Lars Kringelbach says:

Ok. Jeg mangler også lige at få ryddet køkkenet, men ungerne sover da.

p\_madsen@c.dk says:

Marjus er du der?

marjus@hotmail.com says:

hej, ja er her

p\_madsen@c.dk says:

Ok, Morten er først klar efter 20 engang, men Lars og jeg taler lidt om hvad vi gør. HAr du læst den linde strøm af mails?

marjus@hotmail.com says:

har ikke læst mail de sidste par timer

p\_madsen@c.dk says:

Ok. Jeg har varet jer på Henriks mail og kom til at sende den til ham. Har skrevet en ekstra mail også. Du bør nok lige læse de to, da jeg kan forstå på Lars han tænker lidt på de samme ting som jeg.

p\_madsen@c.dk says:

Det korte af det lange er at Henrik skriver vi skal fokusere på udvikling. Vi er kommet til den konklussion at det var også det som var planlagt for resten af projektet.

p\_madsen@c.dk says:

Vi diskuterer hvilke prototyper vi bør lave og om vi er nød til at tage udgangspunkt i drinkmixer som vi har skrevet i synopsis. (Vi har jo også skrevet at vi tilføjer funktionalitet hvis vi får lyst).

p\_madsen@c.dk says:

Vi talte desuden om det var en rimelig konklussion at DrinkMixer ikke er velegnet til modifikation og at vi laver vores eget i stedet.

marjus@hotmail.com says:

har nu læst mail. Jeg tror ikke vi får problemer med Henrik hvis vi fra nu af fokuserer på produktlinie og prototyper

marjus@hotmail.com says:

Hvad vil det sige at vi laver vores eget?

p\_madsen@c.dk says:

At vi ikke modificerer eksisterende klasser i DrinkMixer, men starter forfra på den enkelte feature.

p\_madsen@c.dk says:

Hvis vi vurderer at det ikke er muligt at bøje den del af drinkmixer nok.

p\_madsen@c.dk says:

Er det en funktionalitet drinkmixer ikke har, så giver det sig selv.

marjus@hotmail.com says:

ja det er jeg helt med på. Det er vel også fuldt "legalt" i udviklingen af en produktlinie, at man ikke direkte bruger koden fra et af grund produkterne, da de jo netop ikke er lavet som en produktlinie

Morten has been added to the conversation.

p\_madsen@c.dk says:

Det jeg talte med LArs om var at vi:

1) Valgte to? ting vi gerne vil lave prototype på

2) Valgte patterns hertil så det var allignet med hele arkitekturen

3) Lod to udvikle hver deres prototype og en tredie (elelr to?) arbejde med hovedarkitekturen

p\_madsen@c.dk says:

Hej, Moren er på

p\_madsen@c.dk says:

Morten ikke nogens mor

p\_madsen@c.dk says:

Hej Morten, hvordan læser du det henrik skriver. Er vi på ret spor hvis vi laver de planlagte aktiviteter med arkitekturen og et par? prototyper?

Morten says:

det er et godt spørgsmål. for mig ser det ud til at henrik forventer at vi i gennemgangen af teorien rent faktisk perspektiverer til casen, og ikke ved at gennemgå teori, og så efterfølgende lave en case. Jeg forestiller mig at vi kan ommøblere lidt hvis vi skriver lidt om casen undervejs i teorien, og benytter konkrete patterns i sammenhæng med prototyper til at undebygge de dele der handler om

Morten says:

det, og yderligere relatere direkte til rekonstruktionen og processen i denne, i teoriafsnittet om emnet, så jeg forestiller mig at vi kan klare os ved at ændre layout, og så være grundige i vores praktiske tilgang

p\_madsen@c.dk says:

Hmmm, du mener at teorien skal splittes op og fordeles rundt i dokumentet?

Morten says:

tja, man kan se på det som man vil, jeg ville nok formulere det som at vi drysser lidt praktisk erfaring ind i de allerede skrevne afsnit

marjus@hotmail.com says:

jeg læser det mere som at vi ikke får så meget ud af at skrive om teorien, fordi det er vi jo testet i, men at vi skal fokusere på vores egne erfaringer

Morten says:

så vi på den måde relaterer teoretikernes skrevne ord med vores egne erfaringer, og på den måde kvalificerer debatten

marjus@hotmail.com says:

ja enig i det sidste

p\_madsen@c.dk says:

Nogen der kan oversætte det til noget praktisk. Hvad er det præcis i mener vi skal gøre?

Morten says:

jeg læser det lige igen, og måske har i ret, at vi ikke skal blande det alligevel, men huske at fokusere på de praktiske erfaringer vi har fået med vores case. Jeg kunne sagtens skrive noget om det omfanget af processen for rekonstruktion. Tror i han mener at vi kun skal beskrive det teori som vi rent faktrisk anvender senere, og ikke andre emner?

Lars Kringelbach says:

Jeg sådan kunne jeg godt forstå det. I hvert fald fokusere på den del af teorien som vi anvender.

