Sony iFast

Afstudeerverslag

Auteur
M.C. Van Dorp
Datum
10 Juni 2004
Versie
1.2

Afstudeerverslag
Sony iFast

Matthijs van Dorp

Titel	Sony iFast | Afstudeerverslag
Student	M.C. van Dorp
Onderwijsinstelling	Haagse Hogeschool
Opleiding	Informatica en Informatiekunde
Variant	MBIT
Contact docent 1	Dhr. T.H.M. Spaan
Contact docent 2	Dhr. A.A. Nederend
Bedrijfsmentor	M. Diependaal
Adres	Schipholweg 295, 1171 PK, Badhoevedorp
Telefoon	020-4499329
Email	martin.diependaal@eu.sony.com
Website	http://www.sonybiz.net/imx

Voorwoord

Tijdens mijn afstudeerperiode ben ik bezig geweest met het ontwikkelen van een applicatie die gebruikt wordt in een medische omgeving. Hierbij heb ik veel hulp gehad van collega's en andere contactpersonen die ik hierbij wil bedanken voor hun moeite. Hierbij denk ik vooral aan de leden van het ontwikkelteam van Sony Europe in Amsterdam en de werknemers van Netways ICT components bv.

Daarnaast wil ik mijn bedrijfsmentor Martin Diependaal bedanken voor zijn inzet en commentaar gedurende de hele periode.

Matthijs van Dorp

Inhoudsopgave

1. Inleiding

Dit verslag beschrijft mijn hele afstudeerperiode zoals ik die uitgevoerd heb bij Sony Europe. Tijdens deze afstudeerperiode heb ik een systeem ontwikkeld voor het opslaan van medische gegevens.

Het doel van dit verslag is de examencommissie inzicht te geven in de werkzaamheden die ik uitgevoerd heb tijdens deze afstudeerperiode. Naast werkzaamheden worden er ook beslissingen toegelicht en aan het einde is er een evaluatie van het hele traject.

Het verslag is opgedeeld in hoofdstukken die elk een onderdeel van het traject bespreken. Deze hoofdstukken lopen van het maken van de opdracht tot het implementeren op locatie. Daarnaast zijn er aan het einde extra bijlagen zoals een woordenlijst, bronnen en extra toelichting. De losse producten die ontstaan zijn tijdens het afstudeerproces zijn zijn achterin als extra bijlage bijgevoegd.

2. Organisatie

In dit hoofdstuk bespreek ik de organisatie waarin de afstudeerstage wordt uitgevoerd. Dit hoofdstuk geeft een stukje inzicht in de situatie en onderlinge relaties binnen Sony.

2.1 Opdrachtgever

Sony Business & Professional Solutions Europe, kortweg BPE, is een ondersteunend deel binnen de Sony organisatie die door heel Europa en het Midden Oosten functioneert. De tweehonderd personeelsleden die binnen de afdeling BPE werkzaamheden zijn, zijn onderling opgesplitst in zogenaamde segmenten. Een voorbeeld van een segment is 'Healtcare' of 'Retail'.

Sony Business & Professional Solutions is een onderdeel van Sony Europe dat weer deel uitmaakt van het grote Sony. Sony BPE wordt door lokale Sony vestigingen (Sony Nederland, Sony France etc) ingehuurd om ondersteuning te bieden bij projecten. Sony BPE levert hier dan expertise en relaties waarmee de projecten een stuk technische invulling krijgen.

Om aan de vraag te kunnen blijven voldoen, doet Sony BPE sinds kort ook aan kleinschalige systeemontwikkeling. Er worden applicaties gemaakt waarvan verwacht wordt dat de klanten er behoefte aan hebben. De applicaties staan nooit op zichzelf maar hebben altijd te maken met een specifieke productgroep die men aantrekkelijker wil maken.

> *Een duidelijk voorbeeld hiervan is 'Realshot Manager'. Realshot Manager, kortweg RSM, is een applicatie waarmee de digitale beveiligingscamera's van Sony beheerd kunnen worden. Doordat er samengewerkt wordt met de afdeling die de camera's maakt, heeft RSM een technische voorsprong op de concurrentie.*

Applicaties die afgerond en opgeleverd zijn worden overgedragen aan de afdeling Software binnen Sony Europe waar wordt gezorgd voor support, marketing en distributie.

