SensoByg
Et innovationskonsortium for sensorbaseret overvågning i byggeriet

TEKNOLOGISK INSTITUT

Systemarkitektur og beslutningsstøtte for trådløs monitering

SensoByg-F1-2

Rapport nr. : SensoByg-F1-2
Udarbejdet af : Peter Andersen & Michael Lassen, Alexandra Instituttet;
Morten Tranberg Hansen, Aarhus Universitet;
Flemming Nyboe, Teknologisk Institut;
Jesper Stærke Clausen, Rambøll A/S

Dato : September 2010

Forord

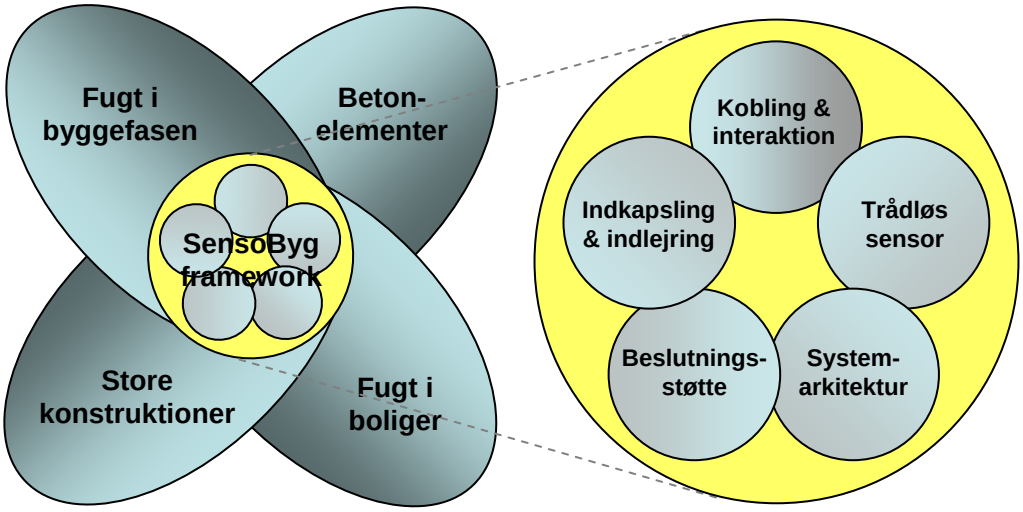
SensoByg innovationskonsortiet blev dannet i 2007. Formålet var at udvikle og demonstrere trådløse overvågningssystemer til brug i byggeriet samt i store konstruktioner så som broer, tunneler mv.

SensoByg blev støttet økonomisk af Forsknings- og Innovationsstyrelsen igennem perioden 2007-2010. Deltagerne i projektet fremgår af rapportens forside. Se også www.SensoByg.dk.

De trådløse overvågningssystemer, som er målet med konsortiets arbejde, er indlejret i konstruktionerne og bygningerne og der er udviklet tilhørende beslutningsstøtteværktøjer. SensoByg har demonstreret muligheder og vurderet teknologier i følgende demonstrationsprojekter:

- D1 – Fugt i boliger og byggeri (byggeriets driftsfase)
- D2 – Store konstruktioner, herunder broer og tunneler samt store stålkonstruktioner
- D3 – Betonelementproduktion
- D4 – Fugt i byggefasen

Foruden disse fire demonstratorer er der en række forskningsemner omkring trådløse systemer og sensorer til indlejring i byggematerialer, som er gennemført. Nedenstående figur illustrerer disse emner i cirklen til højre.



Indholdsfortegnelse

1 Resumé

I herværende rapport gives en oversigt over de resultater der er opnået i de to forskningsaktiviteter vedrørende Systemarkitektur og Beslutningsstøtte omtalt i ovenstående Forord (forskningsaktiviteter F1 og F2).

Vedrørende systemarkitektur gennemgås i denne rapport detaljeret hvorledes de enkelte software komponenter hænger sammen i distributionen af målerdata efter disse er samlet op fra sensorerne, mens de forskellige algoritmer og software arkitekturer der kan anvendes imellem de enkelte sensorer i de såkaldte sensornetværk kun listes kort her idet der i regi af SensoByg er blevet udført et PhD-forløb på Aarhus Universitet med sensornetværk som emne, og resultaterne herfra derfor er beskrevet i den tilhørende PhD afhandling.

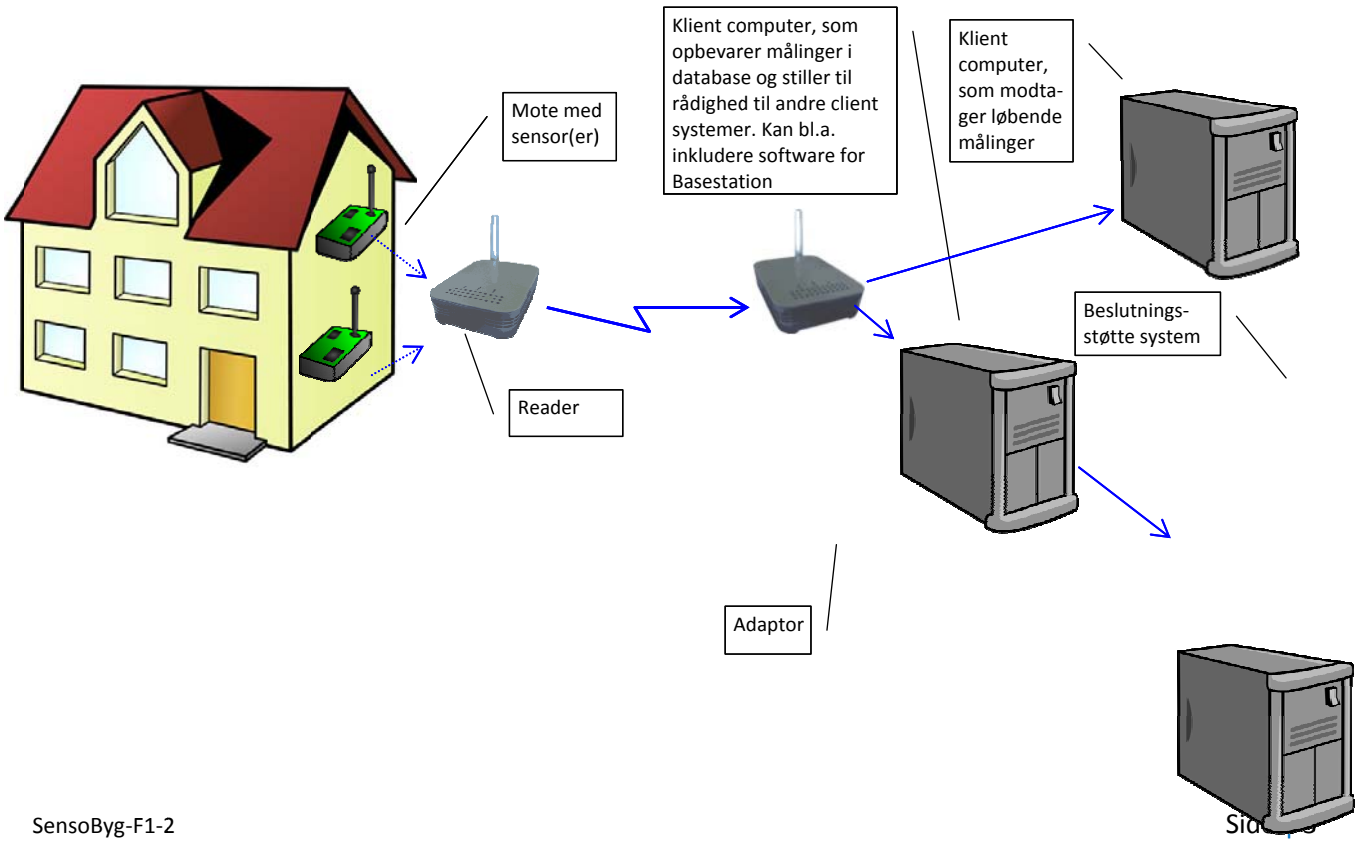
Vedrørende beslutningsstøtte og –systemer skitseres i denne rapport en række eksisterende systemer, der allerede er i anvendelse i bygge-, ingeniør-, og energibranchen, ligesom der skitseres mindre specialiserede systemer udarbejdet i forbindelse med D1-D4 demonstrationsaktiviteterne i SensoByg.
Hovedvægten er imidlertid lagt på en foreslået anvendelse af digitale 3D-bygningsmodeller som et supplement til de ofte tabel- og graf-orienterede eksisterende systemer. I den forbindelse præsenteres en række prototype anvendelser baseret på open-source 3D modellen og værktøjet b-processor.

Denne rapport er tiltænkt et alment teknisk funderet publikum med interesse for opsamling af sensordata og anvendelse af disse i beslutningsstøttesystemer. Afsnittene om den specifikke software arkitektur forudsætter et vist kendskab til software arkitektur diagrammer i UML notationen. Herudover burde rapporten være forståelig uden særlige forudsætninger.

2 Introduktion

I dette dokument beskrives forskellige anvendelser af software i relation til trådløse sensorer. Især fokuseres på *opsamling og distribution af målinger* fra de trådløse sensorer, samt *anvendelse af målingerne i beslutningsstøttesystemer*.
I forhold til figuren i Forordet vedrører de her beskrevne aktiviteter og resultater i de to arbejdspakker F1 Systemarkitektur og F2 Beslutningsstøttesystemer.
I tilknytning til F1 er der på Aarhus Universitet arbejdet med en PhD indenfor trådløse sensorer. Der findes særskilte dokumenter, først og fremmest [1], der beskriver resultaterne af dette arbejde, herunder de relevante dele af den software der er indlejret i de enkelte sensor Motes (se definition nedenfor).

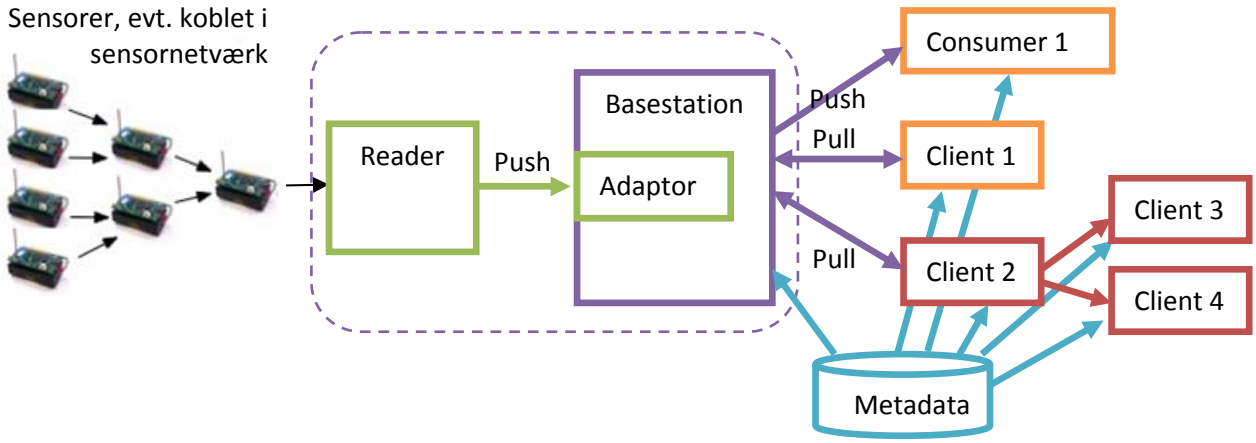
Den typiske anvendelse af sensorer i SensoByg (se demonstratorerne D1-D4) er trådløse sensorer indlejret i beton (eller andre byggematerialer) i bygningsværker, enten etablerede eller under opførsel.
For at illustrere de forskellige software elementer der er i sving i denne sammenhæng tages her udgangspunkt i et etagebyggeri hvor hver lejlighed er instrumenteret med et antal sensorer i forskellige rum, fx badeværelser og køkken.
Relevante målinger disse sensorer kunne udføre er fx *temperatur* og *(relativ) luftfugtighed*. Ofte er der samlet flere sensorer i samme elektronik "dims", og for at kunne skelne mellem den enkelte sensor (som måler fx temperatur) og det elektriske kredsløb indeholdende en eller flere sådanne sensorer kalder vi selve "dimsen" for en *mote* (engelsk betegnelse, der kommer af Sensor Node, se [2]).
Sædvanligvis vil man i et instrumenteret byggeri lave en *opsamling af rå sensormålinger* i nærheden af de enkelte motes – vi kalder denne lokale enhed for en *reader*. Disse målinger vil derefter typisk transmitteres – sædvanligvis som fuldstændig rå data (for ikke at bruge processor- og batterikraft på databehandling ude på opsamlingsstedet) – til det sted hvor sådanne målinger *opbevares* (fx hos boligselskabet). Her vil der sædvanligvis være brug for en dedikeret modtagerenhed, som kender til readeren og de formater denne transmitterer i, vi kalder denne for en *adaptor*. Fra den opsamlende computer vil målingerne derefter *stilles til rådighed* til videre *bearbejdning/analyse* – typisk som led i *forbrugsregistrering, overvågning* eller mere generelle *beslutningsstøttesystemer*. Dette scenarium kan illustreres således:



Figur 1: Typisk sensor scenarium

I sidste led – anvendelsen af sensormålinger – skelner vi mellem to basale anvendelser: En anvendelse, hvor man abonnerer på aktuelle øjebliksdata, og en anvendelse hvor man får adgang til opbevarede historiske data og kan forespørge vilkårligt på disse.
Overvågningssituationen benytter ofte abonnent tilgangen mens mere avancerede analyse- og beslutnings-støttesystemer ofte bruger forespørgsels tilgangen.
Der findes naturligvis mange variationer af dette scenarium. Forbindelsen mellem reader og adaptor kan være ganske tæt (de kan sidde i samme "kasse") men der kan også være tale om at målingerne fx sendes med GPRS over telefonnettet fra reader til adaptor. I visse tilfælde er der begrænsninger i adgangen til det netværk som readeren sidder i (fx i en boligblok) – dette kunne være fx en firewall. Derfor vælges oftest at *push'e* data fra readeren til adaptoren i modsætning til at lade adaptoren *pull'e* (forespørge) efter data fra readeren. Dette fordi firewalls sædvanligvis begrænser indgående trafik, men sjældent udgående.
Som nævnt foregår den initielle bearbejdning af målingerne ofte først i adaptoren, således at readeren blot sender de helt rå sensordata. Disse vil typisk bestå af en sensor-id og et enkelt stort tal, som er et udtryk for værdien af det sensoren måler. Det er så typisk adaptorens opgave – med sit nære kendskab til readeren – at "oversætte" disse rå sensordata til målinger med semantisk betydning, fx at det er en temperaturmåling i enheden Celcius med 2 decimaler.
Kombinationen af en reader, en adaptor, og den software der "har fat i" adaptoren, og derigennem får sensordata til videre distribution kalder vi under et for en *basestation*. Som det fremgår ovenfor kan base-stationen være fysisk distribueret men kan også være samlet ét sted.