Lars Kringelbach says:

Men jeg synes måske vi drager en masse konklusioner ud fra et par enkelte liniers kommentarer. Inden vi rykker det hele rundt, synes jeg det kunne være rart med en kommentar mere fra Henrik.

marjus@hotmail.com says:

jeg synes bare vi nu skal fokusere på prototyperne, få noget erfaring

marjus@hotmail.com says:

ja og vente på uddybning fra Henrik

Morten says:

ja, jeg trækker min kommentar tilbage omkring omstrukturering af dokumentet

Lars Kringelbach says:

Vi kan godt lægge en plan for det videre arbejde, men bare vente lidt med at flytte rundt på ting.

Lars Kringelbach says:

Jeg tror egentlig vi er rimelig enige om hvad der skal gøres, spørgsmålet er bare hvor det skal stå.

Lars Kringelbach says:

Peter: Enig

Morten says:

jeg synes stadig vi skal udføre en arkitektonisk refaktorering og et par prototyper, samt arkitektonisk dokumentation af disse. Refaktoreringer bør være funderet i tactics og styles som vi skriver om

p\_madsen@c.dk says:

Jeg vil give Morten ret i at det kan læses som at vi skal passe på med at skrive om al den teori vi kender, men holde os til noget som er relevant.

p\_madsen@c.dk says:

Morten: Enig.

p\_madsen@c.dk says:

Hvordan får vi hustigst muligt gang i nogle prototyper?

p\_madsen@c.dk says:

Jeg tænkte noget storage som oplagt?

Morten says:

det kunne være interessant at lave en prototype hvor vi tager persisten, laver et repository, og så laver en rest service til at gemme med, og en flad fils implementation for at se at det kan lade sig gøre

p\_madsen@c.dk says:

Morten, der kom lige nogle software begreber ind jeg ikke er bekendt med? Hvad er en rest service?

Morten says:

REST er den protokol som blev brugt i H1 som er parameteroverført via URL encoding. Detblev brugt i radiator service eksemplet

p\_madsen@c.dk says:

Ok, hvorfor er det interessant? Ikke for at kritisere, men som sanity check?

marjus@hotmail.com says:

understøtter det noget af variationspunkterne?

Morten says:

det er en service hvor POST, PUT, DELETE og kan ikke huske den sidste http operation, har hver sin semantiske betydning

Morten says:

fordi vi i vores feature nedbrydning har påersistens igennem web service, og igennem flad fil

p\_madsen@c.dk says:

Så ville du bruge REST prototkol da du kan lave en web service som understøtter det?

Morten says:

dermed er det interessant om vores abstraktino af repository passer på bege teknologier, eller om de er asymmetriske i deres opførsel og interface

p\_madsen@c.dk says:

Og så lave frontends til flad fil og SQL database?

Morten says:

det er fordi det er simplere end at lave en SOAP web service, vi kan stjæle noget af deres kode fra H1

p\_madsen@c.dk says:

Som også forstå den protokol?

p\_madsen@c.dk says:

Vil man så kommunikere via http med fil og sql database?

Morten says:

REST er i sig selv uinteressant, det der er interessant er at se om det vi skriver i feature nedbrydning omkring remote og local persistence, vil være en oplagt abstraktion at tage op som prototype.

Morten says:

hvor vidt servicen gemmer i en flad fil er vel egentligt ligemeget, man kan naturligvis tale om skallerbahed her, men for prototypens eksempel er det vel ligemeg om servicen gemmer i en database

p\_madsen@c.dk says:

Altså en arkitektonisk prototype som understøtter de variationer vi har under storage, eller taler du kun om loacal / remote?

Morten says:

http://en.wikipedia.org/wiki/REST

Morten says:

enig peter, spørgsmålet er om vi vil lave to eller tre persistensprototyper, og dermed tænker jeg at de to implementationer der ligger længst fra hinanden er flad fil, og web service. Alternativt skal vi lave en DB, og en service prototype, da DrinkMixer jo tydeligt viser hvordan man kan lave en implementation med flad fil

marjus@hotmail.com says:

jeg tror ikke jeg er helt med, hvad siger du konkret er formålet med den prototype?

p\_madsen@c.dk says:

Er det vi skal ikke at lave en arkitektur som understøtter alle dele, inkl. at benytte det eksisterende fil interface?

Morten says:

at vise understøttelse for vores variationer igennem repository tactic

marjus@hotmail.com says:

jeg er enig med Peter, synes også at prototypen skal vise hele arkitekturen, og ikke kun at afprøve en tactic

p\_madsen@c.dk says:

Det er vel som udgangspunkt relevant at undersøge enhver knude i vores feature træ?

Morten says:

men hvis vi laver en hel ny arkitektur så vil en prototype der dækker det hele jo være en komplet implementation. Formålet med prototypen er at afdække arkitektoniske risk. Hvis vi vælger repository til at persistere med, så er der en risiko for at der er assymetri imellem behovene for interfacet for en serviceimplementation af repository og en DB, eller fladfils implementation.

Morten says:

dermed bliver det interessant at lave en prototype der kan afdække denne risk.

p\_madsen@c.dk says:

Morten. Jeg skal lige tilbage til adam og eva. Hvis vi anvender repository? Er det en taktisk?

p\_madsen@c.dk says:

taktik?