2.2 Projectinrichting

Binnen Sony is het gebruikelijk een projectgroep samen te stellen met allerlei mensen die op de één of andere manier betrokken zijn. Normaliter zitten hier zowel technici als verkopers en marketingmensen in. Voor dit project is er ook zo'n team gemaakt bestaande uit:

- **Matthijs van Dorp (afstudeerstudent / uitvoerende)**
  Verantwoordelijk voor de te ontwikkelende applicatie
- **Martin Diependaal (bedrijfsmentor / technisch specialist)**
  Verantwoordelijk voor de functionele eisen en projectbeheersing
- **Gerard Bes (verkoper productgroep medical)**
  Levert functionele eisen en wensen/ideeën van klanten
- **Jeurgen Thiem (marketing manager productgroep medical)**
  Levert functionele eisen en wensen/ideeën van klanten
- **Peter Spaans (verkoper Sony Nederland)**
  Legt contact met de klanten

Naast teamleden zijn er ook een aantal externe 'adviseurs' die gevraagd kunnen worden om een mening of oplossing voor een deelprobleem. Voor dit project zijn dat:

- **Henk Demper (Systeemontwikkelaar)**
  Programmeur voor Sony Europe die assisteerde bij het programmeren van de applicatie.
- **Medewerkers Netways (systeemontwikkelaars)**
  Programmeurs die al een basis gelegd hadden voor de applicatie.
- **Ping Fung Kon Jin (Contactpersoon AMC)**
  Onderzoeker bij het AMC met een hoop praktische kennis over apparatuur en werkwijzen binnen het AMC.

Voordat de afstudeerperiode begon was het de bedoeling dat ik ondersteund zou worden door Henk Demper, het hoofd van de software ontwikkelingsgroep in Amsterdam. Vlak voor dat de periode begon werden de prioriteiten van zijn werk echter veranderd en kon hij niet genoeg tijd vrijmaken om mij tijdens het hele traject te begeleiden. Er is toen gekozen om een andere begeleider te zoeken. Dit is uiteindelijk Martin Diependaal geworden. Het nadeel van deze beslissing is dat hij niet bekend is met methoden en technieken die in een ontwikkeltraject gebruikt worden. Daarentegen heeft hij wel een hoop praktijkervaring wat dit compenseert.

2.3 Projectomgeving

Het eerste deel van het project is gewoon op het kantoor van Sony Europe in Amsterdam gedaan. Hier had ik een eigen werkplek en de benodigde computerapparatuur tot mijn beschikking. Het laatste deel van de opdracht is op een externe locatie gebeurd.

3. Opdrachtomschrijving

In dit hoofdstuk definieer ik de opdrachtomschrijving aan de hand van een aantal punten. Daarnaast ga ik even in op de vraag wat deze opdracht nou tot een goede afstudeeropdracht maakt.

3.1 Opdrachtomschrijving

De opdrachtomschrijving wordt aan de hand van een aantal punten opgesteld.

Inleiding

Sony Medical, als onderdeel van Sony Europe, levert en plaatst systemen voor de gezondheidszorg. Het betreft vooral systemen voor het presenteren van patiëntendata en sinds kort ook videobewaking. De afdeling levert apparatuur en software aan onder andere ziekenhuizen en onderzoek centra.

Probleemstelling

Bij elke implementatie krijgt men verschillende soorten input te verwerken, voorbeelden hiervan zijn seriële input, gebruikers input of informatie vanuit digitale of analoge contacten. Het probleem dat zich elke keer weer voordoet is dat er geen goede applicatie is om al deze verschillende soorten input te combineren en te representeren.

Doelstelling van de opdracht

Op het moment is er een applicatie, iFast genaamd, die door een relatie van Sony gemaakt wordt en mogelijk geschikt is. Sony heeft van de relatie de source code van het programma gekregen en kan dit nu aan gaan passen. Binnen Sony zijn hiervoor verschillende ideeën waarvan de haalbaarheid nog bekeken moet worden.