Mere skematisk kan det typiske scenarium skitseres således.



Figur 2: Skematisk scenarium

Som det fremgår opererer vi med et yderligere begreb: *Metadata*. Dette er "data om data" som giver den semantiske sammenhæng mellem rå sensor data og målinger, fx at målingerne er temperaturmålinger i Celsius med 2 decimaler. Disse begreber uddybes yderligere i de følgende afsnit.

3 Systemarkitektur

I dette afsnit vil vi beskrive softwarearkitekturen af sensorframeworket. Systemarkitekturen er objektorienteret, og beskrivelsen indeholder derfor klasse-diagrammer i *UML*-syntaks. Se fx [3] for en introduktion til klasse-diagrammer.

3.1 Basale softwarearkitektur

De grundlæggende begreber i softwarearkitekturen er Mote, Sensor og Measurement.

Figur 3: Mote, Sensor, Measurement

Mote

En mote er en fysisk enhed typisk med strømkilde (eventuelt i form af et batteri), kommunikations chip og en eller flere sensorer. En mote har følgende egenskaber:

- *Id*: Identificerer enheden unikt i systemet.
- *Position*: Placerer enheden i 3D i et velkendt koordinat-system tilhørende en geografisk model eller digital bygningsmodel.

Sensor

En sensor er indlejret i en mote og måler temperatur, fugt eller noget tredje. En sensor har følgende egenskaber:

- *Id*: Identificerer enheden unikt i systemet.
- *Type*: Beskriver typen af sensorens målinger

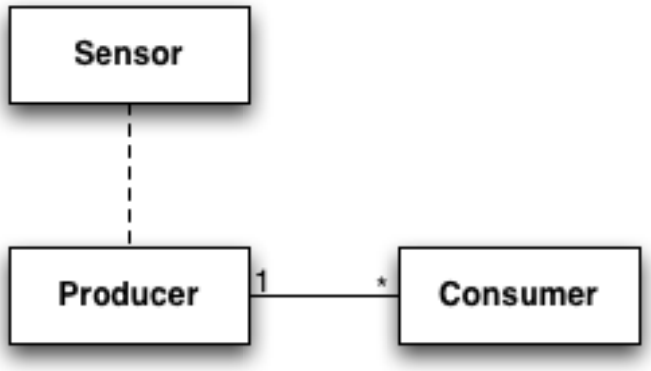
Measurement

En measurement repræsenterer en måling fra en sensor og har følgende egenskaber:

- *Timestamp*: Fortæller hvornår målingen er foretaget.
- *Value*: Den målte værdi

En mote indeholder en eller flere sensorer og en sensor har produceret et antal målinger. Informationen om motes og sensorer kaldes for metadata mens målingerne kaldes måledata.

En sensors evne til at producere målinger er repræsenteret med de særskilte begreber Producer og Consumer.



Figur 4: Producer, Consumer

Producer
En Producer er tilknyttet en sensor og holder styr på et antal consumers, som abonnerer på målinger fra den tilknyttede sensor. Produceren holder øje med sensoren og sørger for at distribuere målinger til de abonnerende consumers, når der er en ny måling fra sensoren. En Producer har følgende egenskaber:

- *addConsumer(consumer)*: Tilføjer en consumer som abonnerer på målinger.
- *removeConsumer(consumer)*: Fjerner en consumer igen.

Consumer
En Consumer kan registreres som abonnerende på målinger fra en sensors Producer. Consumeren vil modtager målinger fra sensoren, når Produceren ser noget nyt. Egenskaber:

- *consume(measurement)*: Denne funktion kaldes, når der er en ny måling fra sensoren.

Den her beskrevne producer/consumer arkitektur bruges til live overvågning af sensorerne i et sensor netværk. Når data er opsamlet i en database eller en logfil eller lignende, kan man have adgang til historiske måledata fra en sensor. En service som giver en sådan adgang kaldes en HistoryService.

HistoryService
En HistoryService giver adgang til historiske målinger fra en sensor. Egenskaber:

- *fetch(sensor, from, to) -> measurements*: Sender measurements fra sensoren med tidsstempler mellem de to angivne datoer.

Efter denne gennemgang af de overordnede begreber i softwarearkitekturen, vil vi i de følgende afsnit beskrive server-klient arkitekturen.

3.2 Serversiden

Dette afsnit omhandler software komponenter på serversiden.
Her har vi følgende begreber til at beskrive dataopsamling og distribution:

- **Reader**: En reader opsamler data trådløst fra sensorer i dens umiddelbare nærhed og sender data videre til en BaseStation.
- **Adaptor**: Kender til kommunikations protokollen i et konkret sensornetværk og kan derfor modtage data fra Readers og omsætte til et generelt format.

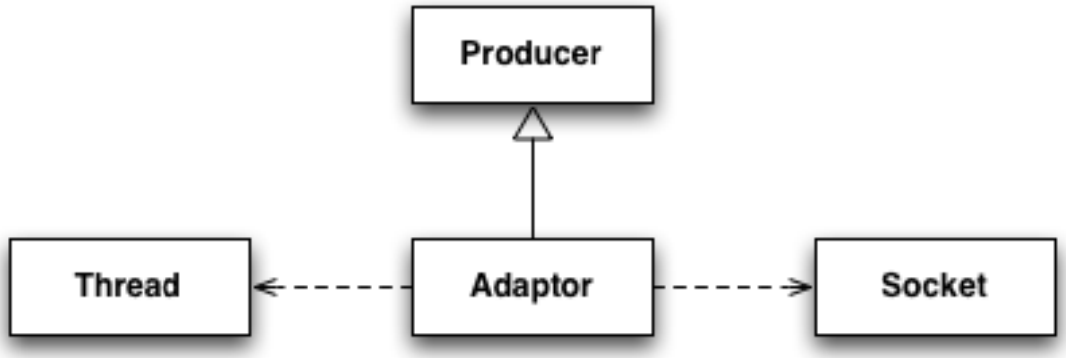
- **Storage**: Har mekanismer til at gemme sensor-målinger, fx i en database**.**
- **BaseStation**: En server applikation, som sørger for at opsamle data fra et eller flere sensornetværk og distribuerer data til klienter og consumers på nettet fx via et http interface.

Dataopsamling og distribution beskrives mere detaljeret i de følgende afsnit.

3.2.1 Dataopsamling

I dette afsnit vil vi beskrive hvilke software komponenter, der er implementeret til opsamling af data fra et sensornetværk. Når data skal opsamles fra et konkret sensornetværk, er der to sider af sagen: 1) adgang til fortløbende sensormålinger fra sensornetværket (via en Adaptor), og 2) gemning af data i et datalager (Storage).

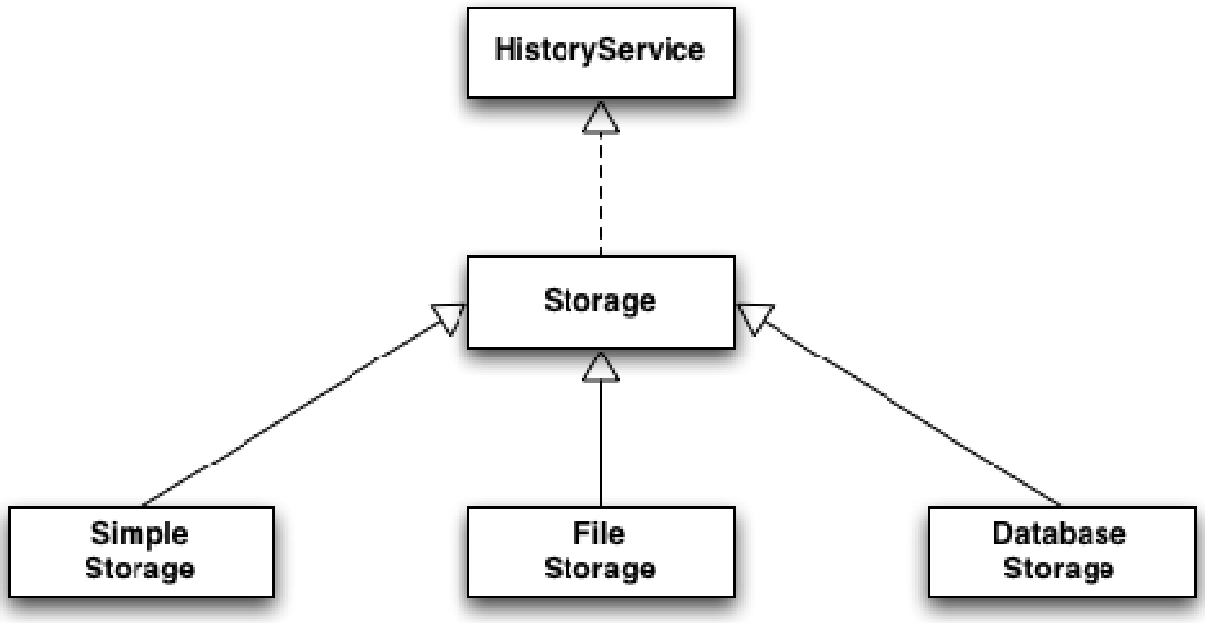
Adgang til data fra et konkret sensornetværk, sker via en Adaptor. En Adaptor er en særlig form for Producer, der oversætter fra det sensornetværk specifikke data format til et generisk data format, som derefter afleveres til de registrerede Consumers. En Adaptor vil typisk have en tråd, som via tcp/ip kommunikation eller lignende modtager data fra Readeren, der eventuelt bruger GSM til afsendelse af data.



Figur 5: Adaptor, Thread, Socket

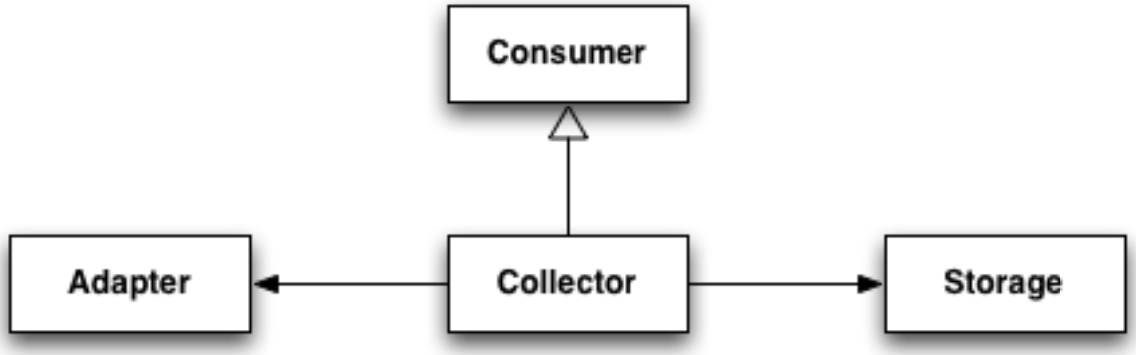
Opsamlede målinger gemmes i et datalager, som er repræsenteret med den abstrakte klasse Storage. Et Storage har metoder til tilføjelse af målinger. Storage er subinterface til HistoryService, som giver adgang til opslag i de lagrede data. Det vil sige at implementationer af Storage også implementerer HistoryService. Der er implementeret tre forskellige storage mekanismer:

- **SimpleStorage**: Data gemmes i memory og forsvinder derfor igen, hvis applikationen genstartes. Bruges til aftestningsformål.
- **FileStorage**: Data gemmes i en fil i form af et Excel regneark i csv format. Bruges til udveksling af data med andre applikationer.
- **DatabaseStorage**: Data gemmes i en SQL database.



Figur 6: Storage implementationer

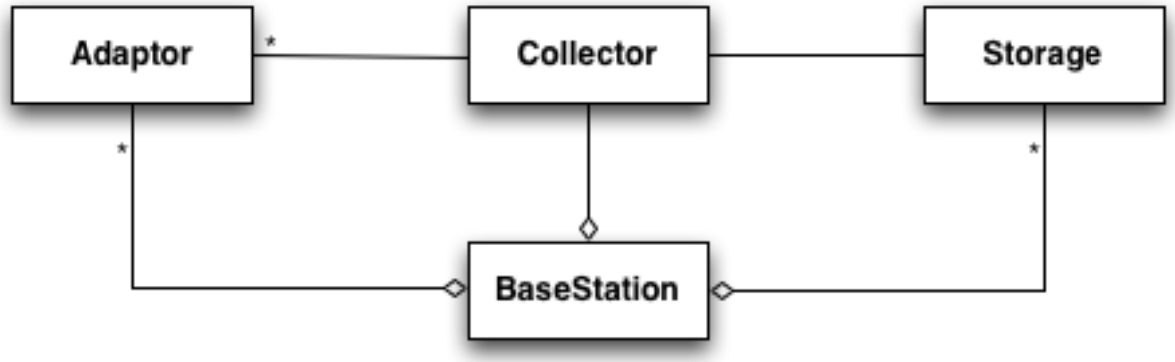
En Collector er en særlig Consumer, som registrer sig hos Adaptoren og gemmer de data, der modtages i et tilknyttet Storage.