Morten says:

peter du har ret, men vi skal fokusere på noget som indvirker på arkitekturen. ikke alle features har dog arkitektonisk relevans, nogle er bare ektensions, hvor andre kan indvirke på performance eller skallerbarhed osv. Jo repository er en taktik, men taktikken manifesterer sig jo i nogle repository interfaces i koden, og der er det vigtigt at at finde det rigtige niveau

p\_madsen@c.dk says:

Grunden til jeg spørger er simpel. jeg kender den ikke og kan ikke lige finde den i bogen. KAn du lige pege mig det rigtige sted hen?

marjus@hotmail.com says:

den er nævnt under modifiability

Morten says:

to sek, der er et eksempel i et af hans slides

Lars Kringelbach has left the conversation.

Morten says:

architecture redesign slides side 15

Lars Kringelbach has been added to the conversation.

Morten says:

det pattern siger i praksis ikke andet end at der er delt data af et antal klienter. Men for at lave en kodeabstraktion over emnet er der en arkitektonisk risk, hvis vi er uenige, så lad os bare finde et andet emne

p\_madsen@c.dk says:

Delt af klienter. Så er det den feature at man kan dele scores mv. du vil undersøge?

marjus@hotmail.com says:

jeg synes emnet er fint nok, havde bare forestillet mig at en arkitektonisk prototype var for hele arkitekturen, men det er måske forkert

Morten says:

alle opskrifter og data som er remote persisteret er jo delt data. man kan så deployere systemet så det ikke er delt, men det er en detalje

Morten says:

hvis vi kigger tilbage på deres prototyper, så var de meget forsimplede, og addresserede kun en eller to risk

p\_madsen@c.dk says:

Umiddelbart havde jeg tænkt at det var en arkitektur som kunne rumme flat file, SQL database og Web service (REST?) vi skulle prototype, men har ikke rigtigt noget godt argument for det.

Morten says:

muligvis hvis man laver en evolutinoary prototype vil man kigge på flere risk, og addressere en større del af arkitekturen

p\_madsen@c.dk says:

Marjus: En arkitektonisk prototype under søger et elelr flere aspects ved en arkitektur ikke hele arkitekturen i alle detaljer

Morten says:

vi har jo brug for noget modeæ, og noget persisten for at lave denne prototype, så er der vel kun GUI tilbage

Morten says:

ca...

Lars Kringelbach says:

Enig. Jeg synes også vi skal holde os til at lave et par prototyper som undersøger en enkelt risk.

p\_madsen@c.dk says:

Er der noget sjov ved at forfølge print?

Lars Kringelbach says:

Morten: Ikke forstået. Hvilken prototype henviser du til?

Morten says:

ok. Jeg mener egentligt at føromtalte emne er rimeligt repræsentativt, men der er jo en del på spil, med både database og noget REST service, så det er nok ikke trivielt at lave den. På den anden side betyder det nok også at den er mere end aktuel at få lavet

Morten says:

Lars: en prototype på at lave et repository interface, og så implementere det som en flad fil, eller en database, og så en implementation som er en REST service, for at undersøge tactic-repository, i forhold til vores variationer med remote service persistens, og lokal fladfilspersistens.

p\_madsen@c.dk says:

Morten, hvis vi ser på vores feature model, hvor meget er så indeholdt i den prototype?

Lars Kringelbach says:

Ok, men vi behøver vel ikke en masse model for at lave den.

Morten says:

næh, me

Morten says:

n en simpel drink model er jo nødvendig for at se om det virker.

Lars Kringelbach says:

Men spørgsmålet er også om vi behøver at lave en faktisk implementation af både SQL og REST.

p\_madsen@c.dk says:

Men vi kan vel vælge kun at teste med den eksisterende model fra DrinkMixer?

Lars Kringelbach says:

Vi kan evt. kopiere fil-tilgangen fra DrinkMixer og tilpasse den til et repository-interface, lave en SQL tilgang, og så bare snakke om REST.

Morten says:

jeg synes egentligt at database persistens er uinteressant fordi man typisk vil benytte sådan noget som NHibernate, til mapning af data. Alternativt kunne vi lave en prototype med NHibernate på MySQL, og en med brug af en OO-database for at afdække om OO-database er en farbar vej

Lars Kringelbach says:

Jeg tror egentlig vi får mere ud af at lave flere små prototyper, end at kode en masse kompleks funktionalitet.

Morten says:

lars: formålet med service prototypen er at det er der der er den største risk i forhold til muligheden for assymetriske behov for information konfiguration, serialisereing og deserialisering mm.

p\_madsen@c.dk says:

Morten: Hvad mener du med muligheden for assymetriske behov for information konfiguration, serialisereing og deserialisering

Morten says:

men vi kan da knalde en OO-database på, da de jo ikke skal mappe data. det er jo i virkeligheden bare en avanceret fladfilsimplementation

Morten says:

et eksempel kunne være at der i et repository finde en Load og en Store funktion. Hvordan sørges der for serialiseringe over HTTP, og sikres at data kan gemmes og hentes igen. Hvordan konfigureres Net adgang i forhold til flad fil, hvilke behov er der for sikkerhed,

Morten says:

osv. osv.