Het doel van de afstudeeropdracht is het bestaande programma te herzien, te documenteren en uit te bereiden.
Er moet functionaliteit toegevoegd worden die duidelijk wordt door interviews met Sony medewerkers en klanten van Sony.
Daarnaast moet er een gebruiksvriendelijke interface gemaakt worden die gebruikt kan worden zonder dat hier een keyboard voor nodig is.

Als er een versie klaar voor gebruik is kan deze waarschijnlijk getest worden binnen het AMC ziekenhuis in Amsterdam. Het ziekenhuis wil graag de trauma kamers beter gaan monitoren en hoopt dat met deze applicatie te kunnen doen.

3.2 Op te leveren producten

Binnen het project zijn de volgende mijlpalen te onderscheiden.

- Plan van aanpak *
- Definitiestudie *
- Detail ontwerp *
- Gebruikers handleiding
- Technische handleiding
- Een werkend systeem

Mijlpalen die met een * gemarkeerd zijn, zijn mijlpalen die alleen voor het afstudeerproces van belang zijn. Deze zullen dan ook alleen in het nederlands beschikbaar zijn. Overige documenten worden in het engels opgeleverd.

3.3 Randvoorwaarden

Vanuit Sony waren er een aantal randvoorwaarden waaraan voldaan moest worden. De meest belangrijke eis was documentatie. Alle source code moest gedocumenteerd worden en gebruikershandleidingen moesten perfect in orde en altijd up to date zijn.

Een andere eis was dat het programma in een taal gemaakt werd die de huidige programmeurs ook beheersten zodat deze later het onderhoud voor hun rekening kunnen nemen. Hierbij is de keuze op C++ gevallen. Omdat C++ een object georiënteerde taal is zal er ook object georiënteerd geprogrammeerd worden.

3.3 Afbakening functionaliteit

Ook al is de hele applicatie gericht op het zo generiek mogelijk zijn; er zijn altijd aanpassingen of uitbreidingen nodig die het pas echt goed bruikbaar maken binnen een bepaalde branche. Voorbeelden hiervan zijn extra sensorimplementaties of een nieuw soort grafiek. Al deze extra (dus niet op healthcare) gerichte dingen vallen buiten de scope van dit project. Er moet wel rekening mee gehouden worden dat deze uitbereidingen er ooit gaan komen.

4. Plan van Aanpak

In dit hoofdstuk bespreek ik het ontstaan van het plan van aanpak en dingen die hierbij kwamen kijken. Ook wordt er ingegaan op de planning van het hele traject.

4.1 Ontwikkelmethodes

Als ontwikkelmethode heb ik gekozen voor SDM. Deze methode heb ik tijdens de opleiding meerdere malen helemaal doorlopen en werkt dus voor mij vrij prettig.

SDM werkt volgens het waterval model. Dit model houdt in dat een mijlpaalproduct helemaal klaar moet zijn voordat er met het volgende product begonnen kan worden. Al deze documenten moeten consistent met elkaar zijn. Als er dus later een fout wordt gevonden in een eerder mijlpaalproduct of er moet extra functionaliteit aan toegevoegd worden moeten alle mijlpaalproducten opnieuw bekeken en aangepast worden. Hier gaat zeker bij grotere systemen een hoop tijd inzitten.

Voordelen van SDM
- Bekend met de methode
- Bewezen methode

Nadelen van SDM
- Bij wijzigingen moet het hele proces opnieuw doorgelopen worden.

4.2 Ontwikkeltechnieken

Als ontwikkeltechniek heb ik gekozen om UML te gebruiken.
Ondanks de naam UML (Unified Modelling Language) is UML geen taal maar een techniek. UML bestaat uit een aantal ongedefinieerde diagrammen en manieren van noteren. Deze diagrammen of modellen bieden een mogelijkheid om de werking, samenhang en acties van een programma te definiëren.

Ik ga de volgende UML diagrammen gebruiken
- Class diagram
- Use Case diagram
- Sequence diagram
- State diagram

Voordelen van UML
- Bekend met de techniek
- Flexibel

Nadelen van UML
- Niet helemaal toereikend
- Sommige diagrammen zijn moeilijk te lezen door leken

4.3 Planning

Om het project beheersbaar te houden is er een planning gemaakt. Deze planning is gevisualiseerd aan de hand van een zogenaamde Ghantt chart. Uit deze planning is af te lezen op welk moment men waar moet zijn in het project.

Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Inrichten werkplek																		
Inwerken en documenteren iFast																		
Maken Definitiestudie																		
Maken Basisontwerp																		
Maken mockups voor overleg																		
Programmeren systeem, inclusief testen																		
Implementatie bij bijv. AMC																		
Uitloop																		
Onderzoek doen naar DICOM																		
Werken aan afstudeerverslag																		

N.B. Deze planning is van te voren gemaakt. Er was toen nog onvoldoende informatie om het stuk 'Programmeren systeem, inclusief testen' gedetailleerder in te plannen. Deze tijd heb ik pas toen ik begon met programmeren ingepland wat resulteerde in de volgende planning.

Week	8	9	10	11	12	13	14	15	16
Mockups verwerken tot code									
Programmeren basissysteem									
Aansluiting Realshot Manager									
Ontwerpen en implementeren configuratiedialogen									
Testen en debuggen									
Implementeren medische toepassingen									
Documenteren systeem									

5. Definitiestudie

De stap na het maken van een plan van aanpak is het maken van een definitiestudie. In de definitiestudie wordt de omgeving van het systeem in kaart gebracht en worden er systeemeisen gedefinieerd.

Officieel maak je één keer een definitiestudie die het hele systeem definieert. Praktisch gezien is dat niet echt mogelijk omdat er altijd wel kleine dingen bij komen. Daarom heb ik tijdens het programmeren de definitiestudie nog een keer aangepast zodat de definities consistent zijn met het programma.

5.2 Systeemeisen

Voordat er kan begonnen worden met het definiëren van het systeem is het nodig een eisenpakket te maken waaraan de applicatie voldoet. Dit moet vrij duidelijk opgesteld worden omdat dit de basis is van je applicatie. De eisen heb ik niet zelf opgesteld maar zijn gemaakt in samenwerking met verschillende medewerkers van Sony. Zij zijn de specialisten op dit gebied en ik ben er dus vanuit gegaan dat hun mening representatief is voor de rest van de branche. Aan de hand van deze gesprekken heb ik een lijst opgesteld, opgesplitst in drie categorieën: Algemene eisen, functionele eisen en userinterface eisen.

Elke eis krijgt een eigen nummer en een soort prioriteit. ‘Basic‘ betekent dat de applicatie dit in ieder geval moet hebben om nuttig te zijn. ‘Luxe’ is een wens die erg handig zou zijn maar niet direct geïmplementeerd hoeft te worden.

Algemene eisen
Dit zijn algemene eisen die niet in de andere categorieën thuis horen.

ID	Algemene eisen	Prioriteit
A1	De applicatie moet gemaakt worden in C++	Basic
A2	De applicatie moet kunnen draaien onder Windows, OS X en Linux	Luxe
A3	De applicatie moet met een aantal verschillende DBMS-en werken	Basic
A4	Er mag geen gegevens redundantie optreden	Basic
A5	De applicatie moet zo generiek mogelijk opgezet worden voor toepassingen in andere branches	Basic