Figur 7: Collector

3.2.2 Distribution

En BaseStation tager sig af distribution af data til klienter, som illustreret i Figur 8 nedenfor.



Figur 8: BaseStation

En BaseStation udgør en server, som via fx http protokollen eksponerer måledata til klienter. En BaseStation kommunikerer via en Adaptor med Readeren, hvorfra sensordata modtages. Disse data opsamles i en Storage (fx i form af en database), og distribueres til klienter.

Der er to forskellige mekanismer for distribution til klienter:

- **Push**: BaseStation sender data til klienten, når der sker noget nyt. En *push-klient* abonnerer således på fortløbende opdateringer fra sensornetværket.
- **Pull**: BaseStation sender data til klienten, som følge af en forespørgsel. En *pull-klient* laver således opslag i historiske data fra sensornetværket.

Begge disse distributions mekanismer er implementeret vha. http-protokollen:

En klient kan lave en forespørgsel på historiske data ved at udføre en GET request på en server-url, som indkoder identifikation af motes og sensorer samt det tidsrum hvorfra sensormålinger ønskes.

En klient kan abonnere på fortløbende opdateringer hos en server ved at registrere en url hos serveren hvortil PUT requests skal sendes, når der er en ny måling på en sensor.

I begge typer af klient relationer serialiseres sensordata i form af en protokol i et særligt XML format.

Protokollen kan bruges uafhængig af Java, men for nemheds indeholder frameworket java-klasser, som giver nem adgang til distributions-arkitekturen.

3.3 Klientsiden

På klientsiden kan klienterne spille forskellige roller:

Overvågning: En overvågnings applikation er en push-klient som har adgang til fortløbende målinger fra netværket. Set fra klientens side foregår dette ved at klienten registrerer en consumer hos en producer for de sensorer, som overvåges. Klient-side produceren sørger for at kommunikere med serveren via push-protokollen.

Dataanalyse: En klient, som har adgang til historiske data fra et sensornetværk via et regneark eller direkte adgang til opslag i en database, kan udføre beregninger på de historiske data, men overvåger ikke sensornetværket via fortløbende opdateringer. Dette foregår ved at klienten laver forespørgsler til en HistoryService på klient-siden, som sørger for at kommunikere med serveren via pull-protokollen.

En klient kan naturligvis spille begge roller på samme tid.

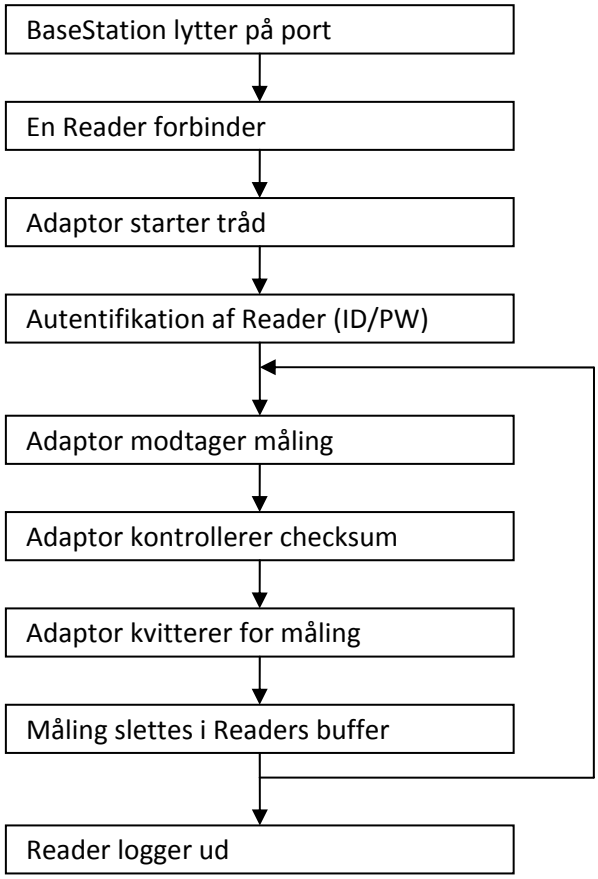
3.4 Et tidligere design og prototype

Som et led i PhD projektet på Aarhus Universitet under SensoByg [1] er der blevet lavet en tidlig arkitekturspecifikation for SensoByg – en arkitektur der i store træk er identisk til den ovenfor beskrevne, se [4].

Der var ingen SensoByg-sensorer eller Brunata-sensorer til rådighed på dette tidlige stadium af projektet, så en reference implementation af den oprindelige Mote side blev implementeret ovenpå det gængse TinyOS operativsystem. Hele denne kodebase er tilgængelig som open-source, se [5].

3.5 BaseStation for SensoByg-sensorer

Som et konkret eksempel på hvad en BaseStation udfører illustreres nedenfor mekanismerne i en BaseStation, implementeret af Teknologisk Institut (TI), som benytter de fysiske SensoByg sensorer og tilhørende Reader + Adaptor:



Figur 9: Flow i TI BaseStation

BaseStationen lytter på en valgfri port. I Readeren indstilles følgende:

1. Adaptorens IP adresse og det valgte port nummer
2. Max. antal målinger i Readerens buffer, før forbindelse
3. Max. tid før forbindelse
4. Password

Indstillingerne 2 og 3 angiver hvor ofte Readeren skal oprette forbindelse til Adaptoren, for at aflevere data. Readeren forbinder når den første grænse nås. Forbindelsen sker via GPRS, dvs. over GSM nettet fra Readeren til teleudbyderen, og derfra over internettet til Adaptoren.
Ved hver indkommende forbindelse til porten, opretter Adaptoren en tråd til håndtering af forbindelsen. Adaptoren kan således håndtere at flere Readere er forbundet samtidigt. Når tråden er startet, er forløbet som følger

1. Readeren autentificerer sig overfor dataserveren med sit ID nummer og password. Adaptoren kontrollerer disse imod en liste, som angives på programmets brugerflade. Hvis readeren ikke er på positivlisten, eller angiver forkert password, terminerer tråden.
2. Readeren afsender nu en måling ad gangen. Hver måling indeholder en 16-bit checksum, som Adaptoren kontrollerer. Hvis checksummen passer, afsendes en kvittering for modtagelsen. Adaptoren omformer den rå måling til en læsbar tekst, skriver den modtagne måling i en fil, med et filnavn som svarer til Readerens ID nummer.
3. Hvis Readeren modtager en kvittering, slettes den pågældende måling fra dens interne data buffer. Hvis ikke, forsøger den at afsende samme måling igen.
4. Når alle samples er kommet igennem, logger readeren af, og dataserveren terminerer tråden.

I punkt 2 åbnes og lukkes output filen i forbindelse med skrivning af hver enkelt sample, for at minimere risikoen for data korruption i tilfælde af hardware fejl. Det overlades således til operativsystemet at håndtere konflikter i forbindelse med anden adgang til filen. Når måledata læses (typisk af anden software), anbefales det at kopiere data filen og åbne kopien, så man ikke blokerer for videre skrivning imens.

Som det fremgår er TI datalagrings løsningen et FileStorage.

3.6 Sensornetværk PhD oversigt

Som nævnt ovenfor vil vi ikke her gå i detaljer med arbejdet udført vedrørende sensornetværk i PhD projektet på Aarhus Universitet – der henvises til [1]. Nedenfor gives blot en oversigt over emner der behandles i PhD'en.

3.6.1 Sikker gruppe kommunikation

En ofte brugt sensornetværk radios hardware sikkerheds mekanisme er blevet testet baseret på tid og energi og fundet favorable i forhold til tilsvarende software implementationer. Derudover er der blevet forslået en gruppe nøgle fornyelses protokol der sammen med de før nævnte hardware mekanismer kan bruges som en komplet løsning til at sikre gruppe kommunikation i et sensor netværk på en energi effektiv måde.

3.6.2 Transmission af flere pakker

Effektiv overførelse protokol af mere end én pakke i et sensor netværk er blevet foreslået. Ideen er simpel: i stedet for at sende én pakke af gangen med en tilsvarende svar på modtagelse fra modtageren sender vi

en blok af pakker og så kun ét svar til sidst, der fortælle hvilke pakker er gået tabt. Bidraget er en implementation af dette i et sensor netværk og en analyse af hvordan dette forbedre nuværende metoder mht. både effektivitet og pålidelighed.

3.6.3 Selektiv Transmission

Selektiv transmission af pakker er blevet implementeret og testet i et sensor netværk. Der antages at alle pakker har en vigtighed (evt. baseret på hvor kritisk en temperatur måling er) og der er så implementeret en dynamisk algoritme for om en pakke skal sendes eller smides væk. Denne algoritme er baseret på online energi estimater af hvad det koster at sende og modtage pakker, og vigtighederne af tidligere pakker beslutter om en pakke skal sendes eller ej. Målet med dette er at maksimere summen af vigtighed en sensor sender gennem dens levetid.

3.6.4 Forening af broadcast

Der er blevet arbejdet med at forene broadcast i et sensor netværk. Målet er at spare på antallet af broadcasts fra forskellige protokoller ved at samle dem i én broadcast. En protokol til at gøre dette er blevet implementeret og testet sammen med to forskellige routing protokoller, en tids synkroniserings protokol, en kontrol udbredelses protokol, og en sensor fejl detekterings protokol. Forening opnås ved at tilbageholde nogle protokollers broadcast, så en vigtig del af dette er at teste hvordan sådanne tilbageholdelser påvirker de overliggende protokoller.

4 Beslutningsstøttesystemer

Begrebet ”Beslutningsstøtte systemer” dækker over en række forskellige software systemer, se fx [6] for en oversigt over begrebet. I SensoByg konsortiet har vi beskæftiget os med forskellige typer beslutningsstøtte systemer til brug indenfor byggeriet i de forskellige demonstrations scenarier, som er defineret i aktiviteterne D1-D4 omtalt i Forordet. Nedenfor følger en kort oversigt over nogle af aspekterne ved udvalgte systemer. Der må generelt henvises til de respektive virksomheder for en mere detaljeret indsigt i disse systemer.

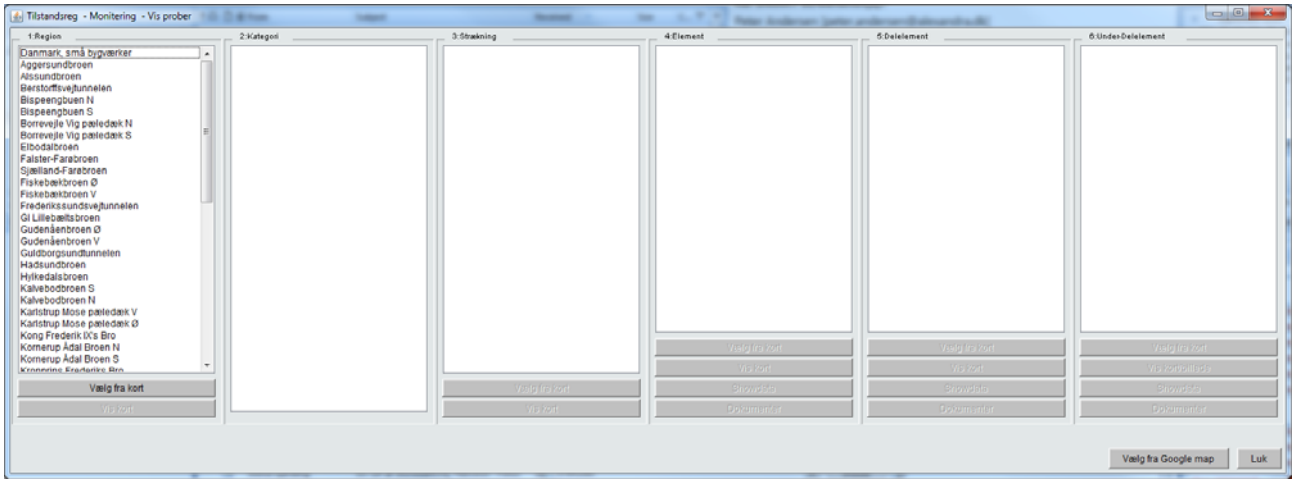
4.1 Traditionelle beslutningsstøttesystemer i byggeriet

Indenfor byggeriet har vi i SensoByg primært beskæftiget os med beslutningsstøttesystemer til overvågning og beregninger vedrørende varmeforbrug og bygningssundhed, prognoseberegning for modning at nystøbt beton, og endelig forudsigelse af potentielle skimmelsvamp angreb i eksisterende byggeri.

