AARHUS UNIVERSITET

I4DAB - Databaser

Handin 4 – The Village Smartgrid

Group: 6

Christoffer Kjellerup Jakobsen - 201610457 (au569759)

Simon Møller Hindkær - 201609970 (au570382)

Marius Søren Linding Møller - 201610460 (au566306)

Tobias Damm-Henrichsen - 201611660 (au553636)

Content

Introduktion

I denne øvelse, skal der arbejdes med et Village Smart Grid, som skal kunne skabe et overblik over de el handler af kilowatt-timer, der er udført i en lille landsby. Ideen er at hver husstand eller virksomhed, allerede ved hvor stor en mængde strøm de enten producerer eller forbruger. Lad os sige at husstand A, producerer mere strøm end den forbruger og husstand B bruger mere strøm end den producerer. Den overskydende strøm i husstand A, kan så blive sendt, via et smartgrid, til husstand B så forbruget af EL er jævnet ud.

Dette overblik vil vi skabe, ved at designe og implementere 3 databaser, hhv. en SQL database der indeholder vores SmartGrid info(Information omkring den lille landsby), en anden SQL database, der indeholder Prosumer info (Information omkring de enkelte husstande i landsbyen) og til sidst en Azure dokumentdatabase, der indeholder Trader info (Information omkring brugt og produceret strøm, og hvordan der er blevet handlet strøm imellem de forskellige prosumers).

Til udfærdigelsen af vores løsningen, har vi fundet inspiration i egne tidligere afleveringer.

Fortolkning af den konceptuelle model og kravsspecifikationen

Formålet med denne opgave, er at designe og implementere tre databaser, med tilhørende REST API, der skal holde styr på køb og salg af strøm, mellem en række borgere og/eller virksomheder i en lille landsby i Danmark. De tre databaser (Prosumer-, SmartGrid- og Trader- databasen), skal indeholde følgende information:

- **Prosumer Info Databasen:**
  En prosumer vil i denne opgave blive anset som enten en husstand eller en virksomhed, som både producerer og forbruger strøm. Denne database indeholder oplysninger og en beskrivelse af de ”Prosumers” som er med i det givne Village Smart Grid. Dette kunne bl.a. være navnet, typen eller antallet af personer der indgår i en prosumer.

- **SmartGrid Info Databasen:**
  Vores ”Mini Smart Grid” vil i denne opgave blive anset som vores lille landsby, bestående af 33 husstande og 12 virksomheder/landbrug. Den vil i denne opgave også blive omtalt som ”Village Smart Grid”. Denne database kommer til at indeholde beskrivelsen af konfigurationen af vores ”Village Smart Grid”. Dette omfatter bl.a. et SmartMeter, en husstand, nogle elektriske enheder og et elstik.

- **Trader Info Databasen:**
  Dette er den vigtigste database i hele systemet for vores elselskab, idet den holder styr på hvorledes der er blevet handlet, hvordan der handles her og nu, og hvordan der skal handles i fremtiden. Denne database skal opdateres flere gange om dagen. Denne database kommer til at indeholde alle oplysninger der indebærer handel. Alt den vigtige information omkring mængden af strøm der er produceret og brugt, vil ligge i denne database.

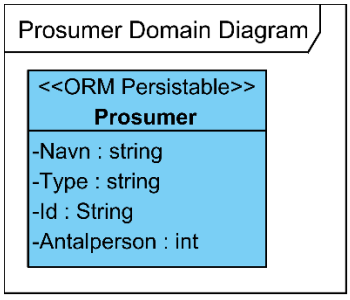
I disse tre databaser, har vi altså en hel del information, som ville fungere som et elselskabs afregningssystem, til at afregne salg og køb af strøm imellem vores prosumers. Hvis vi ser på landsbyen som et lukket samfund, der ikke havde mulighed for hverken at sende eller modtage strøm fra andre end dets egne prosumere, ville dette system optimere strømbruget ved at minimere spildstrøm så meget som muligt.

Vores komplette Mini Smart Grid, også kaldet for landsbyen, består af 33 husstande og 12 virksomheder. Vi kan betragte resten af Danmark som ”The National Smart Grid”. Hvis vi ser på vores Mini Smart Grid på nationalt plan, ville det agere som prosumer i ”The National Smart Grid”. Her ville det altså være muligt for vores lille landsby, at sende dets overskydende strøm ud til andre små landsbyer. Det ville hertil også være muligt for vores landsby at modtage strøm fra andre landsbyer (prosumers) i det nationale SmartGrid, hvis vores landsby skulle være i strømunderskud.