Functionele eisen
Deze eisen zeggen iets over het functioneren van het systeem

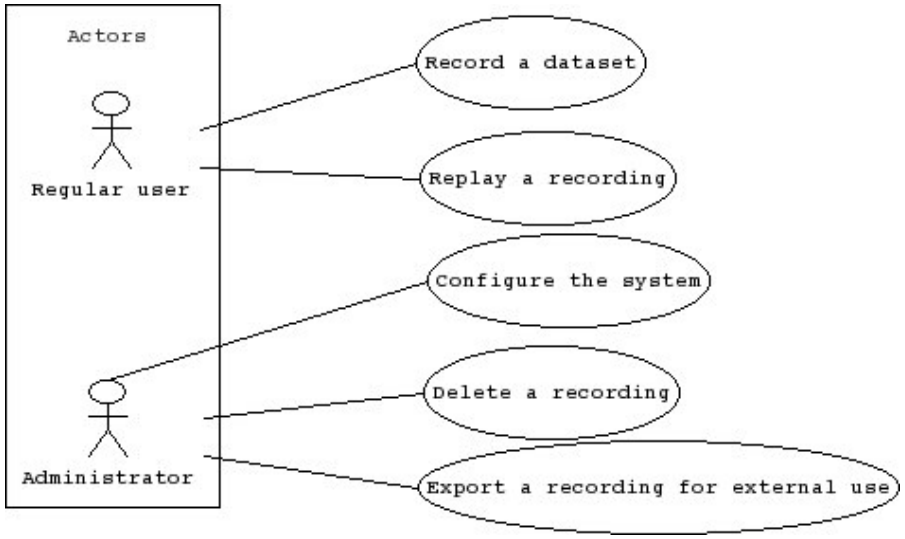
ID	Functionele eisen	Prioriteit
F1	Het moet mogelijk zijn informatie op een generieke wijze te ontvangen	Basic
F2	Het moet mogelijk zijn om ontvangen informatie op een generieke manier op te slaan	Basic
F3	Het moet mogelijk zijn informatie op een generieke wijze grafisch weer te geven	Basic
F4	Het moet mogelijk zijn om opgeslagen informatie weer terug te lezen binnen de applicatie	Basic
F5	Het moet mogelijk zijn om de applicatie op Realshot Manager aan te sluiten	Basic
F6	Het moet mogelijk zijn om de applicatie op verschillende medische apparatuur aan te sluiten	Basic
F7	Het moet mogelijk zijn om een nieuw apparaat toe te voegen	Basic

Userinterface eisen
Deze eisen zeggen iets over de gebruikersinterface zoals de gebruiker deze te zien krijgt.

ID	Userinterface eisen	Prioriteit
U1	Applicatietaal moet engels zijn	Basic
U2	Te gebruiken door beginnende computergebruikers	Basic
U3	Consistente opbouw van schermen	Basic
U4	Leuke icons	Luxe
U5	Systeem moet zowel met alleen een muis als alleen een keyboard te bedienen zijn	Luxe
U6	Applicatie moet te vertalen zijn naar andere talen	Luxe
U7	Alle informatie die getoond moet worden moet een eigen plek hebben	Basic

5.2 Use Cases

De eerste stap van het modelleren is het maken van Use Case diagrammen. Dit is een schematische weergave van de interactie van de gebruiker(s) met het systeem. Uit het diagram blijkt hoe de gebruikers het systeem zien en welke taken ze ermee uit kunnen voeren.



Use-case diagrammen

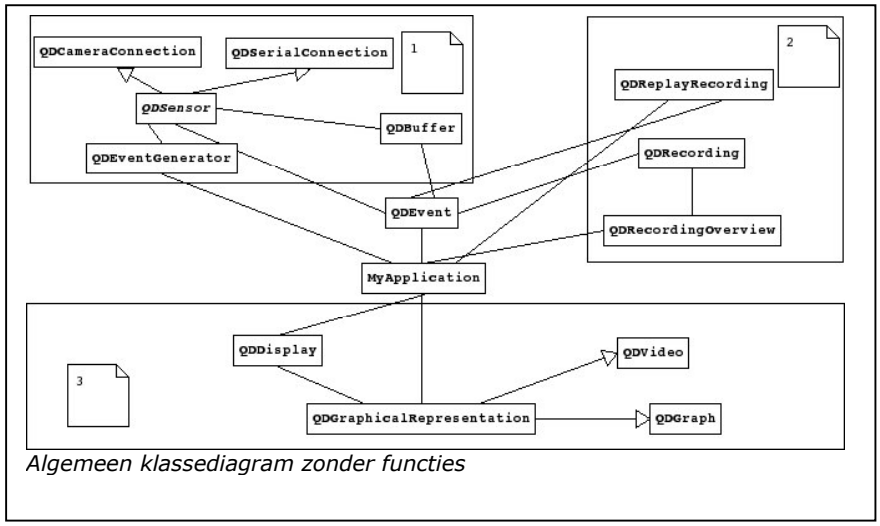
In het diagram komen twee verschillende gebruikers voor. Een regular user die elke dag met het systeem werkt en een administrator die het systeem configureert en overbodige recordings weggooit.
De rol van gewone gebruiker en administrator kan door één persoon vervuld worden.