4.1.1 Rambøll ”SMART-Monitoring”

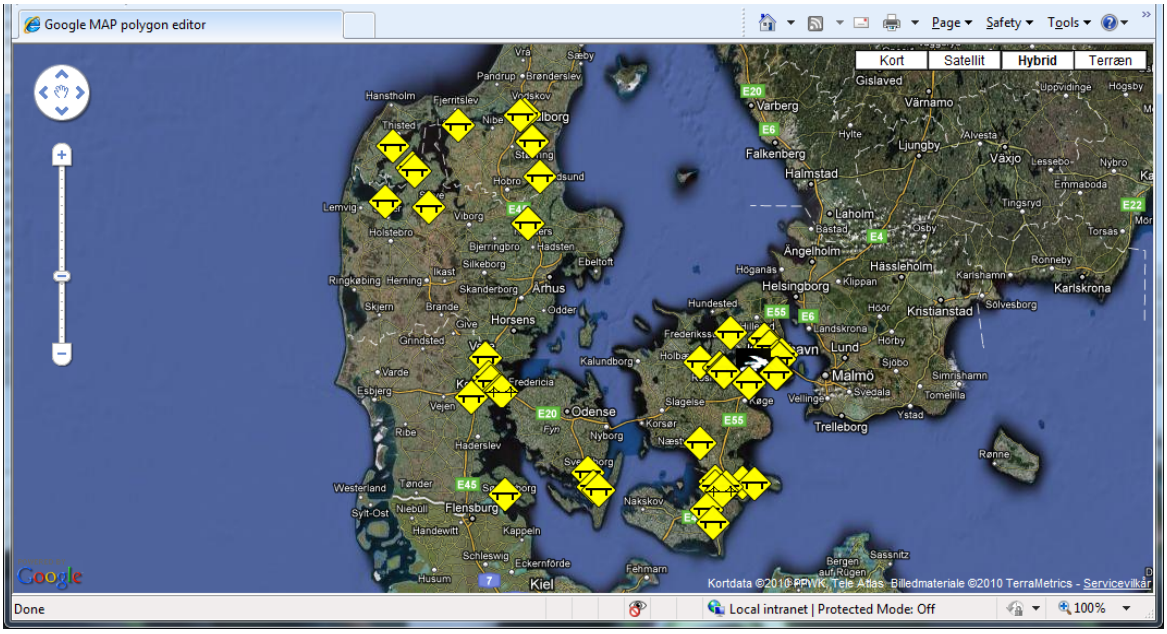
Rambøll SMART-Monitoring systemet [7] (herefter blot betegnet SMART) er et generelt overbliks- og styringssystem[1] til større infrastrukturanlæg såsom veje, broer, tunneler, havne, lufthavne, og spildevandssystemer. Vi vil her kort illustrere anvendelsen af systemet til monitorering af broer i Danmark. Når man går ind i SMART og vælger at få vist alle ”Prober” (synonym for sensorer) fremkommer tabellen vist i Figur 10 nedenfor:

[1] I Rambølls interne terminologi kaldes det et forvaltningssystem



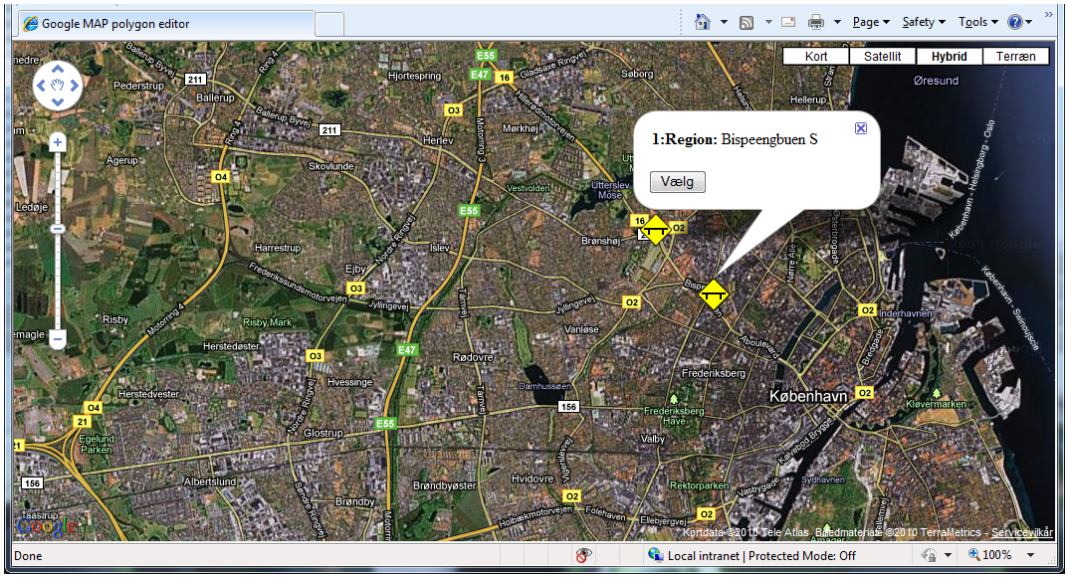
Figur 10: SMART tabel over monitorerede broer

Alle bygværker kan ses i Google Maps ved at trykke på knappen "Vælg Google map" nederst til højre i vinduet. I Google map vises en oversigt over alle registrerede broer i databasen, se Figur 11 nedenfor:



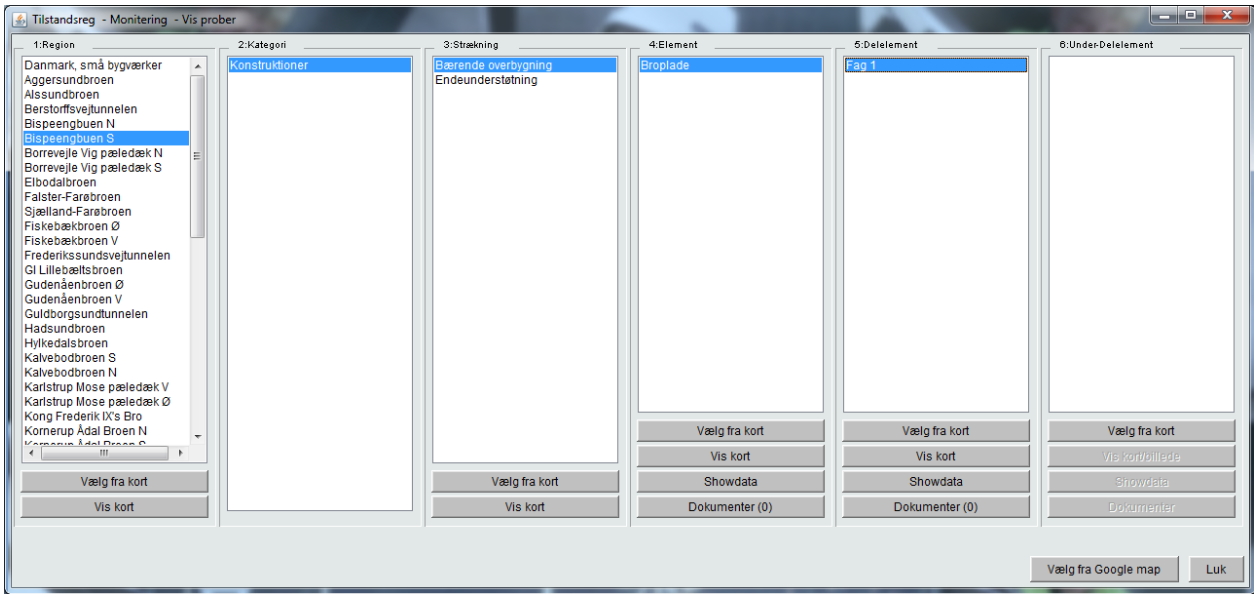
Figur 11: SMART monitorerede bygværker i Danmark

Der kan zoomes ind på de enkelte broer og ved at pege på et ikon for et bygværk, kommer der et kort resumé med info om bygværket på skærmen, som illustreret i Figur 12 nedenfor:



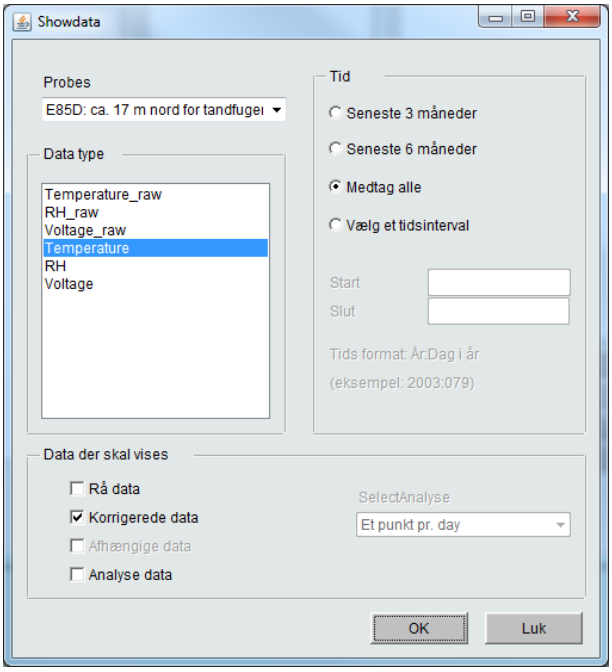
Figur 12: SMART oversigts info for udvalgt bygværk

Tryk på "vælg" knappen og SMART åbnes med info om den pågældende bro, som illustreret i Figur 13 nedenfor. For den pågældende bro, er der f.eks. installeret en sensor i bropladen i fag 1.



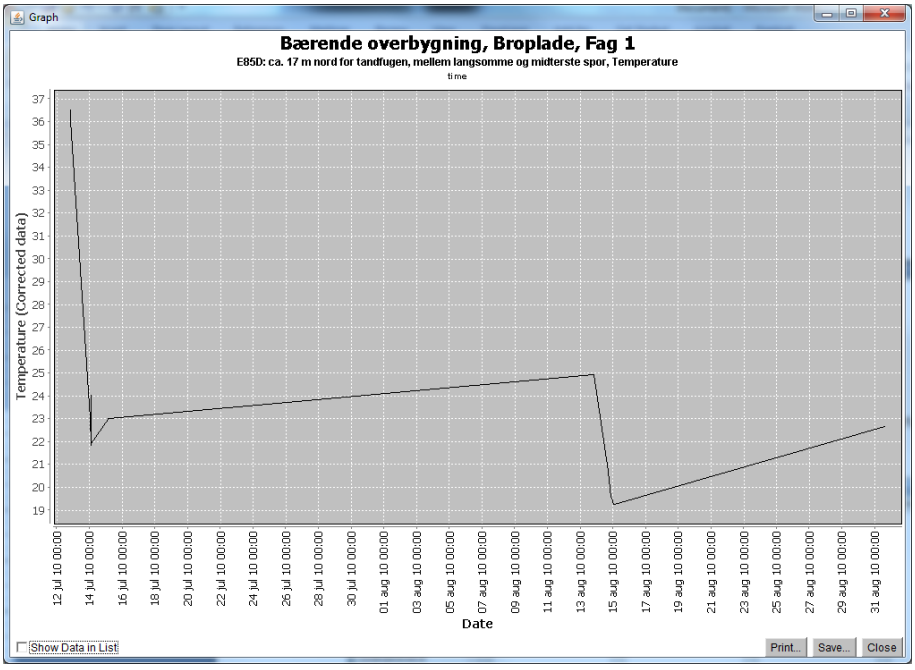
Figur 13: SMART tabel over udvalgt bygværk

Ved et tryk på knappen ”Showdata” vises info om den pågældende sensor, som illustreret i Figur 14 nedenfor:



Figur 14: SMART Sensor Data Valg

I Figur 15 herunder er temperaturen illustreret grafisk på baggrund af målinger ved installation d. 13. juli og i perioden frem til 31. august. Døgnvariationen kan ses ved målinger i en weekend d. 13-15. august.



Figur 15: SMART graf over temperaturmålinger

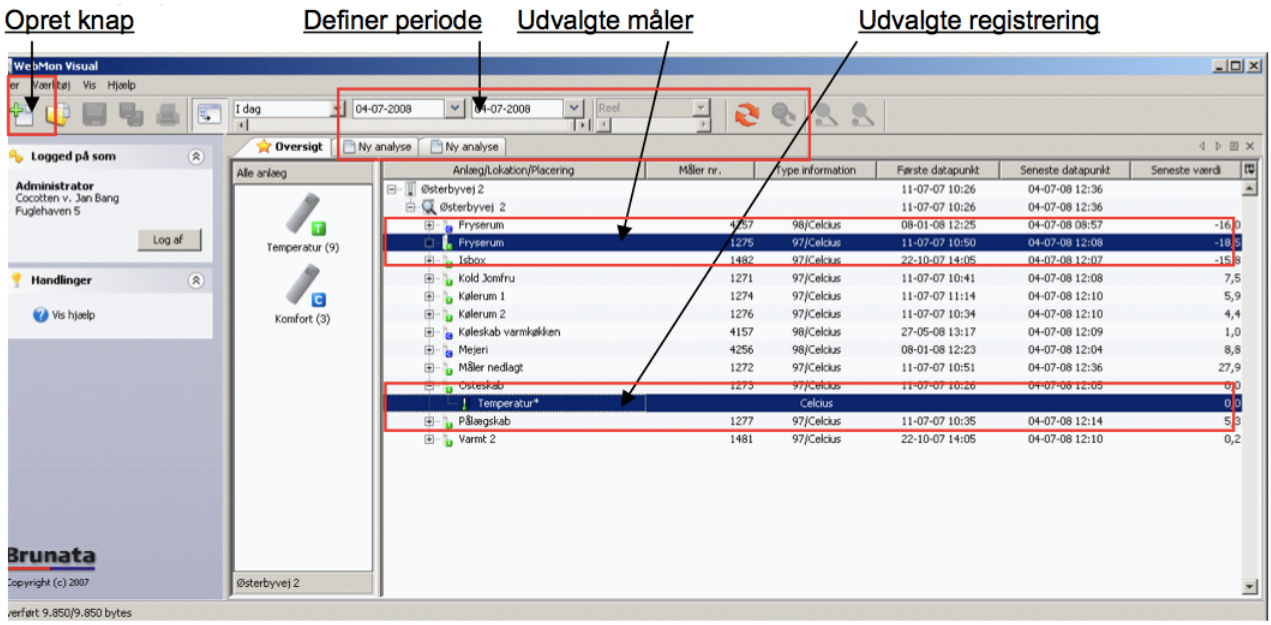
Overordnet kan det for SMART bemærkes, at der benyttes en visuel grafisk model (Google Maps) i den overordnede navigation, men at når man zoomer ind til det enkelte bygningsværk skiftes der til en tabel- og

graf domineret brugergrænseflade, hvor indgående kendskab til bygværkets strukturelle topologi er en forudsætning for at kunne navigere korrekt.

