Introduktion

Det samverkande systemet består av tre olika applikationer samt en deltagarsimulering som ska kunna fungera tillsammans. Detta kräver någon sorts kommunikation, de olika applikationerna måste kunna 'koppla loss' deltagare från simuleringen och styra dessa själva. Applikationerna och deltagarsimuleringen måste också kunna hämta information om bland annat deltagares musiksmak, puls, hårfärg och namn.

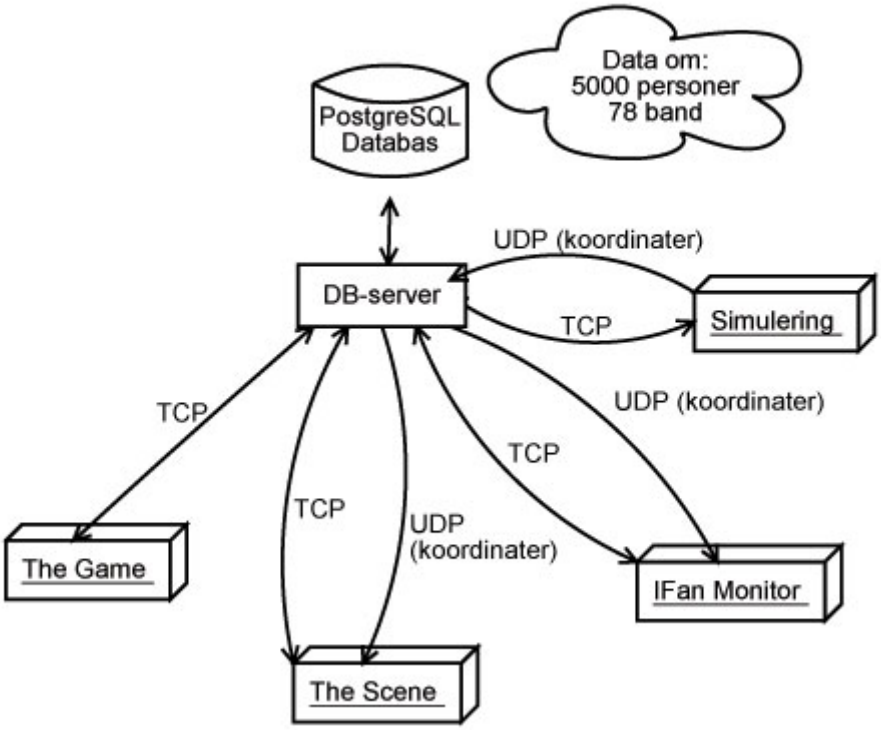
Designöverväganden

Kommunikation mellan systemets olika delar kan skötas på många olika sätt och här presenteras några av de överväganden som gjordes tillsammans med motivering:

- En centraliserad arkitektur valdes. Delsystemen kommunicerar med varandra via en central databas som hela tiden innehåller data om systemets deltagare. Denna centraliserade lösning är framför allt enkel och i detta fall när endast fyra delsystem finns, förhållandevis snabb.
- Kommunikationen mellan delsystem och databas är Internetbaserad. Detta val möjliggjorde parallell och programspråksoberoende utveckling av de olika delsystemen. Alternativet, att alla applikationer utvecklas i samma programspråk och körs tillsammans som en applikation på samma dator skulle inneburit mindre frihet för utvecklarna och stora samordningsproblem.
- Kommunikationen mellan delsystem och databas sker via både TCP och UDP. TCP-protokollet är bra och enkelt när applikationer ska kommunicera via textmeddelanden och är därmed lämpligt vid databasförfrågningar. Det lämpar sig däremot mindre bra när stora mängder data ska skickas snabbt, vilket till exempel är fallet när simuleringsapplikationen ska uppdatera positioner för alla besökare på festivalen. Därför kompletterades TCP-protokollet med positionsströmmning via UDP, främst för att förbättra prestandan i systemet.
- Databasserverns data om deltagare lagras i och hämtas från en PostgreSQL-databas. Information om deltagare på festivalen måste vara beständig och data måste kunna lagras undan någonstans för att systemet ska kunna stängas av och startas om. Den externa PostgreSQL-databasen innehåller bland annat information om deltagares namn och musiksmak men deltagarnas positioner, som är obeständiga och uppdateras mycket ofta, lagras lokalt i en datastruktur i databasservern av prestandaskäl.