Domæne

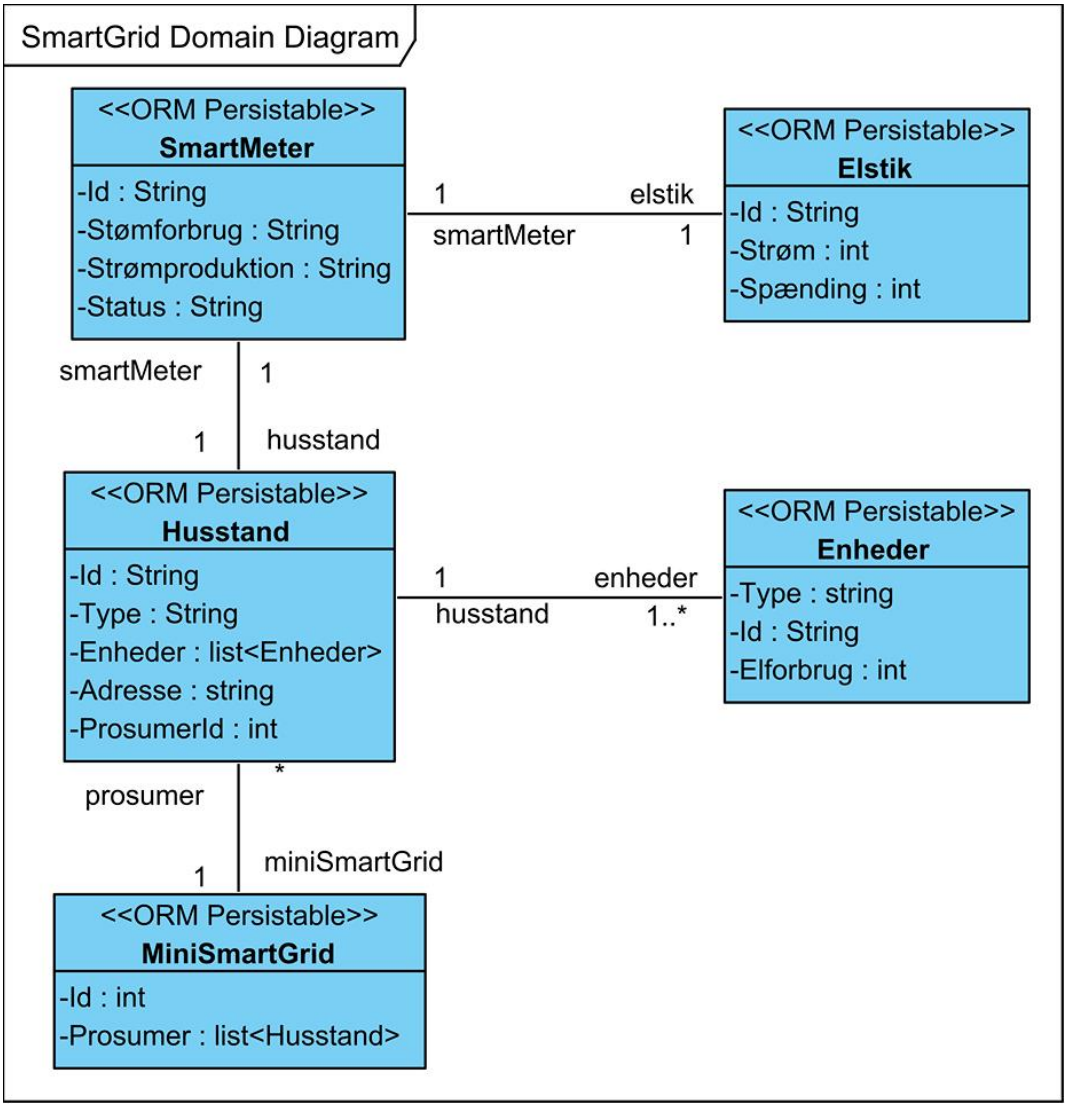
For at udvikle databaserne, er der først opstillet modeller, for at få et overblik over domænet.

Hvis vi starter ud med modellen i figur 1 for Prosumer Info Databasen først, kan vi se at den kun indeholder én entity, netop Prosumer. Vi har valgt at den væsentlige information i denne entity er hhv. navn, type og antallet af personer. Dette mener vi der giver os tilstrækkeligt med oplysninger, selvom databasen kommer til at være forholdsvis simpel. Denne database bliver implementeret som en codefirst relationel database.



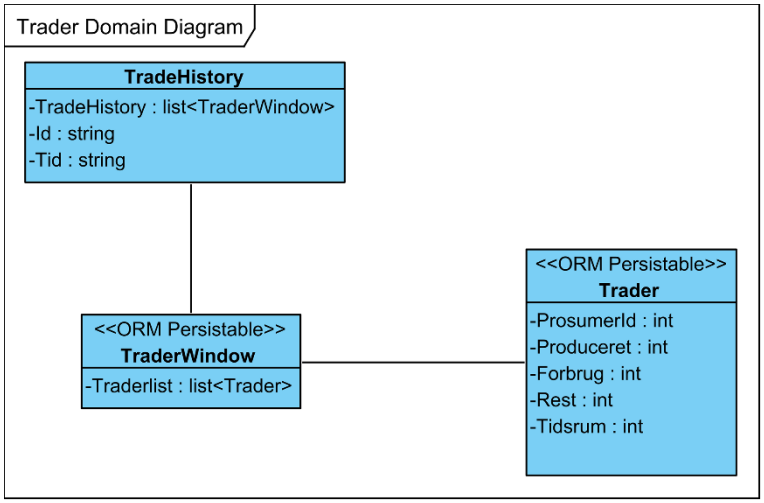
Figur 1 - Domænemodel for "Prosumer Info Databasen"

