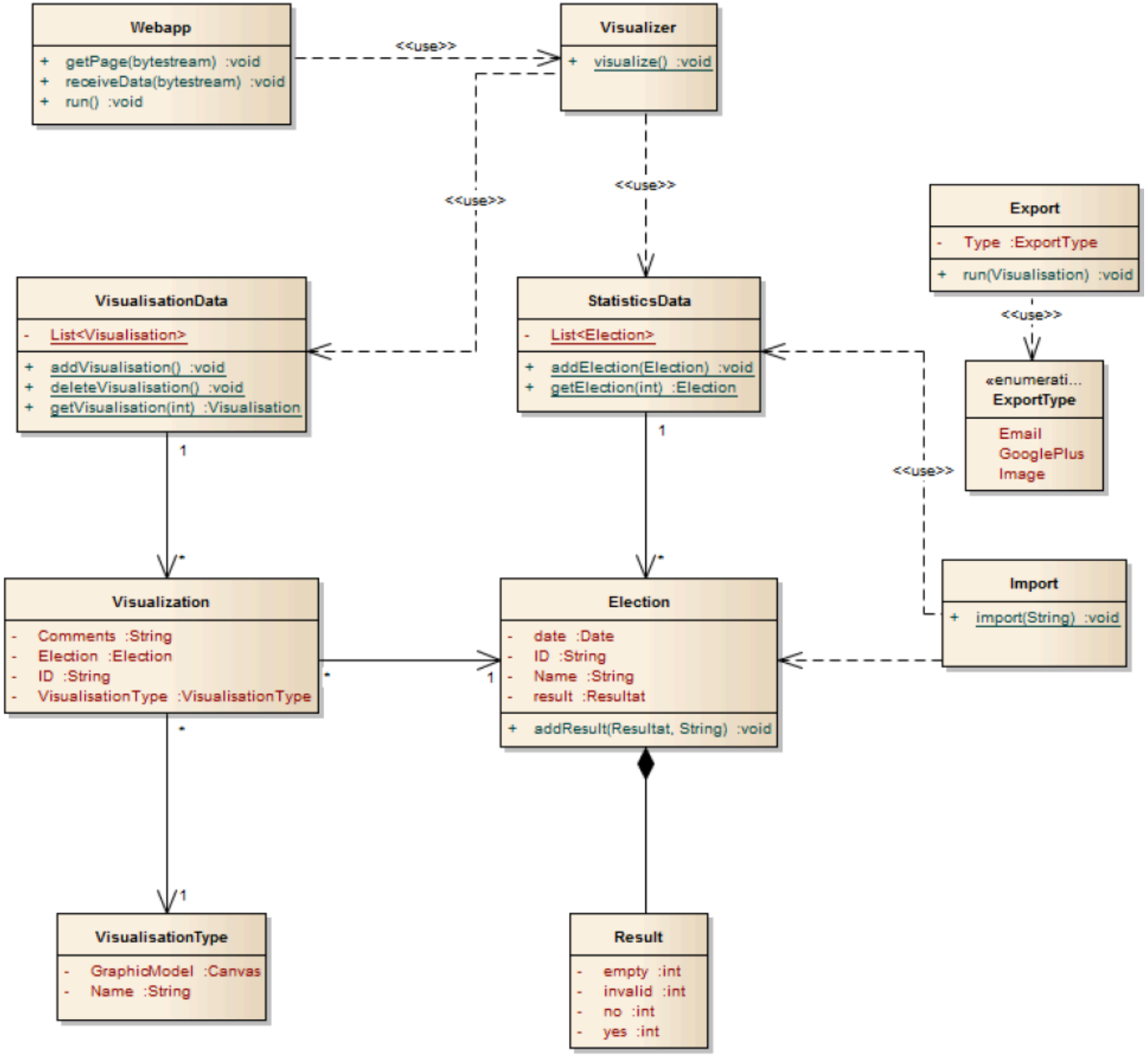
3.3 Klassendiagramm Beispiel

Teilaufgabe 3.3

Erstellen Sie an Hand der Ergebnisse von Teilaufgabe 3.2 ein Klassendiagramm Ihrer App. Dieses soll sowohl Klassen als auch Schnittstellen enthalten.

