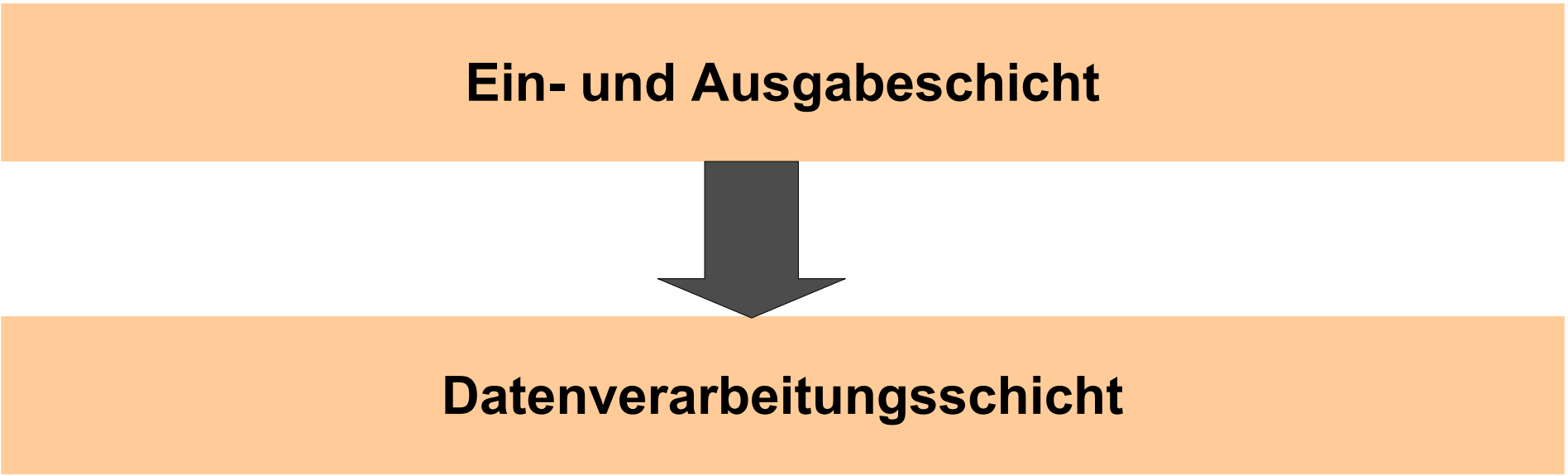
Software Engineering

Zur Architektur der Applikation
Data Repository

Franz-Josef Elmer, Universität Basel, HS 2015

Schichtarchitektur



Verantwortlichkeiten

Ein- und Ausgabeschicht

- Aufbereitung der Eingabe inklusive Überprüfung
- Erzeugung von internen Datenobjekten, die die Eingabe repräsentieren
- Aufruf geeigneter Methoden der Datenverarbeitungsschicht
- Aufbereitung und Ausgabe der Rückgabeobjekte der Datenverarbeitungsschicht

Datenverarbeitungsschicht

- Überprüfung der von der Ein- und Ausgabeschicht erhaltenen Datenobjekte.
- Daten verarbeiten (z.B. Metadaten im Repository suchen, Dateien in das Repository kopieren)
- Rückgabeobjekte erzeugen und an die Ein- und Ausgabeschicht weitergeben.

Kappselung durch API

Die Datenverarbeitungsschicht sollte nur durch eine Schnittstelle (Java Interface) zugänglich sein.

- Es ist naheliegend, dass die Methoden den Kommandos entsprechen.
- Zur API Definition gehören auch alle speziellen Datenklassen der Methodenparameter und Rückgabewerte.
- Die Datenklassen sollen nur Daten enthalten und keine Logik. Sie sind massgeschneiderte Datencontainer. Es ist sinnvoll auf die Instanzenvariablen nicht direkt zuzugreifen sondern Getters und Setters zu benutzen.
- Methoden zu Kommandos mit `-verbose` Flag haben als Parameter ein Callback Objekt (als Java Interface definiert), welches Fortschrittsinformationen entgegennimmt.
- Jedes Interface und jede Klassen enthält ausführliche Javadoc welche vor allem die Semantik der Methoden erläutert sowie die Bedingungen an die Parameter (z.B. ist `null` erlaubt oder nicht) und der Rückgabewerte (z.B. ist eine `null` möglich).

Verwendung der API

- Die Ein- und Ausgabeschicht kennt nur die Interfaces und Klassen der API. Es ist nicht erlaubt, dass sie andere Klassen aus der Datenverarbeitungsschicht benutzt.
- Die API kann von unterschiedlichen Arten einer Ein- und Ausgabeschicht verwendet werden (Kommandozeilenapplikation, GUI Applikation, Web Applikation).
- Die Klasse, welche das Interface der API implementiert, wird nur an einer Stelle ausserhalb der Datenverarbeitungsschicht benutzt, nämlich um eine Instanz zu erzeugen. Dies kann z.B. in der `main()` Methode der Applikation erfolgen.
- Das Callback Interface wird von der Ein- und Ausgabeschicht implementiert. Z.B. eine Version falls `-verbose` gesetzt ist und eine andere falls `-verbose` nicht gesetzt ist (Dummy Callback Objekt). Die Datenverarbeitungsschicht kennt nur das Interface, wie in der API definiert.

Paketstruktur

Java Pakete erlauben eine Gruppierung der Klassen nach Kriterien der Sichtbarkeit. Deshalb ist es sinnvoll die Klassen in folgende drei Pakete zur gruppieren:

- `api`: Enthält all Interfaces und Klassen der API. Diese sind alle public deklariert.
- `processing`: Hier sind alle Klassen der Datenverarbeitungsschicht untergebracht. Diese Klassen sind alle 'package protected' deklariert (d.h. ohne `public` Qualifier). Ausnahme: Die Hilfsklasse (eine sogenannte Factory) welche ein konkretes Objekt der implementierenden Klasse erzeugt.
- `apps.cli`: Es enthält alle Klassen der Ein- und Ausgabeschicht für den die Kommadozeilenapplikation (cli = command line interface). Hier ist auch die Klasse mit der `main()` Methode. Auch diese Klassen sollten package protected sein (bis auf die Klasse mit der `main()` Methode).

Als Beispiel für diese Architektur siehe
https://subversion.cs.unibas.ch/repos/hs15/cs203/showcase/