Systemarkitektur

Nedanstående figur visar övergripande hur kommunikationen sker mellan de olika applikationerna, simuleringen och databasen. Den visar även övergripande hur Hultsfredsprojektets systemarkitektur ser ut.



DB-servern och PostgreSQL-databasen

En central del i detta är **DB-servern**. Den utgör själva navet i systemet och hanterar all kommunikation mellan de olika applikationerna, simuleringen och databasen. Hur dessa delar kommunicerar med DB-servern bestäms av **databasprotokollet** (se separat dokument). ”Bakom” DB-servern finns en **PostgreSQL-databas** där all data lagras (utom simulerade positioner) i totalt tre tabeller:

1. Över besökande personer och information om dessa (se protokollet för mer detaljerad information).
2. Över artister som spelar respektive inte spelar på festivalen och information om dessa (se protokoll).
3. Över vilka personer som är vänner med andra personer.

Databasen är i grunden fylld med 5000st besökare med genererad grunddata om dessa samt 78st artister. Förfrågningar om databasen från applikationerna och simuleringen hanteras enligt protokollet och generar SQL-frågor ur PostgreSQL-databasen vars svar formateras något innan de returneras.

Simuleringen

Simuleringen hämtar i ett inledande skede information om besökares id-nummer och genrepreferenser från databasen för att kunna göra en verklighetstrogen simulering. Detta görs via DB-servern över TCP enligt protokollet. Simuleringen generar sedan kontinuerligt nya koordinater för de besökande personerna som strömmas tillbaks till databasen över UDP. Dessa positioner mellanlagras i en datastruktur i DB-servern, de lagras alltså inte kontinuerligt i PostgreSQL-databasen. Detta för att kunna skicka dessa positioner snabbt via UDP till efterfrågande applikationer.

Applikationerna

Här följer kort hur de olika applikationerna använder sig av databasen. **IFan Monitor** hämtar bland annat information (via TCP) om besökande personer och vilka som är vänner med vem. Man får även simulerade positioner strömmade till sig via UDP för att kunna göra en kontinuerlig visualisering av hur besökare rör sig över området. **The Scene** hämtar bland annat information om besökarnas genrepreferenser och även denna applikation använder sig av simulerade positioner skickade till sig över UDP. **The Game** ”kopplar loss” besökare från simuleringen i ett visst ögonblick genom att ange att simulerade positioner för den besökaren inte längre ska lagras i DB-serverns datastruktur. Därmed styr denna applikation själv hur besökaren rör sig över området.

Databasens innehåll

Databasen lagrar information om 5000 festivaldeltagare och 78 artister, både artister som spelar på festivalen och andra artister. Innehåller är anpassat efter de olika applikationernas krav.

Data om festivaldeltagare

ID-nummer – varje deltagare har ett unikt ID.
Namn
Ålder
Längd
Kön – man eller kvinna.
Hårfärg
Puls
Position – var på festivalområdet deltagaren befinner sig.
Genrevektor – anger vilka musikgenrer deltagaren tycker om.
Vänner – vilka andra deltagare personen är vän med.
Partner – vilken spelpartner i **The Game** personen har
Poäng – poäng i **The Game**
RFID-tag – identifierar en person i **iFan Monitor**
Flaggor som indikerar om deltagaren är simulerad eller inte och huruvida deltagaren är med i **The Game**.

Data om artister

Namn – varje artist har ett unikt namn
Genrevektor – anger vilka musikgenrer artisten hör till.
Info – kort information om artisten.
Flagga som indikerar huruvida artisten spelar på festivalen eller inte.