- Das Klassendiagramm soll nur die Klassen und Interfaces beinhalten die in der vorherigen Teilaufgabe aufgelistet wuerden.
- Wichtig ist das man die UML Notation berucksigtickt
- Operation solten nicht neu benant werden im Klassendiagram
- Paramenter Namen sind identisch wie in der worherigen Teilaufgabe

Muster Loesungen(Entnohmen von Studenten Loesungen)

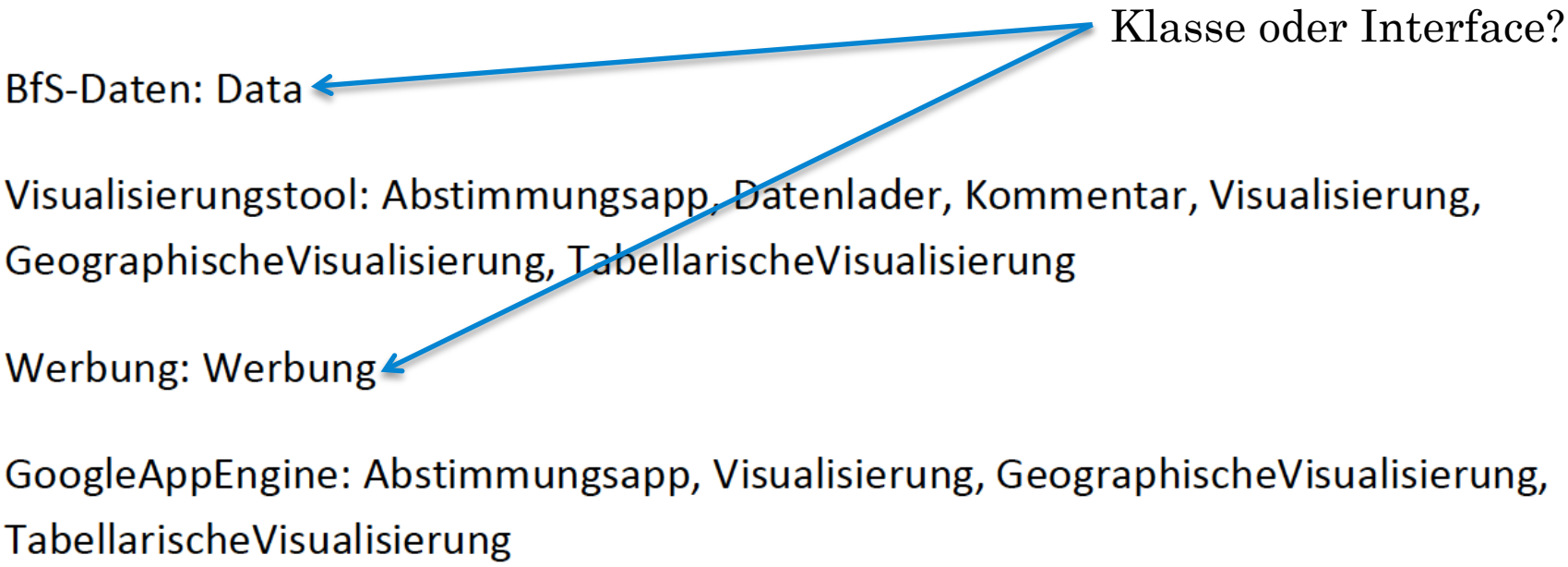


3.4 Komponentenmodell
a) Zuordnung: Klassen & Schnittstellen

Teilaufgabe 3.4 A

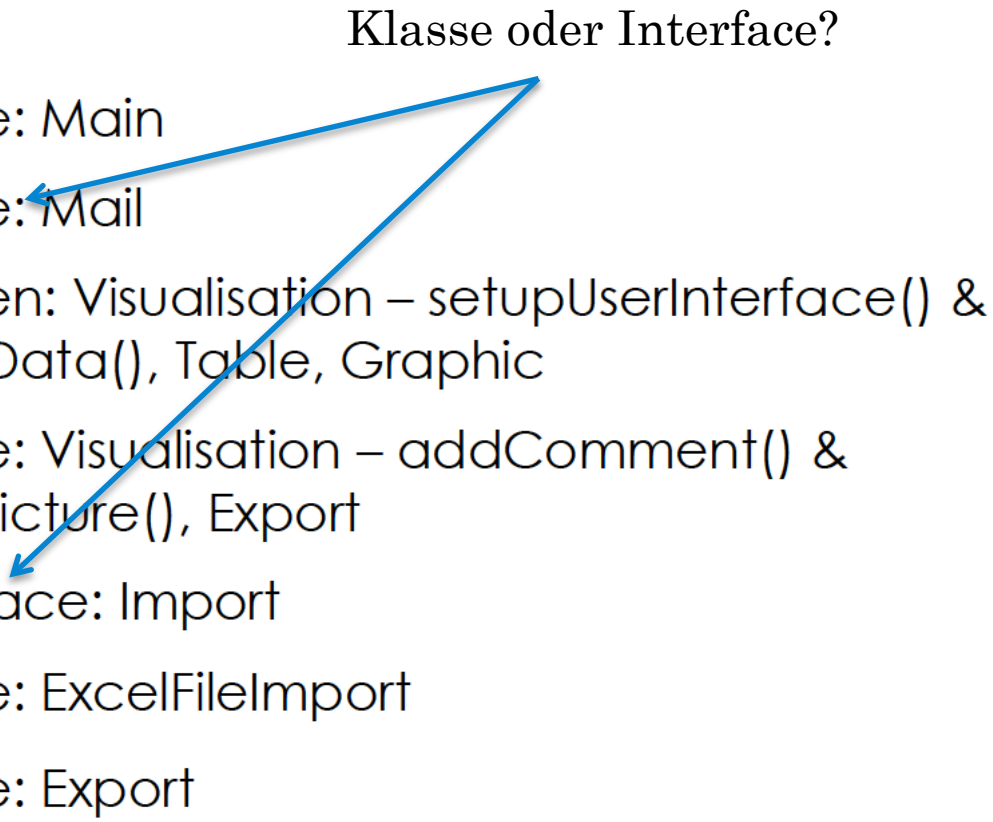
Zuordnung von Klassen und Schnittstellen zu Komponenten (2 Punkte)

Nicht Korekte Loesung (Entnohmen von Studenten Loesungen)



- Welche von denen sind Klassen und welche Interfaces. Es existiert keine definierte einteilung.
- Prioritaer ist das die eigenen Komponenten modeliert werden.
- Die *GoogleAppEngine* Komponente ist eine “Off the shelf Component”, sie wird nur wieder verwendet.

Muster Loesungen (Entnohmen von Studenten Loesungen)



3.4 Komponentenmodell B) Erweitert

Teilaufgabe 3.4 B

Erweitern Sie Ihr Komponentendiagramm aus Teilaufgabe 3.1, indem Sie die Schnittstellen zwischen den Komponenten explizit modellieren.

Beispiel:

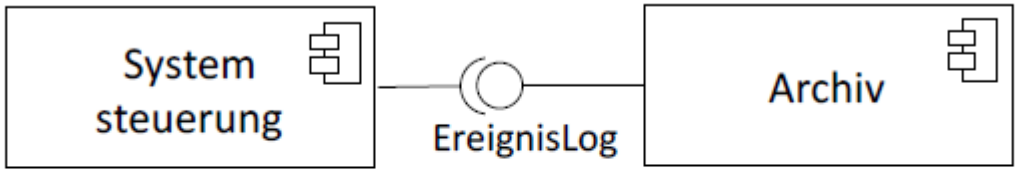
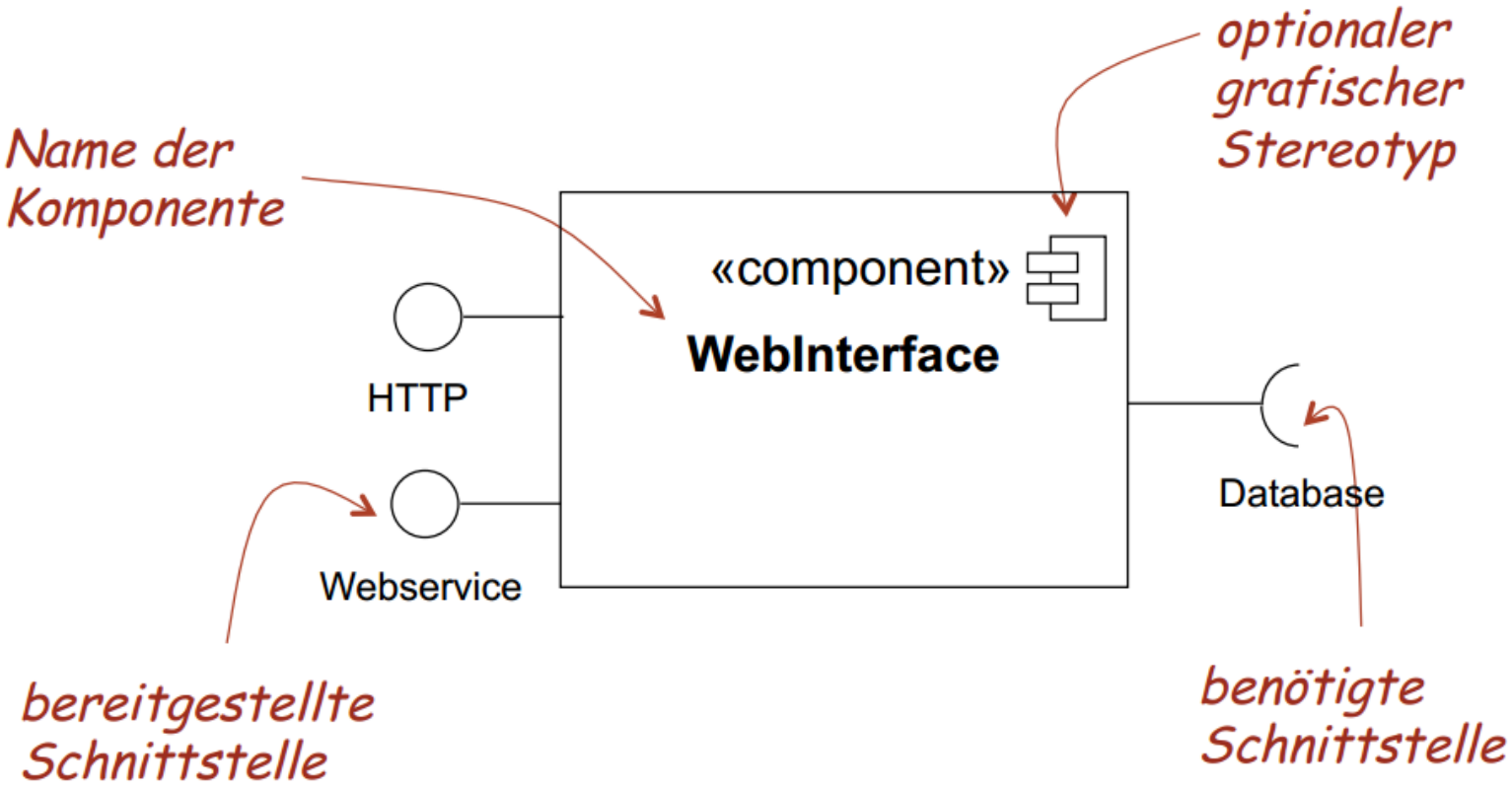


Bild 1. UML-Komponentendiagram. Die Komponente Archiv bietet die Logging-Funktionalität der Komponente Systemsteuerung mit Hilfe der EreignisLog Schnittstelle.