Lars Kringelbach says:

Er pointen med at bruge en SQL database ikke også at flere kan tilgå den på én gang? Så vi kan have flere klienter på samtidig.

p\_madsen@c.dk says:

Lars: Ikke i udgaverne med lokal storage, men i udgaverne med remote storage er det korrekt.

p\_madsen@c.dk says:

Morten: Det er tydeligvis et emne du ved meget mere om end mig, så har svært ved at komme med kvalificaret input til den prototype, men det lyder spændende og som om der er relevante problematikker

Lars Kringelbach says:

Ja, det er klart. Men SQL kan være både remote og lokal. Jeg forestiller mig ikke den store nødvendighed for at bruge en SQL DB, hvis man kun har én bruger på, der kan en flad fil vel være fint nok.

p\_madsen@c.dk says:

Ja, men vi har i vores feature træ besluttet at man kan køer een SQL database lokalt.

p\_madsen@c.dk says:

Den kan vi naturligvis pille af igen.

Morten says:

hmm. har vi skrevet SQL , eller database

p\_madsen@c.dk says:

Begge dele

p\_madsen@c.dk says:

SQl database

Morten says:

lol

p\_madsen@c.dk says:

Hvilket vel er det korrekte ord? En database som man kommuniker med med sproget SQL?

Morten says:

ja, med mindre vi benytter en prototype til at undersøge om det kan lade sig gære at bruge en objektorientereO

Lars Kringelbach says:

Ja, det ser fint ud.

Morten says:

(OO)- database

Lars Kringelbach says:

Men hvad mener vi med lokalt? Hvis det fx. er et værksted med flere klienter på, er det så local eller remote storage?

Morten says:

http://www.db4o.com/Default.aspx?AspxAutoDetectCookieSupport=1

p\_madsen@c.dk says:

Morten, er det ikke ret langt væk fra vores udgangspunkt at lave DrinkMixer til en produktlinie?

Morten says:

så er det vel remote for klienten, det handler jo om deployment

Lars Kringelbach says:

Hvis man har brug for større sikkerhed mht. fejl i data, så er en database nok også bedre end en flad fil.

Lars Kringelbach says:

Ja.

marjus@hotmail.com says:

de performance kvaliteter vi indtil videre har defineret antager vel at det er remote (1000 samtidige brugere)

Morten says:

vi behøver altså herller ikke lave en prototype om emnet

Lars Kringelbach says:

Jeg kan ikke se vi har behov for at hive sådan en teknologi ind. Jeg synes det handler om at lave noget hvor vi kan udskifte storage-laget. Det er ikke vigtigt hvilke teknologier vi kan bruge til storage.

p\_madsen@c.dk says:

Jeg tror også vi bliver dømt på arkitektur og de anvendte patterns ikke på teknologier

Morten says:

næh, men hvis vi bruger en sql database, så skal vi generere scripts, vedlægge installationsfiler og meget mere. Hvis vi bruger en OO-databasem så er det en Jar fil der skal vedlægges, og så er man kørende. Den gemmer også i en flad fil, men har understøttelse for transaktioner, søgning og alt muligt andet som SQL-DBMS også understøtter-

Morten says:

jeg legede lidt med den for nogle år siden for sjov, og fra .NET var det så nemt som en pind at komme igang med

Lars Kringelbach says:

Ok, det kan godt være et argument

Lars Kringelbach says:

Er der ikke også en SQL lite, der er lige til at gå til

Morten says:

det er udelikkende et spørgsmål om vi kan få abstraktionerne til at passe. hvis vi ikke benytter en Service teknologi kan vi ikke udtale os kvalificeret omkring impact for at lave repository over services, og det samme gør sig gældende omkring DBMS

Lars Kringelbach says:

Ikke så god til remote, men jeg går ud fra den nemt kan udskiftes med en rigtig SQL db

p\_madsen@c.dk says:

Jeg synes det lyder helt rimeligt at skifte SQL ud med OO hvis vi mener det giver mindre arbejde, men vi skal bare sikre fokus er på det design der gør vi kan skifte storage laget ud, ikke på det faktum vi bruger en OO database. Måske kunne vinklen så være hvordan man giver objekter til en fald fil?

Morten says:

hvad er ikke godt remote? DB4O? det kan sagtens være

p\_madsen@c.dk says:

flad fil

Morten says:

jeg kender ikke så meget til concurrency på DB4O, men de har nogle benchmarks og whitepapers på deres hjemmeside

marjus@hotmail.com says:

hvis vi bruger en database facade er det ligemeget hvad ligger bag facaden

Morten says:

jep

Lars Kringelbach says:

Nej SQL lite er vist ikke godt til remote (jeg skal lige finde ud af om det er det rigtige navn jeg skriver).

