BGPP Integreret Projekt

Udleveret fredag 28. november 2008

Formål med projektopgaven

I projektet skal I analysere et givet problem og programmere et softwaresystem til at løse det. I skal præsentere systemets formål, opbygning og virkemåde, og I skal afprøve systemet, redegøre for om det virker som ønsket, og vurdere i hvilken grad jeres afprøvning understøtter en sådan konklusion.

Projektet involverer udvikling, test og dokumentation af et Java-program med grafisk brugergrænseflade og brug af en relationsdatabase.

Emne for projektopgaven

I projektet skal der udvikles et softwaresystem til brug for den *ekspedient* der betjener billetlugen og telefonen i en mindre biograf. En *kunde* er en person der står foran billetlugen eller som ringer ind på telefon. Løsningen skal understøtte ekspedientens arbejdsopgaver (tasks) i forbindelse med kundehenvendelser, for eksempel:

- En kunde vil gerne have to pladser ved siden af hinanden til en bestemt forestilling (film og tidspunkt) i dag, helst bagest i salen.
- En kunde henter sine allerede reserverede billetter.
- En kunde vil gerne have 17 billetter til filmen *Jeanne d'Arc* kl 19:00 en hvilken som helst dag i den kommende uge.
- En kunde ringer og vil gerne reservere 4 billetter til en hvilken som helst film kl 19:00 i morgen, men helst ved siden af hinanden og så langt fremme i salen som muligt.
- En kunde vil gerne have yderligere 2 billetter føjet til den reservation på 4 som han har i morgen kl 19:00.
- En kunde ringer ind og vil gerne have flyttet sin bestilling af 6 billetter fra kl 19:00 forestillingen i morgen til kl 21:30 forestillingen med samme film.
- En kunde ringer ind og afbestiller de 6 billetter han bestilte i går.
- En kunde vil både købe 1 billet til *Syriana* ved forestillingen om 25 minutter, og reservere 5 billetter lørdag eftermiddag til *Pelle Erobreren*.
- En lang række varianter kan tænkes: Der er ikke nok ledige pladser ved siden af hinanden; der er slet ikke nok ledige pladser; filmen går ikke på det ønskede tidspunkt; filmen går slet ikke i denne biograf; osv.
- Lav selv en liste af sandsynlige arbejdsopgaver som ekspedienten skal kunne løse ved hjælp af systemet.

Løsningen skal kunne håndtere reservation af pladser, men ikke salg; dvs. billetpriser og betaling skal ikke implementeres. Ved reservation skal der opgives identitet, fx telefonnummer eller navn.

Biografen har kun én billetluge og tillader ikke reservation over nettet, så løsningen behøver ikke håndtere samtidige opdateringer til databasen.

Biografen har flere sale og viser forskellige film på forskellige tidspunkter af dagen i de forskellige sale. En kombination af film, sal, tidspunkt og dag kaldes en *forestilling*.

De forskellige sale kan have forskelligt antal stolerækker og antal stole per række. Antallet af sale og salenes antal stolerækker mv skal fremgå af databasen og må ikke være indkodet i selve billetlugeprogrammet. For enkelheds skyld kan det antages at alle rækker i en given sal har lige mange stole, og at der ikke er nogen midtergange eller krumme stolerækker.

For at biografen kan håndtere skiftende forestillingsprogrammer skal de aktuelle forestillinger fremgå af databasen; de må altså ikke være indkodet i selve billetlugeprogrammet. Det er ikke en del af opgaven at lave en administratorbrugergrænseflade til at redigere disse forestillingsoplysninger. I kan altså bare opdigte nogle sale og forestillinger og lægge dem ind i jeres database (gennem MS Access brugergrænseflade eller SQL-kommandoer).

Til inspiration findes en skitse af en mulig brugergrænseflade i figur 1. I må meget gerne lave jeres brugergrænseflade anderledes, lav den ikke mere indviklet, hellere simplere — fx er tabsene for oven nok overflødige.

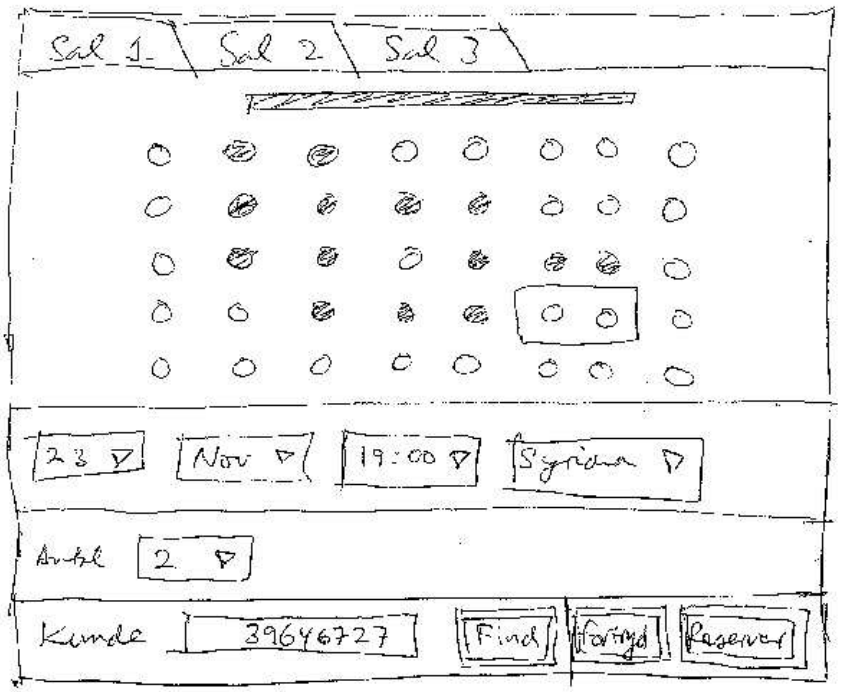

Figure 1: Skitse af mulig brugergrænseflade.