4.1.2 Brunata WebMon Visual

Brunata har forbrugsmålere opsat i en stort antal private hjem og virksomheder i Danmark og resten af verden. Til simpel monitorering, analyse og dataudtrækning tilbydes det webbrowser baserede program WebMon [8]. Til mere avanceret og visuel brug af målerdata tilbyder Brunata det Java baserede egentlige PC program WebMon Visual [9]. Nedenfor vises et par glimt af sidstnævnte.

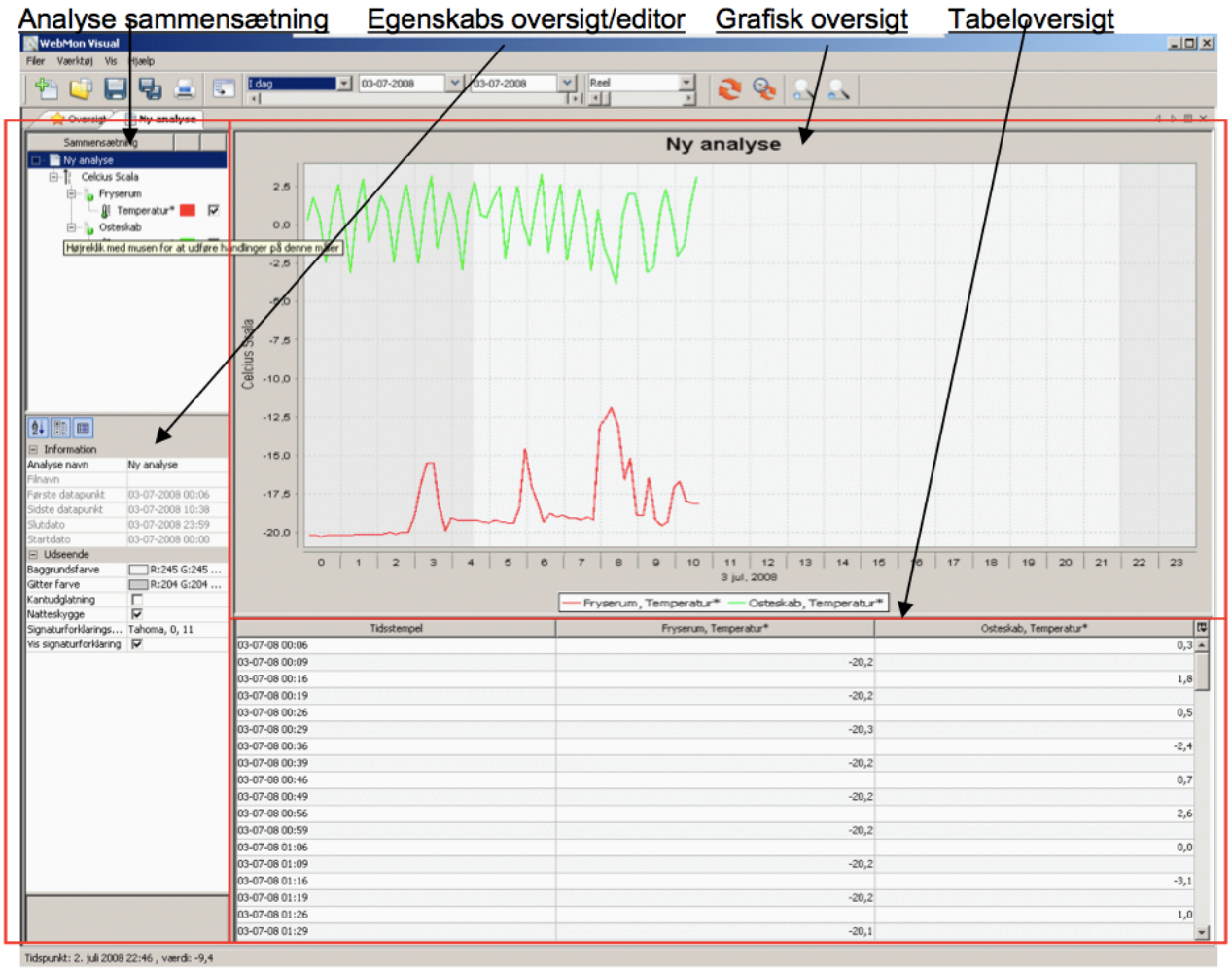
Ligesom ved Rambølls SMART system (se 4.1.1 ovenfor) er indgangs viewet i programmet en tabel over installationer[2], som gengivet i Figur 16 nedenfor:



Figur 16: WebMon Visual lokationsoversigt

Ligesom i Rambøll SMART kan man i WebMon Visual navigere ind til de enkelte målere ved at klikke rundt i disse tabeller og man kan for de enkelte målere få vist målerdata både i tabelform og ved hjælp af grafer. I Figur 17 nedenfor ses både tabeller og grafer for historiske målerdata over temperatur i et gartneri, i det såkaldte kompositoriske view, hvor disse visninger er kombineret i samme skærmbillede.

[2] WebMon Visual synes ikke at have den geografiske navigation såsom Google Maps, som SMART har



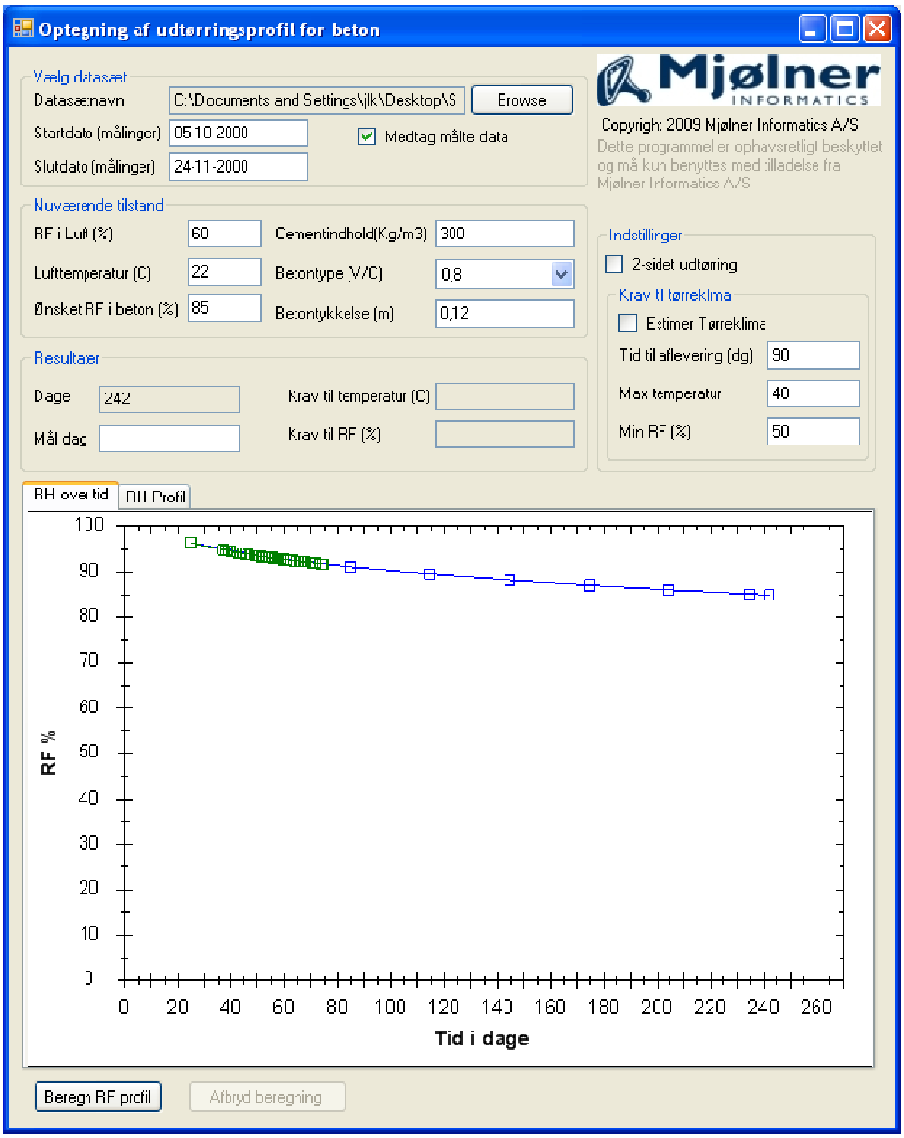
Figur 17: Brunata WebMon Visual Kompositions overblik

Selvom pladsen her ikke tillader en detaljeret gennemgang af Brunata WebMon Visual, illustrerer ovenstående at opbygningen af brugergrænsefladen på mange måder er sammenlignelig med opbygningen af brugergrænsefladen i Rambøll SMART: Det er tabeller og matematiske grafer der dominerer de visuelle udtryksformer.

4.1.3 Prognoser for betonmodning og skimmelsvamp angreb

Udover de ovennævnte eksisterende, allerede i stor stil industrielt anvendte beslutningsstøttesystemer har der været arbejdet med nogle andre systemer indenfor rammerne af SensoByg.

I forbindelse med demonstrator projektet Fugt i Byggefasen er i et samarbejde mellem Lunds Universitet og Teknologisk Institut blevet udviklet nogle avancerede algoritmer til forudsigelse af modningsforløbet i nystøbt beton. I et samarbejde med Teknologisk Institut har Mjølner Informatics implementeret beregningsalgoritmerne, og har i den forbindelse endvidere implementeret en prototype på et beslutningsstøttesystem med disse algoritmer indbygget. Systemet kombinerer målinger fra SensoByg sensorer i det nystøbte beton med prognosealgoritmerne, og bruger således disse aktuelle målinger som seeds til dynamisk kalibrering af prognosen, som algoritmerne beregner.
I Figur 18 nedenfor ses en prognose for den relative fugtighed i betonen for en periode på godt 240 dage, hvor knap 80 dages målinger indgår som kalibrering.



Figur 18: Prototype på beslutningsstøttesystem vedrørende modenhed af nystøbt beton

Selv om formålet med denne prototype og de bagvedliggende algoritmer ikke så meget er overvågning som SMART og WebMon Visual, men i stedet har til formål at forudsige hvornår et byggeri kan gå videre (ved tilstrækkelig opnået modenhed af det allerede udstøbte), så er der mange lighedspunkter til SMART og WebMon Visual i det valgte design af beslutningsstøttesystemet: Rå måleværdier og matematiske grafer udgør de formidlede oplysninger. Hvis man forestiller sig dette benyttet i et større byggeri, hvor der er flere samtidige nystøbte betonsegmenter, vil man formodentlig mangle en form for overblik over hvilke bygningsdele det er man overvåger, ligesom man måske kunne have brug for en simplere grafisk indikation af modenheden af de enkelte elementer (fx bare en farve, og måske en popup med dage til fuld modning).

I forbindelse med demonstratoren Fugt i Boliger har SBI udviklet et antal formler til, på baggrund af historiske fugt- og temperaturmålinger, at udregne prognoser for risikoen for skimmelsvampangreb i det monitorerede byggeri. Der er ikke blevet udformet et egentligt beslutningsstøttesystem baseret på disse formler,

men der er planer om at indføre denne funktionalitet i den B-processor baserede brugergrænseflade, som beskrevet i afsnit 4.2 nedenfor[3].

4.2 Ny ide: 3D bygningsmodel som basis for beslutningsstøttesystem

Som illustreret i de foregående afsnit, benyttes der oftest tabel- og graf-orienterede brugergrænseflader i beslutningsstøttesystemerne. Dette afspejler givetvis at brugerne af disse systemer primært har været eksperter indenfor de enkelte områder, fx ingeniører. Vores hypotese er at både disse eksperter og evt. mindre specialiserede brugere, såsom viceværter i boligkomplekser, vil have glæde af at tilføje en rumlig navigation til disse beslutningsstøttesystemer. En tendens der kendes fra flere andre brugergrænseflader, som fx Google Maps, hvor både Street View og forskellige 3D simuleringer er tilføjet og nu benyttes af "almindelige" brugere. 3D er med andre ord ved at vinde indpas i brugergrænseflader.

Nedenfor beskrives arbejdet med at bruge en 3D bygningsmodel og tilhørende 3D software som en udbygning af byggeriets beslutningsstøttesystemer.

Et beslutningsstøtte system til sensornetværk i en bygning bør generelt indeholde i hvert fald følgende dele:

- System til konfiguration af sensornetværk.
- System til overvågning af sensornetværk
- System til at foretage beregninger på data fra sensornetværk

Vi har i SensoByg konsortiet arbejdet med den ide at bruge en 3D bygningsmodel som basis for disse systemer. Vi har brugt B-processor systemet [10], som er et open-source projekt, under udvikling på Alexandra Instituttet og Århus Arkitektskole. B-processor systemet er et generelt værktøj til håndtering af 3D bygningsmodeller.

Konfigurationen af et sensornetværk indeholder information om antal af motes med tilhørende sensorer, den rumlige placering af den enkelte mote, identifikations-koder på motes og sensorer, samt type information om sensorerne. Disse informationer kaldes under et metainformation om sensornetværket. Metainformationen kan være repræsenteret på forskellig vis: en tabel i et Excel regneark, en xml konfigurations fil, eller en tabel i en database. Denne sensorkonfiguration kan naturligt knyttes til en 3D bygningsmodel, således at konfigureringen foregår i B-processor applikationen ved at motes og sensorer placeres og konfigureres direkte i en 3D model af bygningen. Metadata kan knyttes til bygningsmodellen på mindst to måder:

1) Metainformationen om sensornetværket er indlejret i selve bygningsmodellen og kan eksporteres til en Excel-tabel eller en xml konfigurationsfil, og tilsvarende importeres fra sådanne eksterne filer.
2) Metainformation placeres i en (meta)database, separat fra byningsmodellen.