Morten says:

men der er også stor forskel på om man selv skal bygge concurrency ind i facaden, eller om DBMS håndterer det for en

Morten says:

læs tre minutter her: http://www.db4o.com/s/javadb.aspx

Lars Kringelbach says:

tiny (600K!), \*zero install\*, transactional database

Lars Kringelbach says:

SQLite for .NET (måske er det den, jeg har kun hørt om den)

marjus@hotmail.com says:

jo men man kan vel bare antage det man finder passende (så længe det er realistisk) og så er det man har antaget constraints på de mulige valg af teknologi

Morten says:

Marjus det er jeg enig i, vi skal bare sikre os der rent faktisk er realistisk

p\_madsen@c.dk says:

Morten: Hvis vi vælger den database, hvad er det så vi vil undersøge med prototypen. At vi kan lave en facade som er i stand til at gemme dem direkte i OO DB, gemme dem i en flad fil og sende dem over http (de to sidste kræver serializering)?

Morten says:

peter: vi skal vel vælge hvlike heste vi spiller på. enten laver vi to prototyper hvo vi viser eksempelvis REST overfor DB eller flad fil, eller vi laver en prototype af hvert af vores features for persisten

Morten says:

s

Lars Kringelbach says:

Hov, nu skriver du igen REST overfor DB. Det handler om at vi kan udskifte laget. Dvs. de implementationer, og det er fuldstændig lige gyldig hvilke

Lars Kringelbach says:

Bare lige så vi er enige

p\_madsen@c.dk says:

Hvad er det præcis prototyperne kan tage et objekt? og persistere det? Hhv. til flad fil, SQL og over HTTP?

Lars Kringelbach says:

Det lyder fornuftigt.

Morten says:

jep, men hvis vi laver en webservice prototype, må vi vælge en teknologi at lege med, og de oplagte er REST eller SOAP

Morten says:

ja

Lars Kringelbach says:

Ja, selvfølgelig

p\_madsen@c.dk says:

Rest er den simpleste, korrekt?

Morten says:

her er et simpelt eksempel på hvordan den OO database bruges. det er nu ret simplet: http://developer.db4o.com/Resources/view.aspx/Formula\_One\_Tutorial

Lars Kringelbach says:

For min skyld kunne det også bare være flad fil og DB

Lars Kringelbach says:

Fx. DB4O

p\_madsen@c.dk says:

Husk vi skal kunne argumentere for hvorfor vi har valgt at lave netop den prototype

Morten says:

Rest er bare http://addresse/functionsNave?Parameter1=ddd&Parameter2=bbb, og så bestemmer headeren i HTML dokumentet hvor der står om det er PUT, GET, POST eller DELETE, hvilken type server operation der er tale om. REST er hvist bygget til CRUD services

Morten says:

CRUD er jo netop det vi vil.

p\_madsen@c.dk says:

Ok, så det kræver at vores modelobjekter er serializable?

Morten says:

Næh, det kræver at vi kan overføre værdierne af et objekt som parametre

Morten says:

og eventuelt at man kan kofigurere hvordan data gemmes ud fra funktinosnavn eller lignenden

Lars Kringelbach says:

CRUD: Chopper Riding Urban Dwellers

Lars Kringelbach says:

www.crud.org

Morten says:

Create

Read

Update

Delete

Lars Kringelbach says:

(Sorry, klokken er mange for en søndag aften)

Lars Kringelbach says:

Ja, den fandt jeg også, men det var ikke sjovt.

p\_madsen@c.dk says:

En løsning kunne være at serialisere over REST, right? Jeg med på man kan gøre andre ting (fx. bruge reflection til at finde alle get metoder osv.)

Lars Kringelbach says:

Jeg har tilføjet en konkret prototype på wikien (http://code.google.com/p/saip2008/wiki/Backlog?ts=1222629310&updated=Backlog).

p\_madsen@c.dk says:

Er der nogen der er stand til at beskrive præcis hvad der skal laves.

p\_madsen@c.dk says:

Ups. Det var du åbenbart Lars.

Morten says:

men hvad der det der skal serialiseres, REST er jo URL-overførelse af parametre, hvor netop PUT, GET, POST og DELETE giver semantikken af operationen

Lars Kringelbach says:

Det jeg har skrevet på wikien er nok lidt for tyndt til at fortælle \_præcist\_ hvad der skal laves

p\_madsen@c.dk says:

Er der nogen der kan det, så vi kan komme videre til næste prototype?

Lars Kringelbach says:

Vi skal vel have defineret et interface for håndtering af data.

Lars Kringelbach says:

Så skal vi have lavet en skal der kan tilgå interfacet og konstueres med forskellige implementationer.

Lars Kringelbach says:

Og til slut skal der laves 1-3 implementationer af interfacet.

p\_madsen@c.dk says:

LArs: En hvad?

p\_madsen@c.dk says:

En prototype?

p\_madsen@c.dk says:

Er det nok til at at en kan tage opgaven?

Lars Kringelbach says:

Ja, det var en prototype jeg tænkte på

Morten says:

godt spørgsmål. Vi har vel også en opgave i at udvikle og dokumentere en ny arkitektur, eller snakker vi kun om prototyper nu

p\_madsen@c.dk says:

Det hænger vel sammen. Det naturlige workflow i dette tilfælde ville vel være at udvikle arkitekturen og så undersøge spcifikke tvilsområder nærmere?

Lars Kringelbach says:

Mit forslag er at vi lavet en exploratory prototype "før" vi laver arkitektur design til at undersøge hvordan vi kan designe det, og så laver en vi en experimental efter til at eftervise designet (gerne inden for et andet emne).