5.3 Klassendiagram

Nadat de Use Case diagrammen af zijn kan er begonnen worden met het volgende model: het klassendiagram. Uit een klassendiagram blijkt welke onderdelen er zijn en welke functies deze hebben. Ook wordt hierin aangegeven welke onderdelen of klassen een relatie, en wat voor soort relatie, deze met elkaar hebben.

Een klassendiagram wordt gemaakt door een lijstje van zelfstandig naamwoorden te maken die betrekking hebben op het onderwerp. Als alle woorden genoteerd zijn wordt er een soort selectie gemaakt. Woorden die een groep van informatie of object aangeven worden een klasse. Woorden die een eigenschap of data element voorstellen worden eigenschappen van een object.

Al deze klassen vormen één groot diagram wat op den duur slecht leesbaar is. Daarom heb ik besloten de diagrammen op te splitsen in drie afzonderlijke delen zoals rechts te zien is.



Algemeen klassediagram zonder functies

Deel 1
Het stuk van de applicatie dat op een generieke manier de informatie verzamelt en opslaat in een buffer

Deel 2
Het stuk van de applicatie dat de events opslaat in een file en weer terug kan lezen

Deel 3
Het stuk van de applicatie Dat de events grafisch weergeeft in de vorm van een grafiek of videobeeld.

Deze drie losse onderdelen komen bij elkaar in het hoofdprogramma waarin ze met elkaar verbonden worden.

Het diagram zoals dat rechts staat is een versimpeld klassendiagram. Normaliter worden ook alle functies van een klasse getoond. In dit geval zou dit een veel te groot diagram opleveren. Daarom heb ik ervoor gekozen van elk los onderdeel een apart diagram te maken waarin wel alles getoond wordt.

Ik wil dit graag illustreren aan de hand van het diagram van ‘Deel 1’.

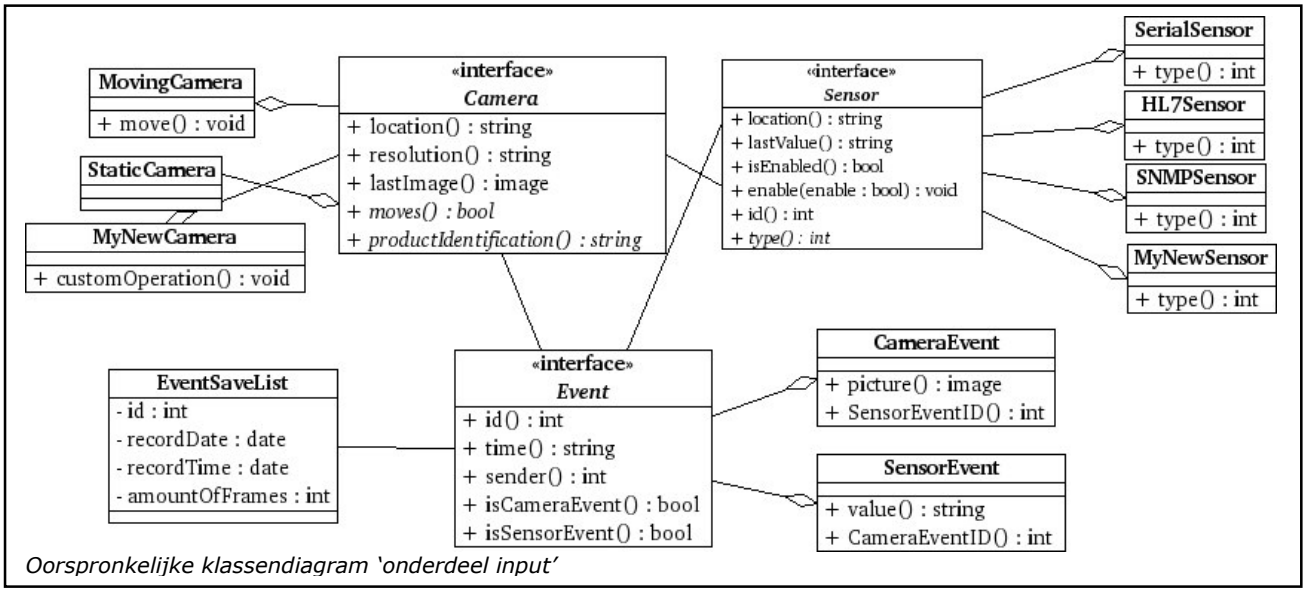
5.3.1 Eerste ontwerp

Het klassendiagram waar ik mee begon, was gebaseerd op de codebase die Sony overgenomen had van Netways. Hier zijn dus ook de namen en functies uitgekomen. Hier heb ik dus weinig zelf aan bedacht.

