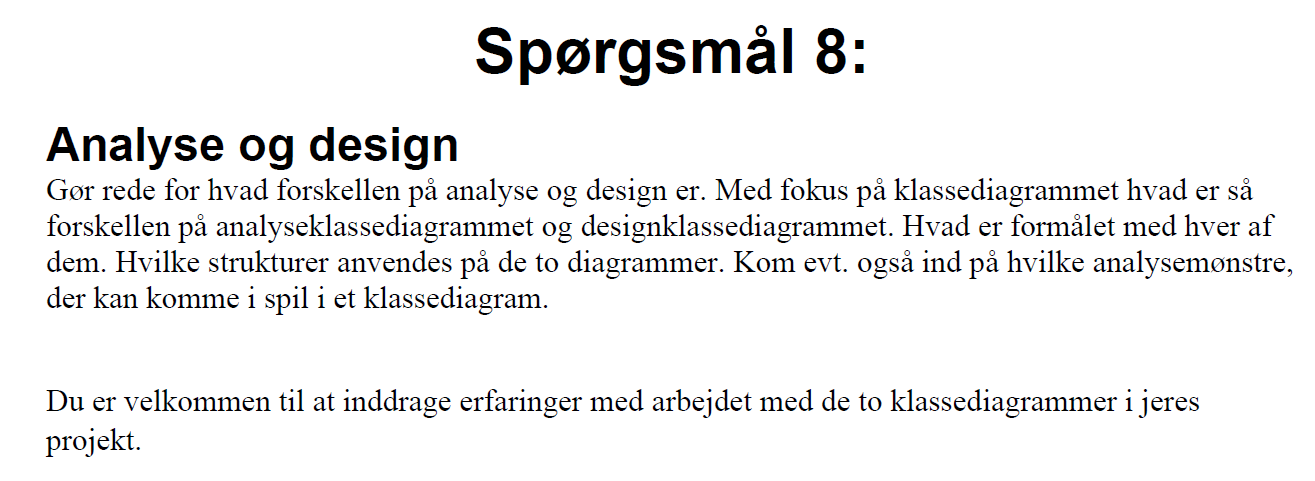
Spørgsmål 8



Inddrag forskellen mellem design og analyse klassediagrammer

inddrag analysemønste

Inddrag bryghus klassediagram

Analyse klassediagram har følgende:

* Klasser.
* Attributter.
* Multipliciteter.
* Associationer i henhold til composition, aggregering eller nedarving, eller almindelig association.
* Samt sammenhængen mellem klasserne.

Design klassediagram er en videreudvikling af analyse klassediagrammer. Her begynder det at ligne hvordan systemet faktisk er kodet.

De har følgende:

* Klasser
* Attributter - private/public
* Metoder - private/public
* Sammenhænge
* Multipliciteter
* Associationer
* Retninger
* Specielle klasser såsom enum’s, storage og controller.

Der har fjernet følgende fra analysen (associations klasser)

Mønstre bruges til at konkretisere en bestemt måde at sætte klasser op på i et klassediagram og bruges oftest til løsning af et generelt defineret problem – bruges når/hvis man støder på et problem som er gentagende

Analyse mønstre bruges i klassemodelleringen; design mønstre bruges i forbindelse af valg af arkitektur – f.ex. 3-lags-modellen – programmeringsmønstre (design patterns) bruges i selve kodningen – f.ex. singleton

Rollemønster: bruges når et overordnet individ kan være af flere forskellige roller, f.ex. en person som kan være student eller teacher – lav her en Person klasse; gør det lettere i fremtiden hvis der f.ex. kommer nye roller (vikar, praktikant, osv)

Genstand-beskrivelsesmønster: velegnet til, når man har et antal objekter, der har nogle generelle egenskaber til fælles, f.ex. attributværdier - modellers ved at lave et objekt af den overordnede klasse, som så holder på fælles værdierne for de underordnende objekter

Hierarkimønster: et system hvor der potentielt kan være uendelig mange niveauer – en skole består af afdelinger, som består af uddannelse, osv

Samlingsmønster: et hierarki hvor struktur og dybde ikke kendes på modelleringstidspunkt – f.ex. en indholdsfortegnelse til en opgave

For at et system er brugbart, fleksibelt og forståeligt, kræves det at systemet har en god kobling og samhørighed. En høj samhørighed er når en klasse har relativt få metoder der er let at vedligeholde, forstå og genbruge.

For at have en god kobling, skal der tildeles ansvar som derved giver høj samhørighed. Man skal derfor sørge for at GUI har lav kobling, og det er Controller klassen som har alt ansvaret og derfor høj kobling. Målet er at have høj samhørighed og lav kobling.

Adfærdsmønster/Tilstandsdiagram: Beskrivelse af mulige (lovlige) hændelsesforløb for alle objekter af en klasse

• Elementer i diagrammet

• Tilstand (Boks med tekst)

• Starttilstand (Prik)

• Sluttilstand (Prik med cirkel om)

• Hændelse (tekst over transition pil)

• Transition (Pil + “hændelse” + (betingelse) + /hvad hændelsen trigger/)

• Anvendelse / betydning

• Få et overblik over systemet

• Fælles forløb for alle objekter i en klasse

• Bliver kun brugt hvis objekternes adfærd er tilstrækkelig kompliceret

• Hvordan realiseres tilstande • Med udgangspunkt i banksystemet:

• Hvis saldoen kommer under 0 vil hændelsen blive ført over i boksen overtrukket da de parametre nu er overholdt. Den vil herefter befinde sig i den hændelse indtil denne parameter ikke længere er overholdt.

• Med udgangspunkt i enrollment systemet:

• Hvis en student går igennem en examination, har studenten enten bestået eller fejlet, hvis bestået -> studenten går videre til næste semester, hvis fejlet -> studenten går tilbage til starten af semesteret

• Kode: Kan være attribut, enum, metode.. alt efter behov