Lars Kringelbach says:

Hvis vi laver exploratory burde vi dog nok "explore" fx. to forskellige arkitekturer

Morten says:

jep, men eftersom vi ikke nødvendigvis behøver at definere metoder, og har lagt os fast på at benytte et repository, kan vi jo udvikle et repository interface igennem prototypen, og dokumentere sekvensen bagefter

p\_madsen@c.dk says:

Er den prototype vi har specificeret? nu en kandidat til en exploratory?

Morten says:

det syntes jeg lyder rimeligt

Lars Kringelbach says:

Jeg sidder faktisk og bliver lidt i tvivl. Vi undersøger i hvert fald et design.

Lars Kringelbach says:

Men det kunne lige så godt falde ind under experimental hvor vi tester en arkitektur.

p\_madsen@c.dk says:

Men det driver ikke nødvendigvis arkitekturen med mindre vi vil undersøge om det overhovedet er muligt at lave systemet som både local og remote?

Morten says:

hvem gemmer den her diskussion, så vi har adgang til den senere, den indeholder jo en del latent viden

p\_madsen@c.dk says:

Ind på Wikien med den som en note?

Morten says:

ja jeg mener det vi undersøger er om vi kan lave en service implementation og en anden implementation igennem det samme interface, som er relativt abstrakt

Lars Kringelbach says:

Exploratory prototypes used to clarify requirements

to the architecture together with its stakeholders, to explore

aspects of the target system, and used to discuss

alternative architectural solutions

p\_madsen@c.dk says:

Ok. vi har defineret en prototype. Vi har defineret en opgave med at definere hele arkitekturen. Så vi mangler en eller to prototyper mere?

Lars Kringelbach says:

Dvs. hvis vi kunne finde på en anden arkitektur til persitens, så ville det klart være exploratory.

Lars Kringelbach says:

Experimental prototypes used to gauge the adequacy

of a proposed architecture, or details hereof, before investing

in a large-scale implementation of it. Experimental

prototypes are typically used to measure software

architecture qualities; e.g., to make quantitative

measurement of performance or portability.

Lars Kringelbach says:

Hvis vi bare laver denne prototype mht. persistens, så mener jeg det er en experimental

Lars Kringelbach says:

Jeg skal nok smide det på wikien.

Morten says:

min hjerne slår snart fra drenge, tror snart vi skal lave et wrap up hvad angår mig.

Lars Kringelbach says:

Også her.

marjus@hotmail.com says:

hvad skal vi måle på hvis det er en experimental?

Lars Kringelbach says:

Jeg synes Peters sidste besked var et godt wrap up.

Morten says:

adequacy, er det vi er i tvivl om ikke om det rent faktisk kan lade sig gøre i det hele tage?

Lars Kringelbach says:

Vi skal måle op mod vores kvalitetsscenarier.

Lars Kringelbach says:

Nemlig

Morten says:

så vi kan kigge på skallerbarheden

Lars Kringelbach says:

Vi har nogle simple scenarier der siger at vi skal kunne udskifte forskellige dele af systemet. Det skal vi eftervise.

Lars Kringelbach says:

Det kunne være fedt hvis vi kunne finde et andet område, hvor vi kan lave en exploratory, dvs. implementation af to forskellige arkitekturer der opfylder det samme.

Morten says:

men det kan blive relativt avanceret at implementere et scenarie hvor 1000 klienter banker løs på en server med et varierende interval, og datamængder, samt at de ikke er placeret på samme maskine for at få netværkstrafik med i billeder

p\_madsen@c.dk says:

Vi har defineret en prototype omkring storage. Det er er tilbage i feature model er GUI, Guides, Print, Pricing og Search

Lars Kringelbach says:

Nemlig, derfor undlader vi nok at kigge på vores performance-scenarier

p\_madsen@c.dk says:

Morten: Jeg tror også den prototype kan blive ret voldsom, hvis den skal være retvisende. V

Morten says:

måske vi skal synge en sang om hvorfor vi ikke tager det med, altså pga. kompleksiteten, og dermed at vi understreger vi har styr på det, men har valgt anderledes

p\_madsen@c.dk says:

Hvad med pricing, bliver det bare et strategy pattern og dermed uinteressant (PaSOOS)

Lars Kringelbach says:

Tja, men der er mange scenarier vi ikke håndterer i projektet.

Morten says:

der er noget omkring varierende domæner, som kan være interessant at kigge på. Hvordan skal vi persistere og loade data fra forskelllige domæner i systemet uden at skulle lave en komplet ny service struktur, og dB layout for hvert produkt

Lars Kringelbach says:

Jeg tror godt at pricing kunne være relevant

Morten says:

oki

Morten says:

en eller anden form for rule enging

Morten says:

engine

Morten says:

?

p\_madsen@c.dk says:

Pricing er lige til at gå til. Det er et strategy pattern

Lars Kringelbach says:

Hmm... man kunne selvfølgelig også overveje at opstille en prototype, hvor vi kaster en masse forespørgsler efter vores flade fil impl. kontra vores DB impl. og se om der er forskel.

