5.3 - Architecture

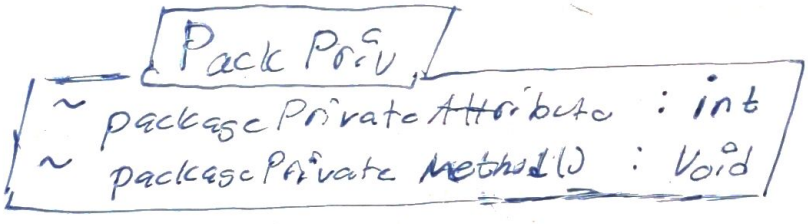
Visar "den stora bilden" på hur systemet,
visar systemet delas in i delsystem.
Tex en klass som anropar DB, ej vill en
hel databas.

Patterns

En vanlig och välkänd lösning till ett upprepande
problem. Ett problem som löses i många
program med typ samma lösning, göra en
formell beskrivning av problemmets varianter
på lösningar. "Kokbok"

Packages and Package Private Visability

Att en viss deklaration endast är synlig för det egna package:n visas med ~ (tilde)



53

MVC Architecture Pattern.

MVC = Model - View - Controller

Delar in systemet i tre delsystem:

- Model
- View
- Controller

View - User interface

Model: Business logic.

[Fig 5.18]

View: Sköter all GUI, kod till GUI får ej förekomma någon annanstans.

Controller: Ansvarar för att anropa rätt metoder i rätt objekt.

Model: Programmets representation av verkligheten. ansvarar för funktionaliteten, logiken.

Fördelarna med MVC

- Hög sammanhållning
- Låg

[Fig 5.19]

Lätt att uppdatera vicen tex.

The layer architectural Pattern

Mera generell än MVC, säger att systemet ska delas upp i lager. [Fig 5.2c]

Högre upp, närmare anv.

Längre ner, längre från anv.

The DTO (Data Transfer Object) Design Pattern.

För att förenkla metodanrop, minska antalet parametrar. Skapas "dumma klasser", enda syftet är att skyffla data.

Skilja mellan entitet och DTO

Två DTO klasser är lika om alla dess attributer är lika.

Entiter är lika om något ID är lika tex bankkonto.

Annat sätt: DTO är read-only.

5.4 - A Design Method

Designa ett program steg för steg.

1. Använd MVC och layer-modellerna.
   Ett paket för varje lager. Börja med [Fig 5.2]

2. Design one system operation at a time:

   SSD visar vilken input/output data programmet ska ha.

   Använd interaktions diagram, som visar flödet.

   [Ej] klass diagram.

(3) Sträva efter hög cohesion
   Låg coupling
   hög grad av inkapsling
   litet, väl def. public interface

4) Maintain a class diagram.

Efter system operationerna har designats, summera klass diagrammet. Ska visa paket, klasser, metoder, associationer. Attributer får vara med.

5) Implement the new design in code.

Görs i iterationer, implementera → utvärdera

6) Börja om på steg 2, och designa nästa system operation.