3.4 Komponentenmodell
B) Erweitert

Beispiel: WebInterface Komponente



Kohäsion & Kopplung

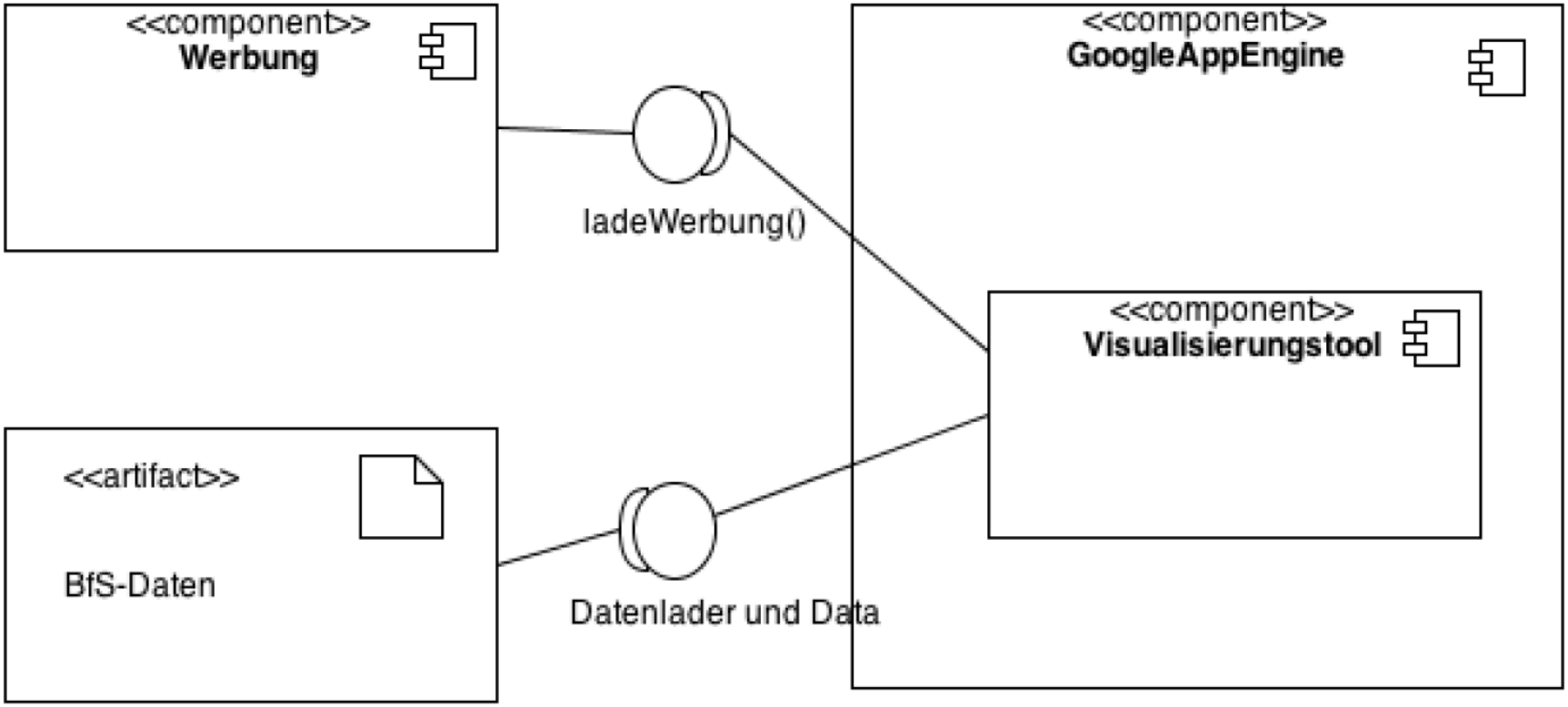
Hohe Kohäsion:

- Kohäsion = "Zusammenhalt"
- Die Dinge sollen in Struktureinheiten zusammengefasst werden, die inhaltlich zusammengehören.

Niedrige Kopplung:

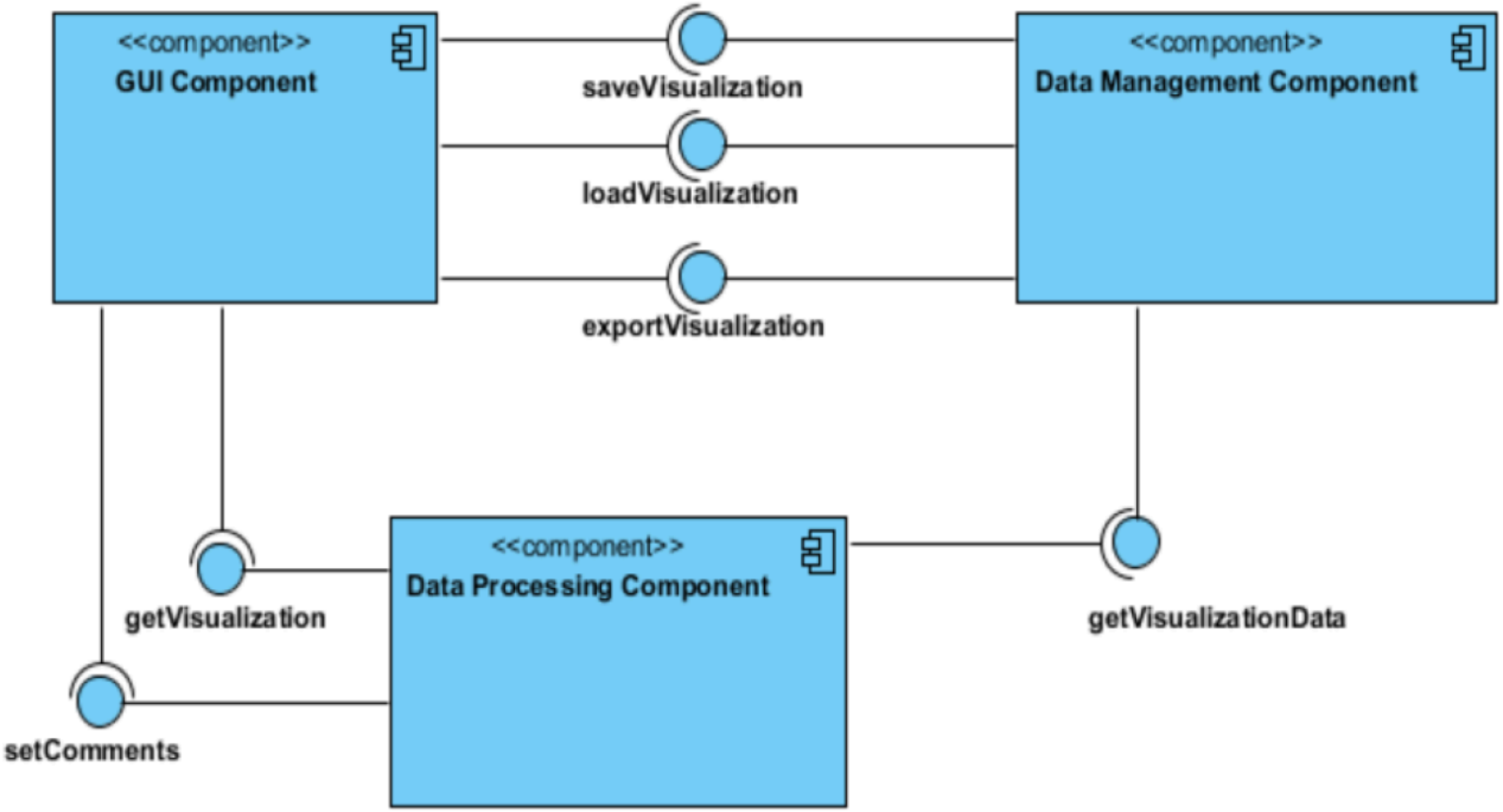
- Kopplung = Abhängigkeiten
- Einzelne Struktureinheiten sollen möglichst unabhängig voneinander sein.

Nicht Korekte Loesung (Entnohmen von Studenten Loesungen)



Die Interfaces sind nicht modelliert worden. Was ist *Datenlader und Data*?

Muster Loesungen (Entnohmen von Studenten Loesungen)



3.5 Dokumentation und Annahmen

Teilaufgabe 3.5

Dokumentieren Sie alle getroffenen Annahmen explizit in einer Liste. (In einem realen Projekt würden diese Annahmen fortlaufend mit den Interesseneignern geklärt.)

- Jede Annahmen mus Dokumentiert werden.
- Die Annahmen muessen auf den Informationen basieren die im rahmen des Projektes besprochen wuerden, 8.10 Besprechung und Anforderungsspezifikation Dokument

Muster Loesungen (Entnohmen von Studenten Loesungen)

- Umfang der Informationen der Abstimmungen

  In einem ersten Schritt gehen wir davon aus, dass nur die totale, ja und nein Anzahl Stimmen relevant sind.

- Anzahl Abstimmungen pro Visualisierung

  In einem ersten Schritt soll pro Visualisierung nur eine Abstimmung ausgewählt werden können.

GWT STOCKWATCHER
DESIGN PATTERNS LOESUNGEN

Teilaufgabe 3.6

Durchsuchen Sie das StockWatcher GWT Tutorial und finden Sie eine Stelle, an den ein Architekturmuster oder Entwurfsmuster verwendet wird. Für das gefundene Muster nennen Sie:

- Den Namen des Musters und seine Kategorie,
- Die Liste der Beteiligten in diesem Muster,
- Die Java Elemente (Klassen, Schnittstellen, Methoden), welche im Rahmen dieses Musters zusammenarbeiten und welche Rolle sie in diesem Muster einnehmen.