Morten says:

linker du lige tak

Morten says:

så kan vi bygge videre på den oprindelige prototype, og det ville være evolutionary. Dermed leger vi videre med prototype teorien

Lars Kringelbach says:

Det kunne vise det med at forskellige produkter kan have forskellige kvaliteter (vi kan evt. flytte en DB over på en anden stor server der kan klare et større pres)

p\_madsen@c.dk says:

Hvis vi skal kunne skifte ud dynamisk er det, men er ikke sikker på det er fuldt dækkende

p\_madsen@c.dk says:

http://en.wikipedia.org/wiki/Strategy\_pattern

p\_madsen@c.dk says:

The strategy pattern is intended to provide a means to define a family of algorithms, encapsulate each one as an object, and make them interchangeable. The strategy pattern lets the algorithms vary independently from clients that use them.

p\_madsen@c.dk says:

Lyder rigtigt

Lars Kringelbach says:

Jeps

p\_madsen@c.dk says:

Og Henrik nævnte det da Lars og jeg talte med ham, så han er nok enig

Morten says:

hehe

Lars Kringelbach says:

p\_madsen@c.dk says:

Men Morten har ret i at man kunne bruge en rules engine til at definere de enkelte strategies

Morten says:

men spørgsmålet er om vi kun understøtter flad skat ligesom moms på 25%, eller om vi understøtter forskellige skatter på forskellige produkter i samme kørende instands, således at en tandrem beskattes med 10% og en bilradio beskattes med 15% fordi det er elektronik

Lars Kringelbach says:

Vi understøtter klart den sidste model

p\_madsen@c.dk says:

Pricing modulet tager som udgangspunkt et item? Hvis et item har en type så er det du skriver også muligt at gøre

p\_madsen@c.dk says:

Skal vi putte det på som mulig prototype, og hvis ja, hvad er det for en risk vi vil undersøge?

p\_madsen@c.dk says:

At vi er i stand til at køre med vidt forskellige modeller for prisberegning?

Lars Kringelbach says:

Jep, det lyder godt.

Lars Kringelbach says:

Har vi evt. to løsninger på det problem, så vi kan "explore" det lidt?

p\_madsen@c.dk says:

Man kan jo iffe sig ud af det?

Morten says:

En alternativ løsning kunne være abstract factory, hvor man kunne give produkttype med, og som så kan generere din strategy model for dig, og så kan man skifte factory ud for hvert enkelt land, og derees special beskatninger

Morten says:

man kunne også lave en prototype omkring features driven at load time, via dependency injection framework så som Sprint, eller castle

Lars Kringelbach says:

Men det behøver jo ikke kun være beskatninger der varierer. Det kan lige så godt være tidspunkt på dagen, eller mængde rabatter osv.

Morten says:

Spring

Morten says:

ja nemlig ja, det havde jeg elt glemt

p\_madsen@c.dk says:

Men det giver god mening at visse strategier er landespcifikke. I Dubai er det fx. weekend fredag og lørdag

marjus@hotmail.com says:

kan vi lave en exploratory prototype for sådan en prismodel uden at implementere for meget funktionalitet?

Morten says:

og alkohol må man ikke sælge, så det kan ikke beskattes

p\_madsen@c.dk says:

Men ved ikke om de mangler en drinks guide

Lars Kringelbach says:

Jeg kan ikke lige rumme det i hovedet nu, men det skal vi vel kunne.

p\_madsen@c.dk says:

Morten: Det må man bestemt gerne

Morten says:

nå, må man også gerne drikke det

p\_madsen@c.dk says:

Jeg har da ofte købt noget OG drukket det

p\_madsen@c.dk says:

p\_madsen@c.dk says:

Nå tilbage på sporet. Vi er vidst alle trætte

Lars Kringelbach says:

(Vi skal vist ikke vente for meget hjælp fra Peter i næste uge

Lars Kringelbach says:

)

p\_madsen@c.dk says:

Jo i næste uge

Morten says:

hvornår er deadline for opgaven

p\_madsen@c.dk says:

Rejser på fredag

p\_madsen@c.dk says:

Fredag d. 10

Lars Kringelbach says:

Ja okay, næste uge igen

p\_madsen@c.dk says:

er deadline

Morten says:

så vi har to uger fra nu

p\_madsen@c.dk says:

Ja

Morten says:

oki

p\_madsen@c.dk says:

Målet med i dag må være at vi har fire relevante opgaver, en til os hver?

Morten says:

oki

p\_madsen@c.dk says:

Vi har defineret storage prototype. Den overordnede arkitektur. Og en pricing prototype?

p\_madsen@c.dk says:

Så vi mangler en?

p\_madsen@c.dk says:

Er det ikke lidt volsomt at lave en GUI prototype?

Morten says:

jo,

Lars Kringelbach says:

jo

Morten says:

men mener egenglit

Lars Kringelbach says:

Jeg synes ikke vi skal lave flere prototyper. Jeg synes vi har nok.

Morten says:

egentligt at den sotrage prototype er ret stor

p\_madsen@c.dk says:

Ok. Kan storage prototypen deles på to hænder?