Regler for projektarbejdet

- Opgaven udleveres ved forelæsningen fredag 28. november og projektrapporten skal afleveres i **tre eksemplarer** senest **onsdag 17. december kl 15:00 i studieadministrationen**.
- Projektet udføres i grupper à 2 personer.
- I skal selv danne grupper og meddele Peter Sestoft (sestoft@itu.dk) navn og email på gruppens to medlemmer senest onsdag 3. december kl 17:00. Studerende der ønsker at lave projektet alene, jvfr. Eksamensbekendtgørelsen BEK nr. 231 af 22/03/2006, §4 stk 3, skal bekendtgøre deres ønske om dette ved at sende mail til Peter (samme tidsfrist).
- Systemet skal programmeres i Java. Man kan frit vælge databaseserver; Microsoft Access og MySQL er to oplagte muligheder. Forelæsningen tirsdag 2. december handler om Java og databaser.
- Om rapportens form og indholde, se vejledningen *Udformning af rapporter* der findes på kursushjemmesiden.
- Projektrapporten (eksklusive bilag) bør ikke overstige 25 sider.
- Rapportens forside skal oplyse kursus, projekttitel, forfatternes navn og ITU-email, dato.
- Der er instruktorhjælp til projektet tirsdag 1300-1500 og fredag 1000-1200 fra og med tirsdag 2/12 til og med fredag 12/12.
- Hver projektgruppe har mulighed for korte vejledermøder med Peter.
- Brug meget gerne kursets nyhedsgruppe `it-c.courses.BGPP` til at stille spørgsmål af almen interesse, fx om fortolkning af denne opgavetekst.
- Plagiat er forbudt. I må ikke bruge andres tekst, illustrationer eller programkode uden udtrykkelig kildeangivelse.

Vurdering af projektarbejdet

Hovedvægten i vurderingen lægges på projektrapporten, især:

- En præsentation, præcisering og afgrænsning af krav til systemet, fx i form en liste af de arbejdsopgaver (tasks) og de data som systemet skal håndtere.
- Begrundede designbeslutninger vedrørende brugergrænseflade, databasens design, og programmets interne struktur (klasser og objektstrukturer).
- Teknisk orienteret beskrivelse af databasedesign og programmets interne struktur.
- Begrundet afprøvning af det resulterende system, og vurdering af i hvilken grad systemet opfylder sit formål. Afprøvningen kan vedrøre brugergrænsefladens funktionalitet og unit test af klasserne.
- Kort brugervejledning til det resulterende system.
- Kort konklusion om i hvilken grad produktet opfylder de opstillede krav, herunder en mangelliste.
- Reflekterende beskrivelse af arbejdsprocessen, fx om I har nået projektets læringsmål.
- Bilag:
  - kondenseret projektdagbog, og endelige versioner af arbejdsblade
  - udskrift af Java-kildeteksten til det udarbejdede softwaresystem (overskueligt, med en ret lille font eller i to spalter på tværs af papiret)

Der lægges vægt på at brugergrænsefladen understøtter hyppigt forekommende arbejdsopgaver godt, og at den programmeringsmæssige løsning er velstruktureret, veldokumenteret og vedligeholdsvenlig og ikke indeholder overflødige dele.

Læg mærke til at projektrapporten både omhandler *produktet*, altså softwaresystemet og dets opbygning, og *processen*, altså hvordan I har arbejdet, med mest vægt på omtalen af produktet.

Gode råd

Det er meget bedre udtrykkeligt at beslutte at et specielt delproblem ("samme film vises i to forskellige sale samtidig") slet ikke håndteres i systemet, og eventuelt skrive det på en mangelliste, end at programmere en uigennemtænkt, uafprøvet og udokumenteret løsning til delproblemet. Sådan en delløsning kan komplicere den samlede softwareløsning, reducere dens brugbarhed og vanskeliggøre videreudvikling.

Lad være med at opfinde komplicerede softwareløsninger på opgaver som ekspedienten meget bedre kan løse visuelt, fx "finde fire ledige sæder ved siden af hinanden helst foran i salen men ikke ude til siden". Brug hellere opfindsomheden på at lave en god visuel understøttelse til ekspedienten, eller på at give mulighed for at bruge tastaturet i stedet for musen, der kan være meget belastende ergonomisk.

Lad være med at lave alt på en gang. Forsøg at se på hvert delproblem for sig, og lav arbejdsblade (jvfr. BPAK-kurset) til hvert delproblem, så I kan huske hvad I har overvejet og besluttet og hvorfor:

- Eksempel: Lav listen af ekspedientens arbejdsopgaver færdig.
- Eksempel: Undersøg om jeres brugergrænsefladedesign kan understøtte alle arbejdsopgaverne.
- Eksempel: Gennemtænk databasedesignet, opdigt nogle eksempeldata, og se om alle nødvendige data er til stede for at den tænkte brugergrænseflade ville kunne fungere.
- Eksempel: Find ud af hvordan en Java-komponent kan tegne en sal med givne dimensioner (antal stolerækker og antal stole per række), og vise de ledige, reserverede og solgte sæder med forskellige farver.
- Osv.

Hvis noget virker meget uoverskueligt, så *Solve a Simpler Problem First*. Lav en løsning der er åbenlyst utilstrækkelig, men dog et skridt i den rigtige retning. Derefter er I meget klogere, og kan tage et nyt skridt. Fx kan man starte med kun én sal i biografen, eller kun én stolerække, eller med at man kun kan bestille én billet ad gangen.

Husk at bruge tegninger og tabeller, ikke kun tekst, når I skriver rapport.