Overvågning af et sensornetværk foregår, som beskrevet i afsnit 3, ved at overvågningsapplikationen abonnerer som Consumer på målinger fra en BaseStation. Overvågningsapplikationer får på den måde altid at vide, når der er en ny måling fra sensornetværket, så brugeren af systemet har mulighed for til en hver tid at se den mest aktuelle måling fra sensornetværket. Vi har brugt B-processor som overvågningsapplikation, hvor overvågningen tager udgangspunkt i en 3D bygningsmodel. B-processor viser sensorplaceringer i bygningen og kobler op som consumer til en basestation, hvorfra sensormålinger modtages. Den aktuelle tilstand af en mote og de tilhørende sensorer kan undersøges ved at vælge motes direkte i brugerfladen. En

[3] Teknisk set i form af en plugin, se afsnit 4.5

alarmtilstand i en mote eller sensor vises visuelt ved en iøjnefaldende ændring af den grafiske fremtræden af moten i brugerfladen.

En applikation som foretager *beregninger* på historiske sensormålinger, skal optræde som klient til en basestation eller sensordatabase, hvorved al sensordata fra det ønskede tidsrum kan hentes. Vi har brugt B-Processor som beregningsapplikation med udgangspunkt i 3D modellen af bygningen, se afsnit 4.4 nedenfor. Når en sensor vælges i 3D brugerfladen, kan man i kontrolpanelet for sensoren hente data fra et specificeret tidsrum. De hentede målinger caches i B-Processor, så beregninger efterfølgende kan foretages på målingerne.

Overvågning af den aktuelle tilstand af sensornetværket spiller nogle gange sammen med beregninger på historiske målinger, idet en alarmtilstand kan være en følge af beregninger på historiske målinger. Det er ofte sådan at det er målingerne igennem længere tid, som er kritiske og ikke bare en enkeltstående måling. Derfor vil en overvågningsapplikation til beslutningsstøtte både have brug for aktuelle opdateringer fra basestationen såvel som adgang til historiske data fra basestationen.

4.3 B-processor generel beskrivelse

B-processor [10] er en applikation til håndtering af digitale bygningsmodeller i 3D. En digital bygningsmodel indeholder både bygningsrelevant information og geometrisk information om de forskellige bygningsdele, som indgår i bygningen såsom døre, vinduer, vægge osv. Ideen er at B-processoren kan bruges til at håndtere alle de forskellige aspekter af arbejdet med den digitale bygningsmodel: arkitekt-modellering af bygningen, ingeniørberegninger af statik, energitab, og også overvågning af bygningen, efter at den er færdigbygget og taget i brug.

B-processor består af en kerne med en plugin-arkitektur. Kernen indeholder geometriske værktøjer til modellering af en bygningens geometri. Denne geometri kobles sammen vha. objekter, som repræsenterer bygningsdele og rum i bygningen. Et klassifikationssystem muliggør at bygningsdelene kan klassificeres som døre, vinduer osv. Plugin-arkitekturen bruges til at udvide kernen med særlig funktionalitet. Det kan være særlige modellerings værktøjer til specifik anvendelsesorienteret modellering af fx facade-systemer eller badeværelser, eller beregningsværktøjer til fx energiberegninger.

I SensoByg sammenhæng har vi udvidet B-processor med sensor-netværk funktionalitet, så B-processor kan fungere som konfigureringsværktøj, overvågningsværktøj og beregningsværktøj til sensornetværk.

B-processor arbejder på digitale bygningsmodeller i et B-processor specifikt xml-baseret filformat. En bygningsmodel består af et hierarki af bygnings-specifikke objekter, som er knyttet til geometri. Den hierarkiske opdeling afspejler at sammensatte bygningsdele består af elementer, som igen består af mindre dele. En ydermur kan fx være delt i lag – teglsten, isolation og indervæg – og hver af disse lag består igen af mindre dele såsom teglsten, stolper og bjælker. Til ethvert objekt i B-processor er knyttet en flade-geometri, som består af flader, kanter og punkter i 3D.
B-Processor brugergrænsefladen består af 3 paneler og en toolbar, se for eksempel Figur 19 og Figur 22 nedenfor. Panelet til venstre er en hierarkisk liste med objekterne i bygningsmodellen. Panelet i midten viser 3D modellen og panelet til højre viser egenskaber ved det valgte objekt i hierarkiet. Toolbaren i toppen indeholder primært modelleringsredskaber til 3D modellering af bygningen.
Det er ikke her muligt at medtage en detaljeret gennemgang af brugen af B-Processor, der henvises til [10].

4.4 Eksempler på B-processor anvendelser i SensoByg

Sensor-framework integrationen i B-processor består af en generel del og en anvendelsesorienteret del. Den anvendelsesorienterede del kræver konkret implementation af "HistoryService" for at give adgang til historiske sensordata eller konkret implementation af "Adaptor" for at give adgang til live sensing service.

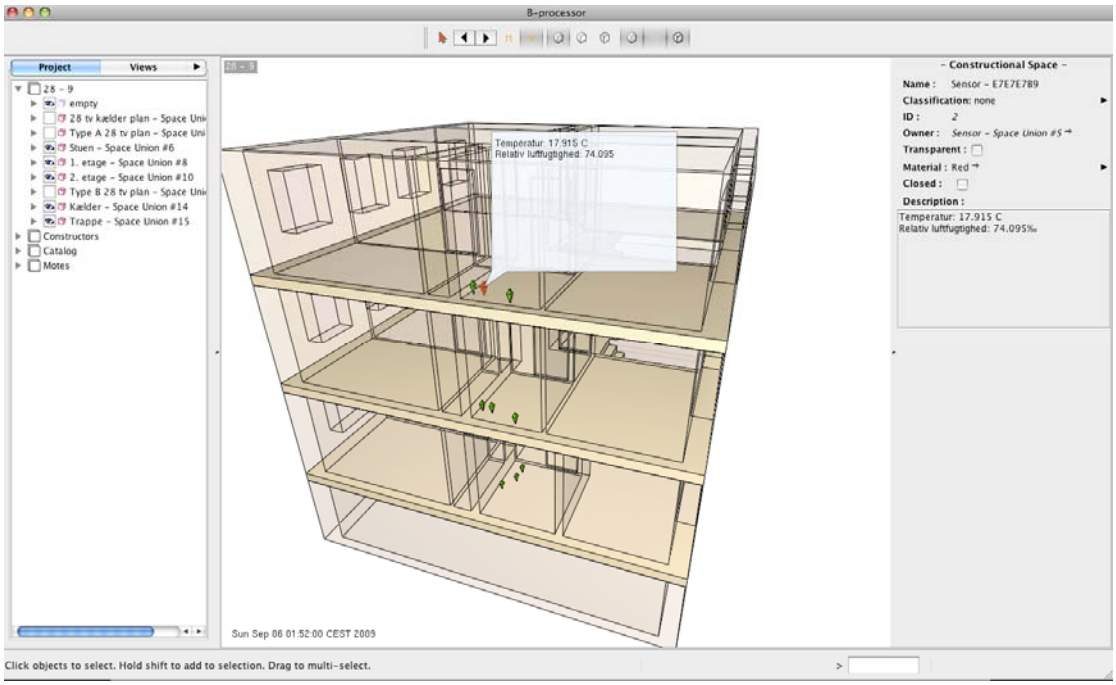
Vi har arbejdet med et antal anvendelser af sensor-frameworket i B-processor baseret på Brunatas database af sensormålinger:

- Baseret på import af data fra Excel regneark (Brønshøj Lejlighedskompleks, SensoByg sensorer)
- Baseret på kommunikation med Brunata WebMon server i hhv.
    - Alexandra kontormiljø (Brunata sensorer)
    - Privat lejlighed (Brunata sensorer)

4.4.1 Brønshøj Lejlighedskompleks

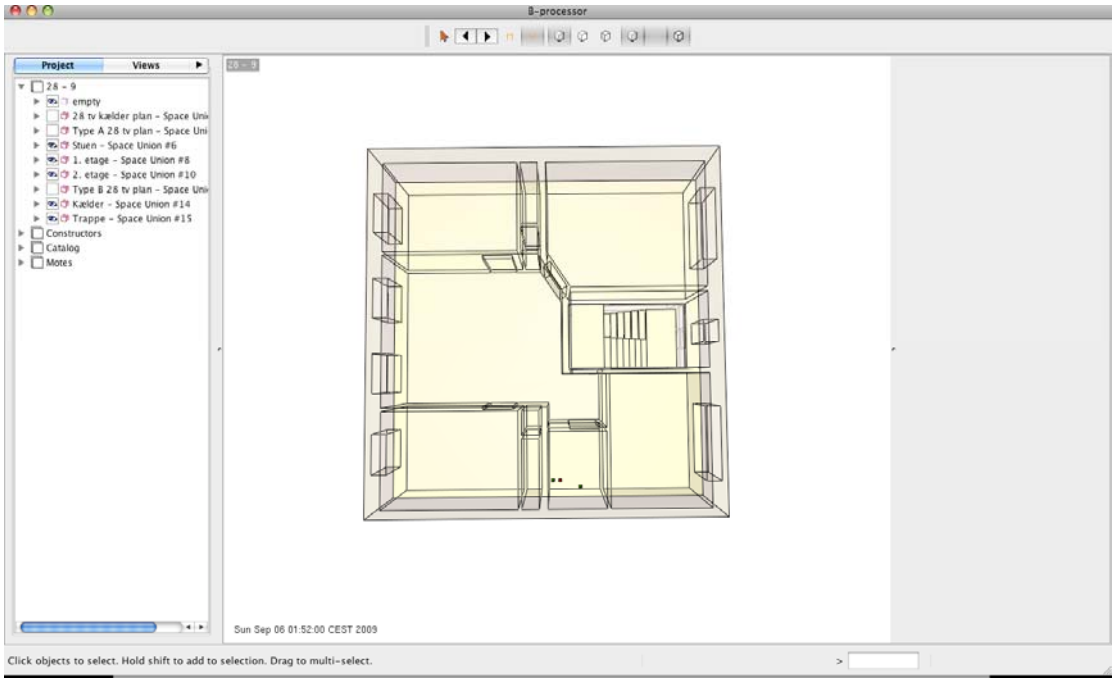
I Brønshøj Lejlighedskompleks anvendelsen, blev en enkelt opgang i komplekset modelleret i B-processor og 3 sensorer blev placeret i et badeværelse på hver etage, se nedenstående Figur 19. Sensordata blev indlæst fra en tabel fra et Excel-regneark udtrukket fra Brunatas database.

For at muliggøre at overskue alle sensorer i opgangen, vises væggene gennemsigtige, så man kan se igennem væggene samtidig med at man kan fornemme væggenes geometri. Et særligt selektionsværktøj kan bruges til at klikke gennem væggene, så en sensor kan vælges direkte i 3D modellen. Når en Sensor er valgt, vises en infoboks i form af en taleboble med information om sensorens aktuelle tilstand. En sensor er markeret med grøn, når situationen er normal og rød, hvis sensoren er i alarmtilstand.



Figur 19: Instrumenteret Brønshøj boligkompleks opgang

Der er i denne anvendelse ikke nogen live forbindelse til en basestation server. I stedet indlæses data, som nævnt, fra et Excel-regneark.
Toolbaren indeholder to knapper – tilføjet til B-processor vha. den omtalte plugin arkitektur - til at bladre frem og tilbage (tidsmæssigt) i de indlæste data. Ideen er at de indlæste data kan afspilles i en form for simulering.



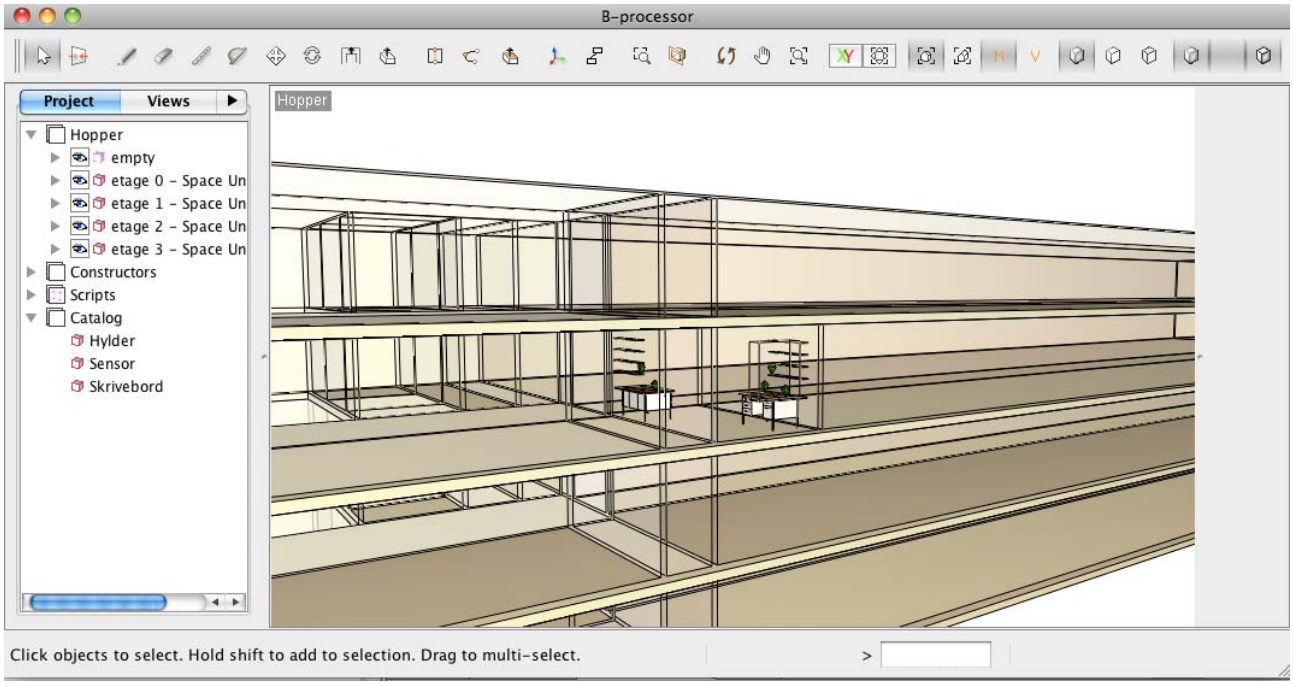
Figur 20: Brønshøj lejlighed set ovenfra

B-processors indbyggede værktøjer til 3D navigation kan bruges til at se bygningen fra forskellige vinkler. I Figur 20 ovenfor ses en enkelt lejlighed ovenfra. I en fremtidig udgave af systemet vil man have mulighed for at få forudberegnet gode kameravinkler, som der kan skiftes til automatisk.

