Projektrapport

Johan Lilja
joli1407

DT060G
Mittuniversitetet

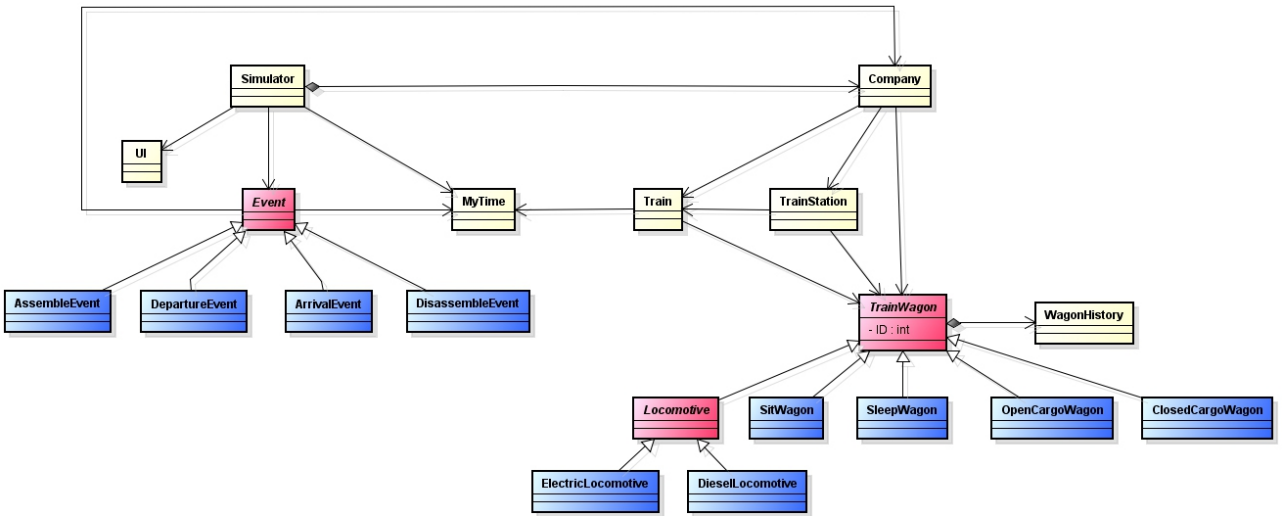
Projektbeskrivning

Projektets mål är att skapa en prototyp till ett simulationsverktyg som ska hjälpa tågbolag att strukturera sina rutter och tåg för att undvika förseningar och omöjliga avgångar. Simulationen är *discrete event-driven* och låter användaren styra stegandet i simulationen i egenvalt långa intervaller. Data till simulationen så som till exempel tågstrukturer, vagnar, tidtabeller, stationer och kartor läses in från textfiler. Underlättande av vidareutveckling var av hög prioritet vid projektets utförande.

Material

- Visual Studio 2013
- Windows 7 x64
- Astah community (UML)

Utförande



Programstrukturen är tänkt att separera företagets ägodelar, simulatorn som verkar baserat på dessa och programmets grafiska gränssnitt till en resonabel nivå för att underlätta vidareutveckling. Till exempel måste tågen godkännas av *Company* för att sättas ihop, då det i framtiden kan krävas andra faktorer för att tåget ska kunna gå, t.ex. tillgänglig personal.

Fordonshierarkin innehåller två interfaces, *TrainWagon* och *Locomotive*. Alla vagnar ärver *TrainWagon* medans loken ärver *Locomotive*, som i sin tur ärver *TrainWagon*. De olika klasserna i hierarkin innehåller olika datamedlemmar vars syfte är detsamma för alla deriverade klasser; exempelvis innehåller *TrainWagon* en vagns ID eftersom alla vagnar behöver ett sådant. Virtuella funktioner med kedjeanrop till de ärvda klasserna utnyttjas för att komma åt informationen tillhörande varje typ av vagn.