Het diagram stelt een stuk van het systeem voor dat verantwoordelijk is voor het ontvangen van een event. Dit event wordt vervolgens op een éénduidige manier beschikbaar gemaakt voor de rest van het systeem.

Er zijn twee verschillende bronnen van input namelijk camera's en sensoren. Dit zijn allebei interfaces zoals dat in UML heet. Elke implementatie van of een camera of een sensor overerft alle eigenschappen waardoor er een standaard manier van aanspreken ontstaat.

De camera's en sensors creëren allemaal events die beschikbaar zijn voor de rest van de applicatie(niet getoond in dit diagram). Daarnaast kan een event aan een EventSaveList gekoppeld worden en dus opgeslagen worden in een file of database.



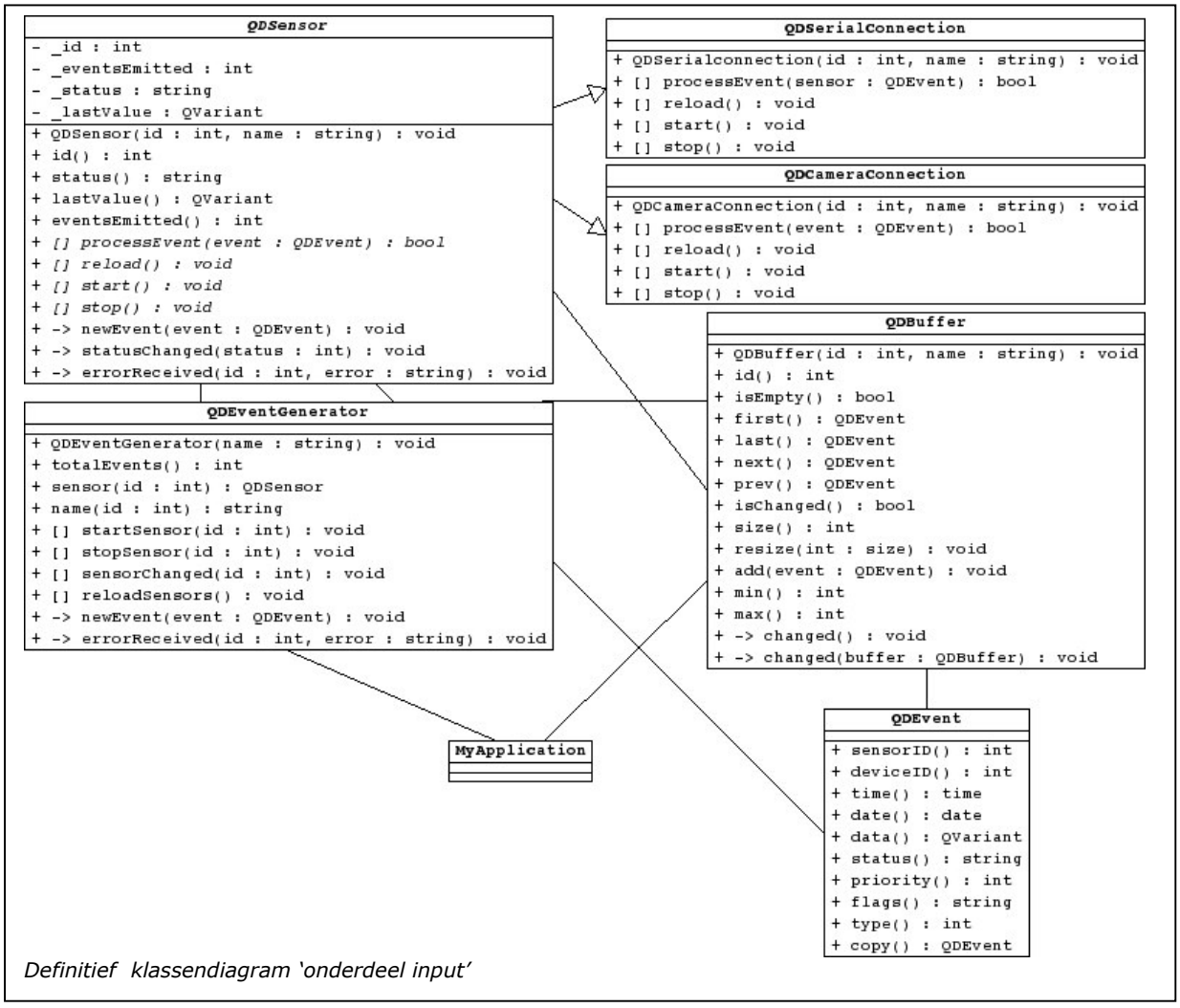
Oorspronkelijke klassendiagram 'onderdeel input'