Herefter kommer vi til en database med mere kød på. Her er snak om domænemodellen for ”SmartGrid Info Databasen”, som kan ses i figur 2. Denne database indeholder 5 forskellige entities. Vi har først og fremmest en husstand der kan have en masse enheder. Disse enheder kan enten bruge eller genere strøm, så typen kunne f.eks. være et solcellepanel (som producerer strøm) eller et køleskab (som forbruger strøm). En husstand kan indeholde 1 eller mange enheder. Herudover har hver husstand også et SmartMeter. Ikke mere eller mindre end 1. Dette SmartMeter står for overvågelsen af hvor meget strøm en given husstand enten producerer eller forbruger og rapporterer informationen over internettet. Herudover kan vi også se at 1 MiniSmartGrid, med en række prosumers, kan have en eller mange husstande.



Figur 2 - Domænemodel for "SmartGrid Info Databasen"

Sidst men ikke mindst, kommer vi til modellen for Trade Info Databasen, som kan ses i figur 3. Dette er vores vigtigste database, idet den skal opdateres flere gange om dagen, og skal samle alt handelsinformationen. Vi har i denne database tre entities. Den består af en Trader entity, som indeholder id'et for vores prosumer, og kan på den måde få fat i hvor meget strøm hver enkelt prosumer har produceret eller brugt i et givent tidsrum.



Figur 3 - Domænemodel for "Trade Info Databasen"

Vi har valgt dette design, fordi vi mener at vi kan løse problemdomænet, ved at opstille vores databaser på denne måde. Prosumer domænet er forholdsvist simpelt, fordi vi også at den data vi inkluderer i databasen, ville være tilstrækkelig information for et elselskab, som jo er dem der skal anvende systemet.

Analyse

Der er stor forskel på brugen af vores databaser. Vores smartgrid Info Database, samt vores Prosumer Info Database, er en del mere statiske, end vores Trader Info Database. Tiden imellem at der skal skiftes ud i Prosumer og Smartgrid databaserne, er væsentlig større, end i vores Trader. Dette giver meget god mening, idet at der går længere tid før man evt. bygger nye huse, eller flytter fra sine huse.

Af den grund, har vi valgt at implementere Smartgrid Info databasen og Prosumer Info databasen, som SQL databaser.

Vores Trader Info database derimod, skal opdateres hver time, hvilket gør denne database meget kritisk. Det er derfor vigtigt at der bliver lavet backup af denne database, samt at sikkerheden på denne database er i top.

Af den grund, har vi valgt at implementere vores Trader Info Database som en Azure DocumentDB.

Til test af vores databaser, anvender vi et REST API. REST API webservice anvender HTTP request til CRUD-operationerne. Vi kan således tilgå/ændre/slette vores objekter fra Azure Cosmos databasen. Dette tillader brugerne at anmode om ressourcer som for eksempel kan fremkalde et svar formateret som JSON.

Vores REST API bruger et Repository pattern. Repository kan ses som en mediator mellem domænet og data mapping layers. Nogle af fordelene ved dette er at man slipper for at duplikere query logik. Videremere afkobler det vores applikation fra frameworks som entity framework.

Databasen der bruges til Trader Info databasen, er en Azure Cosmos DB, der er installeret op Microsofts Azure portal. Fra denne bruges URI og primary key, og der kan i øvrigt holdes øje med databasen i forbindelse med test. Her blev der først fastlagt en struktur for det dokument der skulle gemmes i databasen. Her kunne vi bruge begreberne fra Domain Driven Design Entity, Value Object, Aggregate og root samt inddrage begreberne omkring kardinalitet, connectivity, degree, deltagelse og navigering. Dette hjalp med at lave nogle antagelser, ift. multiplicitet og tilhørsforhold af de forskellige entiteter.

Konklusion

Det er lykkedes os at implementere tre selvstændige databaser, der ville kunne bruges af et el selskab til at styre el handler imellem prosumers i en lille landsby. Der er testet CRUD funktioner ved hjælp af et REST API, der er implementeret ved hjælp af et repository pattern, og det kan konkluderes at vores CRUD funktioner for alle tre databaser virker hensigtsmæssigt.

Det er dog ikke lykkedes os at implementere en samlet forbindelse imellem de tre databaser, men hver database fungerer som en Micro Service.

Muligheden for at duplikere dette ”Village SmartGrid” og anvende det på nationalt plan, er en mulighed. Dette kræver blot at der oprettes flere SmartGrids, som ville blive anset som prosumers i det nationale SmartGrids øjne.