Morten says:

det er både service, OODB,

Morten says:

og flad fil, med mindre man slår flad fil og DB sammen, og siger at OODB også er flad fil

p\_madsen@c.dk says:

Er der ikke en del forskel på at give noet et objekt og selv søre for det bliver gemt i en flad fil?

Lars Kringelbach says:

Jeg vil tro at alle prototyperne kan deles, når først det initielle arbejde er gjort.

p\_madsen@c.dk says:

Så det første arbejde går på at skrive lidt tekst og tegne et diagram eller to?

marjus@hotmail.com says:

synes vi bare skal bruge service og OODB, tror det bliver begrænset hvad vi lærer ved at lave flad fil også

Morten says:

ja, og oodb er også flad fil, bare lidt mere avanceret

Lars Kringelbach says:

Bortset fra at vi så slipper for to "komplekse" implementationer, som egentlig ikke viser mere end en flad fil

p\_madsen@c.dk says:

Morten du talte for at slippe for prototypekodningen da vi holdt møde. Er du stadig af den holdning og helst vil den overordnede arkitektur?

Morten says:

jegg mener bestem at service impl viser noget andet end flad fil, men man kan overvejer om man skal bruge SOAP, fordi det jo er et objektorienteret system

Morten says:

jeg er sgu lidt ligelgad, bare vi kører løs

Morten says:

jeg kan sagtens lave noget prototype af den OODB

Lars Kringelbach says:

Jeg har ikke noget imod at se på prototyper

p\_madsen@c.dk says:

Jeg vil gerne være fri for storage delen, men vil godt kigge på pricing

Lars Kringelbach says:

Jeg kan godt se på pricing model.

Morten says:

hehe

Lars Kringelbach says:

Der kom jeg vist for sent

Lars Kringelbach says:

Jeg kan også se på arkitektur delen.

marjus@hotmail.com says:

jeg har ingen præferencer, hvad som helst

Lars Kringelbach says:

Uanset hvad skal vi nok dele en eller flere af opgaverne.

Morten says:

det ser ud til der ikke er nogen der gider kigge på service delen

marjus@hotmail.com says:

jo så kigger jeg på service

p\_madsen@c.dk says:

Er det ikek noget med at der er to på storage prototypen til at starte med? Lyder til den lige skal tales helt på plads?

Morten says:

jo mere jeg tænker over det, tror jeg rest passer ret dårligt fordi det ikke er oplagt OO, men alting kan jo tweakes

Lars Kringelbach says:

Jeg starter med at kigge på arkitektur, og så kan jeg snakke med dig peter om pricing.

p\_madsen@c.dk says:

Eller er der to på arkitekturen?

p\_madsen@c.dk says:

Lars: Lyder fint.

Morten says:

laver vi nye projekter i code mappen, eller hvordan skal vi strukturere kodningen

Lars Kringelbach says:

Ja, det lyder fint.

p\_madsen@c.dk says:

En folder og så projektet der i. Så kan vi gemme andet løsøre der i også?

Morten says:

ok. så må vi håbe jeg kan oprette et projekt . hvordan med byggefiler? antager vi bare at dem der vil lege med det har eclipse

Lars Kringelbach says:

Skal vi overveje at mødes igen inden aflevering (måske kun dig og mig morten)

Morten says:

det kan vi godt

Lars Kringelbach says:

Ja, lad os bare antage det.

Morten says:

men hvis det skal være i den her uge er det lidt presset, for der er 80 års fødselsdag torsdag og lørdag, og fredag har vi en aftale herhjemme

Lars Kringelbach says:

Hvis det er i weekenden skal vi snart aftale det. Jeg .....

p\_madsen@c.dk says:

Forventer vi benytter Eclipse. Der vælger du bare new Java project så spiller det?

marjus@hotmail.com says:

ja alle har eclipse så det er det letteste

p\_madsen@c.dk says:

Jeg kan desværre ikke tilbyde at mødes i denne uge og herefter er der sgu lidt langt at køre

Lars Kringelbach says:

Nå, det passer nok fint med mig. Jeg kan vist heller ikke lørdag, og måske søndag.

Morten says:

vi mangler jo ud over at lave prototyper, også at skrive noget information om vores erfaringer, og den viden vi har fået ud af det

Lars Kringelbach says:

Ellers skal vi bare tage en hverdag igen. Det er lidt nemmere at rykke en aften ud af kalenderen når man har børn.

marjus@hotmail.com says:

jeg kan være i Århus fredag, ellers ikke

Lars Kringelbach says:

Nemlig

p\_madsen@c.dk says:

Morten: Ja, så vi skal vel helst afslutte prototyper i løbet af ugen?

Lars Kringelbach says:

Nå, skal vi ikke til at kalde det en dag.

Lars Kringelbach says:

Puha... det lyder da presset, men det er nok et godt mål at have.

p\_madsen@c.dk says:

LArs: HØRT! Jeg ligger hele denne samtale op på Wikien på en eller anden måde.

Morten says:

je det er presset

Morten says:

vi må gi det et skud i bøssen

Lars Kringelbach says:

Jeg har allerede oprettet en side til det.

Lars Kringelbach says:

http://code.google.com/p/saip2008/wiki/Samtaler?ts=1222632638&updated=Samtaler