4.4.2 Alexandra Kontormiljø

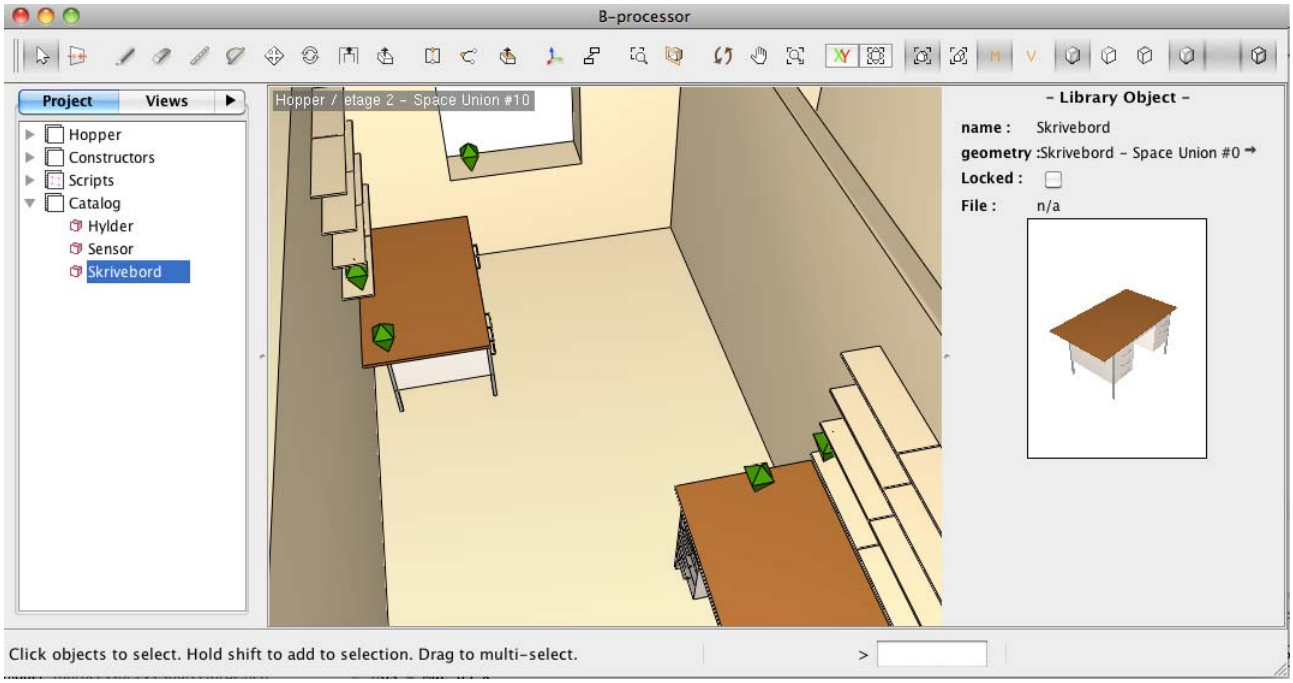
I "Alexandra Kontormiljø" anvendelsen, blev en enkelt bygning fra Alexandras kontormiljø på Katrinebjerg i Århus modelleret i B-processor, og fem sensorer blev placeret i et kontor på 2. Etage. Modellen af bygningen var forholdsvis udetaljeret, idet kun kontoret med sensorer blev modelleret – resten af bygningen var udetaljeret (se Figur 21).

Sensordata blev hentet live fra en Brunataserver (Brunatas BaseStation), som leverer data fra en Brunata database. I forsøgsopstillingen brugte vi fem sensorer fra Brunata, som måler lufttemperatur og relativ luftfugtighed. Sensorerne sender med fast interval de nyeste målinger til en Brunata **Reader** (kaldet *Lan Data Controller*), som via en tcp/ip forbindelse sender data videre til den Brunata server, som B-processor henter data fra.



Figur 21: Alexandra Kontormiljø set fra siden.

I denne anvendelse er der således en live forbindelse til en BaseStation, hvor B-processor optræder som klient. Protokollen er baseret på http, hvor data sendes i et Brunata-specifikt XML format. B-processor har mulighed for at hente data ud af Brunata databasen, men kan ikke abonnere på fortløbende opdateringer. B-processor er således en pull-klient, men viser de nyeste målinger ved periodisk at checke ved serveren om der er noget nyt.



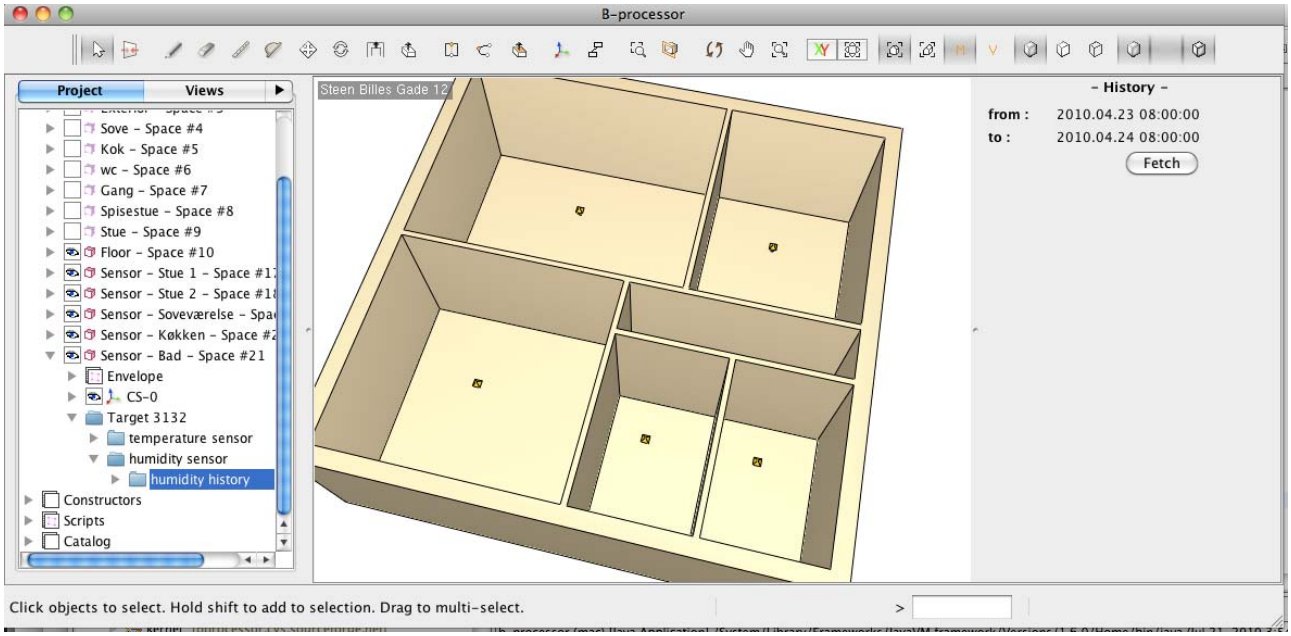
Figur 22: Kontor med sensorer.

Figur 22 viser et nærbillede af kontoret med sensorerne og møbler. Til venstre i vinduet ses et katalog, der både indeholder en sensor komponent og et par eksempler på kontormøbler. B-processor indeholder en komponent-orienteret brugerflade til at indsætte sensorer på en flade i et rum. Denne brugerflade kan også bruges til møblering: En komponent vælges i kataloget og trækkes ind i rummet, hvor komponenten kan placeres på en flade – fx gulv, væg eller et møbel. Komponent brugerflade blev i første omgang udviklet til sensorplacering, men blev udvidet til også at omfatte møbler og badeværelses-elementer. Ideen er at placeringen af fx badeværelseselementer kan have betydning for sensorplaceringen i et badeværelse. I første omgang var håndtering af komponenter meget langsom, så derfor undlod vi badeværelses-elementer i modelleringen af Brønshøj opgangen. Efterfølgende er komponent-håndteringen optimeret tilstrækkeligt til at møbler kan inkluderes i modelleringen.

4.4.3 Privat Lejlighed

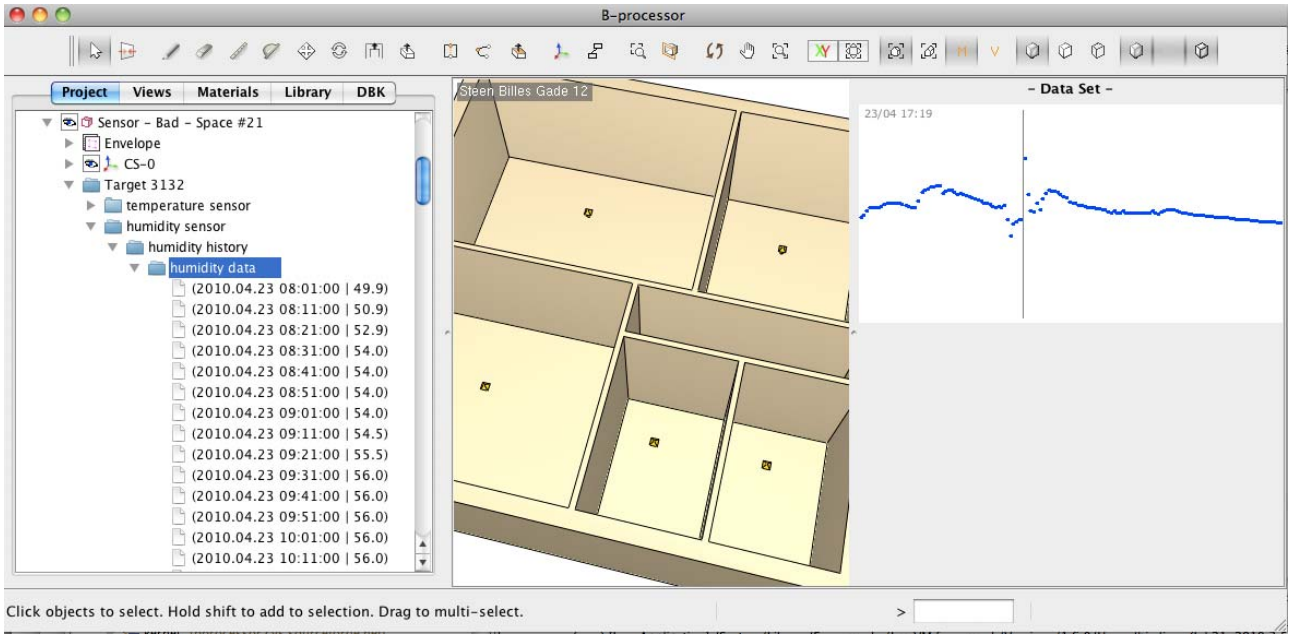
I "Privat Lejlighed" anvendelsen, blev en privat lejlighed modelleret. Modellen af lejligheden var helt udetaljeret, da kun rumopdelingen blev modelleret (se Figur 23).

Sensordata blev hentet live fra en Brunata BaseStation på samme måde som i anvendelsen "Alexandra Kontormiljø". Forskellen på denne version af systemet i forhold til den foregående version, er at implementationen nu følger det design som er beskrevet i afsnit 3.



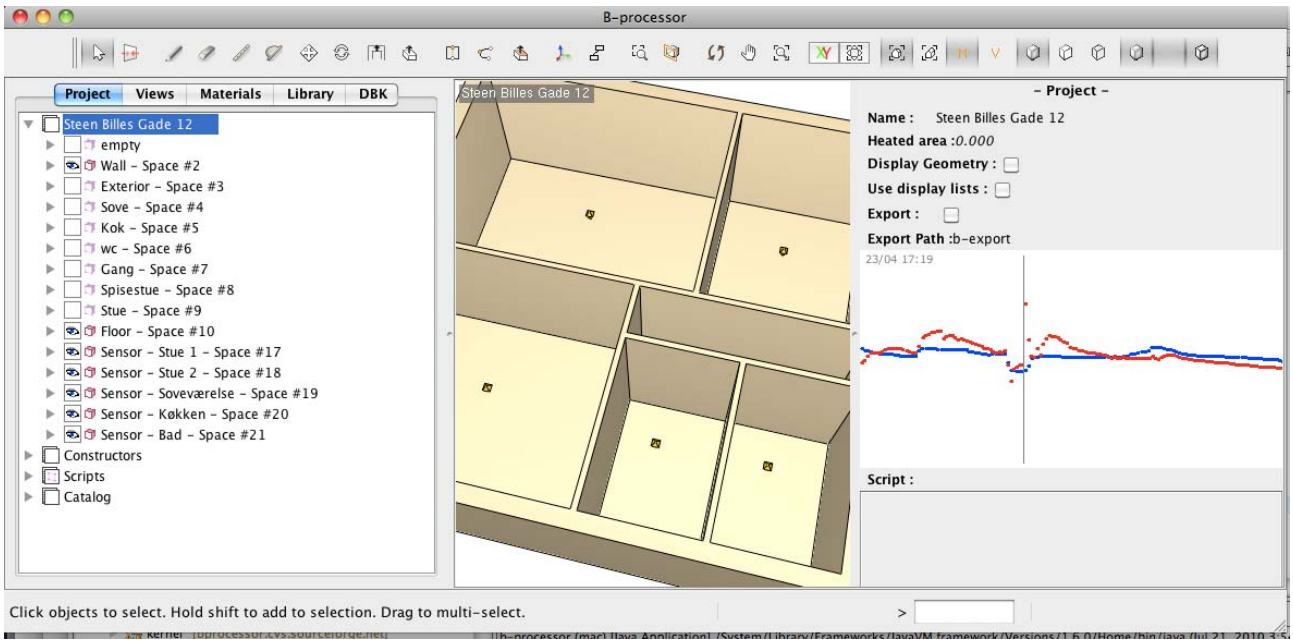
Figur 23: Model af privat lejlighed.