Loesungen: MVC, Client-Server, Observer

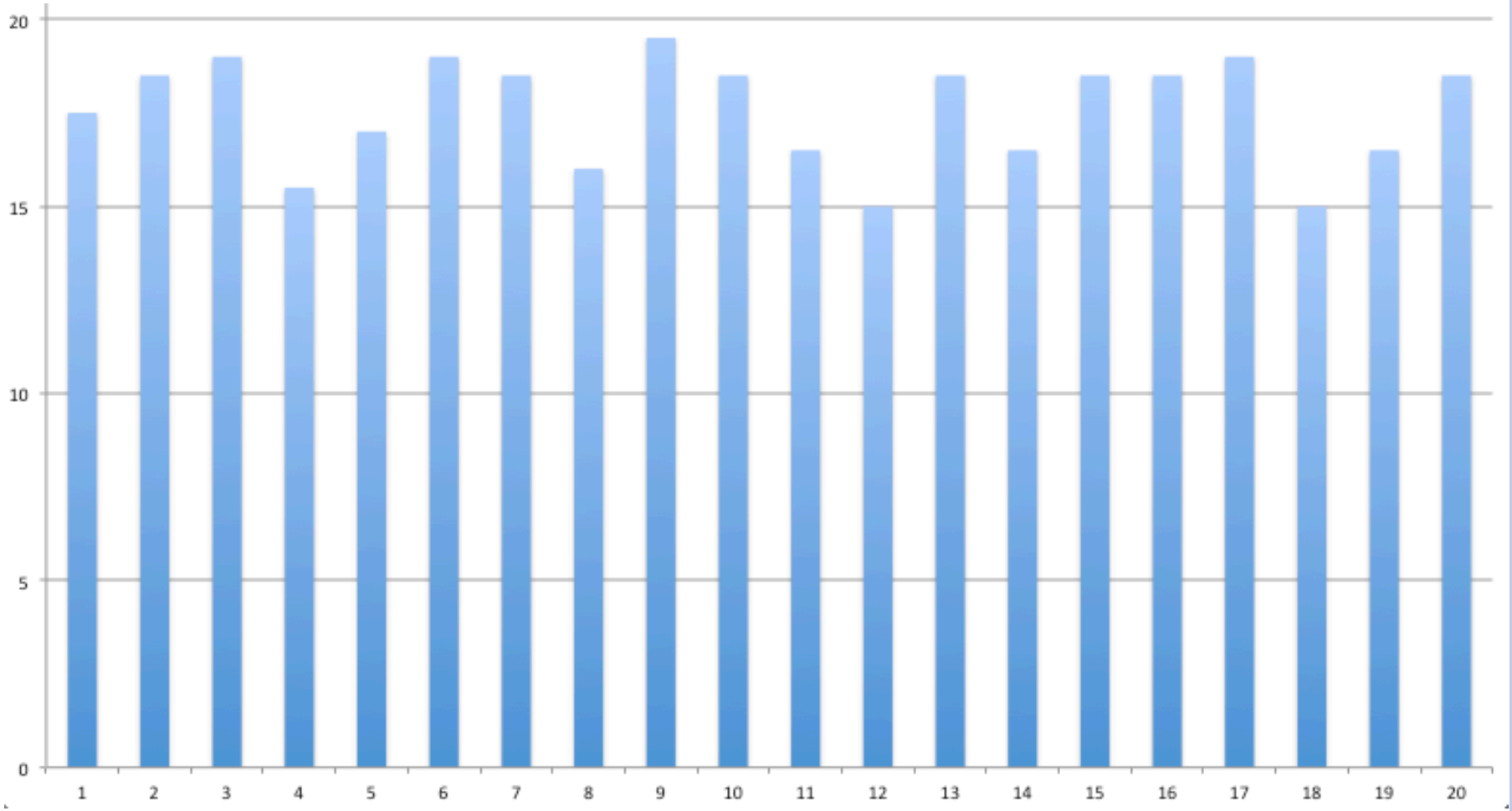
Beispeil fuer Client-Server

Client-Server System – Architekturmuster - GWT StockWatcher Tutorial - Starter Applikation

Liste der Beteiligten:

Clientseite:	GreetingService.java (stub)
	GreetingServiceAsync.java (Asynchrones Gegenstück zum stub)
Serverseite:	GreetingServiceImpl.java
Beide:	FieldVerifier.java

PunkteVerteilung



X Axe: Die Gruppe, Y Axe: Punkte

Fragen?