Klassen *Company* agerar ägare till alla tåg och stationer medans stationer och tåg i sin tur äger vagnar. I fallet med ägandet av vagnar så förändras detta under simulationens gång då vagnarna hela tiden ägs av objektet där de ”fysiskt” befinner sig under en viss tidpunkt, detta kan innebära antingen ett tåg eller en station. Att vidhålla detta ägandetänk är ämnat att sätta tydliga gränser för vad ett objekt kan och får göra med ett annat. Alla pekare eller referenser som lämnas ut från ett ägande objekt pekar till const och pekare är av typen *weak_ptr* för att förtydliga icke-ägande och undvika problem med cirkulärt ägande. Att tvinga alla manipulerande händelser på exempelvis ett tåg att ske genom ett anrop till *Company* är dock inte bara ett val baserat på inkapsling; Projektets mål är att skapa en prototyp till ett simulationsverktyg och detta innebär att vidareutveckling på stor skala bör tas i åtanke. Ett exempel på detta är när simulatorn ber ett tåg att försöka koppla ihop sig genom ett indirekt anrop till *Company*. I nästa fas av utvecklingen kan funktionalitet för kontroll av t.ex. spårtillgänglighet och tillgänglig personal lätt läggas till i *Company*.

Klassen *TrainStation* innehåller pekare till vagnar och tåg som befinner sig på stationen samt avstånd till andra stationer med förbindelser. Pekare till tågen är av typen *weak_ptr* då ägaransvaret tillhör *Company*. Vagnarna placeras i en *multimap* för att effektivt kunna undersökas och hämtas per typ av vagn även vid ett stort antal vagnar. Funktionalitet finns för att få information om stationen så som namn och avstånd till en annan station samt närvarande tåg och vagnar. Även här har vidareutveckling tagits i åtanke. För tillfället är avstånden symmetriska, det vill säga att det är lika långt från A till B som från B till A. Men detta kan komma att ändras genom t.ex. enkelriktning eller omläggning av räls. Därför har varje station sina egna värden för avstånd till andra stationer.

Klassen *Train* innehåller all information för ett specifikt tåg så som tågnummer, hastigheter, tider, planerad vagnstruktur samt påkopplade vagnar. Funktionalitet finns för att hämta information angående allt tåget innehåller.

Klassen *MyTime* används för att beskriva en tidpunkt eller en mängd så som längden av en försening. Operatorer är överlagrade för att fylla båda dessa syften med funktionalitet så som utskrift på 24 timmars-format, addition eller subtraktion av tider samt inmatning.

Klassen *WagonHistory* används för att enkelt logga när en vagns ägande överförs från tåg till station eller vice versa.

Klassen *Simulator* står för presentation av användarval samt driver själva simulationen framåt. Datamedlemmar består av simulationsinställningar, en prioritetskö med simulationens event samt pekare till ett grafisk gränssnitt för att presentera och efterfråga information till och från användaren. En huvudmeny tillåter användaren att sätta inställningar angående simulationstider, detaljnivåer för utskrift samt simulationsstyrning. Under simulationens gång erbjuder en simuleringsmeny möjligheter att få aktuell information om stationer, tåg, vagnar och den aktuella tidtabellen för alla tåg.

Klassen *UI* står för all ut och inmatning mellan programmet och användaren. Detta ger en separation mellan data och presentation (model/modelview) som gör gränssnittet enkelt att byta ut. Många utmatningsfunktioner i klassen tar pekare till const objekt som innehåller data som ska presenteras. Därmed kan gränssnittet självt formatera informationen som önskat på ett säkert sätt som förhindrar oönskade operationer.

Slutsatser

Det har varit intressant och utmanande att ta sig an ett lite större och friare arbete än de tidigare laborationerna. Personligen upplever jag att min förmåga att planera projekt har blivit bättre. Det framkom en del mindre brister i strukturen framåt slutet av projektet som jag gärna hade åtgärdat om det funnits mer tid. Bland annat ytterligare ett interface till tåg och stationer i form av något slags vagnbehållare då detta hade underlättat hämtning av information om vem som äger en viss vagn för tillfället. Jag anser dock att dessa funktionaliteter har implementerats med godtagbara kompromisser som enkelt kan bytas ut eller vidareutvecklas senare.

Vad gäller kursen i sig självt så skulle jag vilja göra samma påpekande som jag gjort i tidigare kurser på programmet. Jag anser att det bör schemaläggas tid med handledare över exempelvis adobe connect som är öppet för alla deltagare och där studenter kan titta in och diskutera eller få svar på diverse frågor. Forum är inget bra substitut för detta då det tar en väldig tid att hålla en konversation och det har även hänt att frågor går obesvarade som man kan se i den här kursens forum.