B-processor viser her, ligesom i de tidligere prototyper, de mest aktuelle målinger på de valgte sensorer. Men ud over dette, er der en brugerflade til at udføre forespørgsler på historiske målinger mellem to tidspunkter. I Figur 23 er man ved at foretage en forespørgsel på en fugt sensor på en mote med to sensorer: en temperatur-sensor og en fugt-sensor.



Figur 24: Sensormålinger er hentet.

I Figur 24 ses resultatet af at hente målinger fra det angivne tidsrum. Til venstre i vinduet ser man de individuelle målinger repræsenteret til inspektion. Til højre ses en graf over de valgte målinger, hvor tidspunktet med den maksimale måleværdi er markeret. I Figur 25 er vist to grafer, som er fremkommet ved at måledata er hentet for to forskellige sensorer.



Figur 25: to grafer

Ligesom i Figur 19 kunne det overvejes at integrere informationen (her graferne) i selve 3D repræsentationen, fx i en taleboble.

4.5 Værktøjer (plugins)

En bygnings-model i B-processor indeholder både bygningsrelevante objekter og geometriske objekter. De bygningsrelevante objekter består af bygningsdele (vinduer, vægge, døre osv.) og funktionsrum (stuer, kontor, badeværelse osv.). De geometriske objekter er knyttet til de bygningsrelevante objekter, og består af en flade-model med flader, kanter og punkter.

Brugerfladen er opdelt i tre paneler (se Figur 23): Hierarkisk objekt-view til venstre, 3D geometrisk view i midten, og attribut-panel til højre. Det hierarkiske objekt-view viser de bygningsrelevante objekter i bygningsmodellen. Som det ses i skærmbilledet, udgør det både funktionsrum, bygningsdele og sensorer. Det geometriske view viser bygningsmodellens geometri, og attribut-panelet viser egenskaber ved det valgte objekt i objekt-view.

B-processors brugerflade er primært udviklet til *modellering* af bygninger og derfor er de fleste værktøjer i værktøjslinjen beregnet til modellering: tegneredskaber, opmålingsredskaber, ekstrusion osv.
I SensoByg sammenhæng har vi tilføjet et antal værktøjer til B-processor brugerfladen. Disse værktøjer er i høj grad tilføjet direkte til B-processors kode-base, da det har været enklere i prototype sammenhæng. Der pågår en udsplitning af SensoByg specifik funktionalitet i form af *plugins* til B-processor kernen. Nedenfor gennemgås de SensoByg specifikke værktøjer.

4.5.1 Selektion

Vi har lavet et særligt værktøj til sensor-selektion (se Figur 19). Med dette værktøj kan man vælge en sensor direkte i det geometriske view, hvorefter en taleboble vises for den valgte sensor med sensorspecifik information.

Den almindelige måde at vælge objekter i B-processor foregår sådan at de bygningsrelevante objekter vælges i det hierarkiske objekt-view til venstre, mens de geometriske objekter (flader, kanter og punkter) vælges i de geometriske view. Denne måde at selektere geometri er naturligvis velegnet til modellering, da man i modelleringssammenhæng har et behov for at kunne vælge geometri med henblik på manipulation af geometrien. I SensoByg sammenhæng virker det forstyrrende at man kan vælge flader, kanter og punkter i bygningsmodellen, da man i forbindelse med sensorovervågning ikke er interesseret i at ændre (eller ødelægge) den eksisterende geometri i bygningsmodellen. Det nye sensor-selektions værktøj håndterer den problemstilling ved netop *ikke* at tillade selektion af geometrien (flader, kanter og punkter) men kun selektion af de relevante objekter (ved at klikke på den tilhørende geometri).

Tale-boblen viser i figuren kun ganske lidt information (kun de aktuelle temperatur og fugt målinger) – det er meningen at man i en udvidelse af systemet har mulighed for at vise mere information, som er relevant for anvendelsen, som også diskuteret i nogle af anvendelseseksemplerne ovenfor.

4.5.2 Basestation connection

Vi har tilføjet funktionalitet til B-processor som henter sensormålinger live fra en Brunata-server. Forbindelsen til Brunatas server er i indeværende version af systemet hardcoded og systemet kan derfor ikke konfigureres til at bruge andre servere. Der er to aspekter af forbindelsen til serveren, som kræver serverspecifik software:

- Format af URL: Data hentes fra serveren via en http forbindelse og det er server-specifikt, hvordan formatet er af de URL'er, der bruges til at hente data.

- Format af XML: Data returneres fra serveren, som et XML dokument i et server-specifikt format.

Den fortløbende opdatering af den nyeste sensormåling kører som en proces i baggrunden og dette værktøj manifesterer sig derfor ikke direkte i brugergrænsefladen.

Muligheden for at hente historiske data for en given sensor, manifesterer sig i brugerfladen som et kontrolpanel i attribut-panelet (se Figur 23), hvor man kan angive de datoer, som måledata ønskes mellem.

4.5.3 Play

Dette værktøj bruges til at "afspille" historiske sensormålinger. I systemet vises sensortilstanden til et givet tidspunkt, i stedet for det aktuelle tidspunkt. Systemets initialiseres til at vise sensormålinger mellem to tidspunkter og sensormålingerne for de angivne tidspunkter hentes ud af databasen. Man har så knapper i brugerfladen til at bladre frem og tilbage i tiden, hvorefter tilstanden af sensorerne opdateres med de tilhørende målinger. I Figur 19 kan man se et eksempel på brug af dette værktøj. De to "frem" og "tilbage" knapper i værktøjslinjen bladrer frem og tilbage i tiden og det aktuelle tidspunkt er angivet nederst i geometri-viewet. Taleboblen for den valgte sensor viser de sensormålinger fra databasen, som hører til det valgte tidspunkt.

4.5.4 Navigation

B-processor har et antal værktøjer til at navigere i 3D-modellen af bygningen, så man kan indstille skærmbilledet til at give den ønskede oversigt over sensorerne. Figur 19 viser en bygning set fra siden, så man kan danne sig et overblik over alle sensorer i en opgang, mens Figur 22 viser et nærbillede af et enkelt kontor, så man kan få et mere detaljeret billede af sensorplaceringen i lokalet.

De generelle værktøjer til 3D navigation består i zoom, panering og rotation. Disse tre operationer kan bruges til at anlægge en vilkårlig synsvinkel på bygningsmodellen. Derudover er der to særlige navigationsoperationer:

- **Fit-to-screen**: Indstiller skærmbilledet sådan at det valgte objekt fylder hele skærmen ud. Det valgte objekt kan fx være en bygning, et rum eller en enkelt sensor.
- **Focus-on**: Indstiller skærmbilledet, så man kigger direkte på en valgt flade. Den valgte flade kan for eksempel være en gavlflade eller en loftflade alt efter om man ønsker at se bygningen fra siden eller fra oven.

Modellen kan også indeholde et antal forud definerede kameraindstillinger, som man kan skifte i mellem ved at vælge fra en liste. Det gør det muligt eksempelvis at skifte hurtigt mellem en synsvinkel, som giver et overblik over et helt bygningskompleks, og en synsvinkel, der fokuserer på et enkelt rum.

4.5.5 Positionering af sensorer

Positionering af en sensor foregår via komponent-systemet:

- En figur tegnes i 3D vha. modellerings-værktøjerne og grupperes til et sammenhængende objekt, som kaldes en "union" i B-processor.
- Den tegnede "union" konverteres til en komponent (med en menukommando), og vil herefter optræde i et komponent-katalog i det hierarkiske objekt-view (se Figur 22).

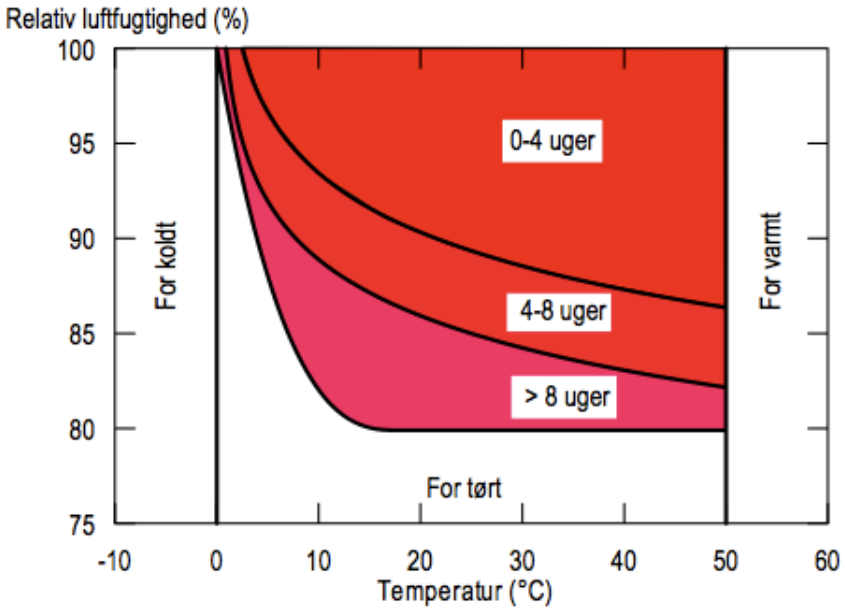
- Komponenten klassificeres som "sensor" i klassifikations-systemet og udstyres med en ekstra "mote-id" attribut, som knytter "sensoren" til en mote i sensor-netværket.
- Instanser af sensor-komponenten trækkes fra kataloget ind på en flade i bygningsmodellen og kan placeres frit på fladen.

Brugen af komponent-systemet betyder at en sensor kan gives en grafisk udseende, der matcher hvordan sensoren ser ud i virkeligheden. Hvis man ønsker at placere en sensor præcist i 3D, kan man bruge de indbyggede lineal og vinkelmåler til opmåling og markering af præcise positioner.

4.5.6 Analyse og beslutningsstøtte

I B-processor kan man også tilføje plugins, der udfører beregninger på bygningsmodellen og måledata fra sensornetværket. Feedback fra disse beregninger kan være visuel ved at der ændres farve eller tilføjes grafiske elementer til 3D visningen af bygningsmodellen. Et simpelt eksempel på visuel feedback kan ses i Figur 19, hvor en af sensorerne i 3D modellen er markeret med rød, mens de andre er markeret med grøn. Den røde markering skyldes at det aktuelle sensormålinger for temperatur og relativ fugt overskrider en bestemt grænseværdi. Feedback fra beregninger kan også være en tabel med afledte værdier, eller en graf, som man kan se et eksempel på i Figur 24, hvor grafen er placeret i attribut-panelet. Man kunne også forestille sig at placere grafen lidt tættere på 3D-modellen – for eksempel i taleboblen, som viser sensorspecifik data.

I alarm eksemplet i Figur 19, er alarm beregningen udelukkende baseret på den aktuelle måling. Beregninger kan også være baseret på sensormålinger over tid. For eksempel har SBi i delprojekt D1 beskrevet (se [12]) en proces til beregning af risiko for vækst skimmelsvamp i bolig. I Figur 26 ser man en visuel fremstilling kopieret fra [12], som viser sammenhæng mellem temperatur og relativ luftfugtighed over tid og risikoen for vækst af skimmelsvamp.



Figur 26: Visuel fremstilling som viser periodelængden med tilhørende forhold mellem temperatur og relativ luftfugtighed som skal være opfyldt førend der er risiko for vækst af skimmelsvamp.

Disse beregninger kunne implementeres i B-processor og her vil man så have mulighed for at visualisere risikoen direkte i den grafiske præsentation af bygningen i 3D-viewet. Det kunne for eksempel ske med en særlig farvelægning af de rum, hvor der er problemer.

5 Referencer

[1] SensoByg-F1-1: *Wireless sensor networks*, Morten Tranberg Hansen, Aarhus Universitet, 2010

[2] Mote – Sensor Node: http://en.wikipedia.org/wiki/Motes

[3] UML Class Diagrams - http://en.wikipedia.org/wiki/Class_diagram

[4] SensoByg tidligere systemarkitektur
http://code.google.com/p/sensobyg/source/browse/trunk/doc/architecture_description.pdf

[5] SensoByg TinyOS baserede reference implementation (open-source)
http://code.google.com/p/sensobyg/source/browse/trunk/

[6] Decision Support Systems - http://en.wikipedia.org/wiki/Decision_support_system

[7] Rambøll SMART - http://smarthome.ramboll.dk/

[8] Brunata WebMon - http://brunata.dk/online-service/webmon/

[9] Brunata WebMon Visual - http://brunata.dk/online-service/webmon-visual/

[10] B-processor: http://www.b-processor.dk/

[11] B-processor open-source: http://sourceforge.net/projects/bprocessor

[12] SensoByg-D1: *Trådløs monitering af byggeri og boliger*, SBI 2010