Na er nog eens goed naar gekeken te hebben heb ik besloten om het diagram aan te gaan passen. Ik was niet tevreden over de manier van communiceren en de functionaliteit van het model was niet voldoende om het goed te kunnen gebruiken. Er was geen mogelijkheid om een aantal events tijdelijk in het geheugen te houden om er bijvoorbeeld een grafiek van te maken of een lange termijn gemiddelde.
Ook was het diagram niet generiek genoeg. Er werd onderscheid gemaakt tussen een event dat uit een sensor kwam en een event dat uit een camera kwam. Hierdoor neem je een stukje flexibiliteit weg wat niet de bedoeling is.

5.3.2 Tweede ontwerp

Aan de hand van de fouten uit het eerste model en wat voortschrijdend inzicht ben ik gekomen tot het tweede model. Dit model is completer en maakt geen onderscheid tussen de verschillende soorten sensoren. Dit heeft als nadeel dat de naamgeving soms wat cryptisch is. De basis klasse voor het genereren heet namelijk QDSensor. Dit zou betekenen dat alle 'kinderen' van deze klasse ook sensoren zijn wat een beetje vreemde benaming is voor een camera.
De naamgeving is voor een deel nog steeds gebaseerd op de oude codebase die niet zomaar weggegooid kan worden.
Namen van nieuwe klassen en extra toegevoegde functies zijn bedacht op de manier zoals die in hoofdstuk 5.3 besproken is.

Naast het extra stukje flexibiliteit is er nu ook de mogelijkheid om de events voor langere tijd in het geheugen op te slaan via een soort van buffer. Ook is het nu mogelijk een onbeperkt aantal sensoren te definiëren die aan te spreken zijn via één centrale klasse (QDEventGenerator).



Definitief klassendiagram 'onderdeel input'

Ik heb er voor gekozen om de sensoren op een éénduidige manier aan te spreken en heb hier in het ontwerp al rekening mee gehouden. Onafhankelijk van hoe de sensor implementatie intern werkt zijn de in- en uitvoer stromen gelijk aan elke willekeurige andere implementatie. Dit zorgt voor een stuk abstractie waardoor nieuwe sensor implementaties makkelijk te verwerken zijn.
Het idee kreeg ik hiervoor tijdens de voorbereiding van de realisatie. Door voor deze implementatie te kiezen is het in de toekomst mogelijk een plugin systeem toe te voegen waar functionaliteit gewoon geladen wordt als dit nodig blijkt te zijn.

In het model wordt er gebruik gemaakt van overerving, één van de grootste voordelen van objectgeoriënteerd programmeren. Overerving betekent dat een '*child*' alle publieke functies krijgt van zijn '*parent*' plus alle functies die voor de child klasse zelf gedefinieerd zijn. Dit is te zien in de klassen QDSensor en QDSerialConnection. In de definitie van QDSensor zijn er vier functies cursief gedrukt. Deze vier functies worden in QDSerialConnection opnieuw geïmplementeerd.
Elke nieuwe sensorimplementatie die de volgende vier functies heeft en overerft van QDSensor is geschikt om te gebruiken binnen de applicatie.

```
virtual bool processEvent( QDEvent* event);
virtual void reload();
virtual void start();
virtual void stop();
```

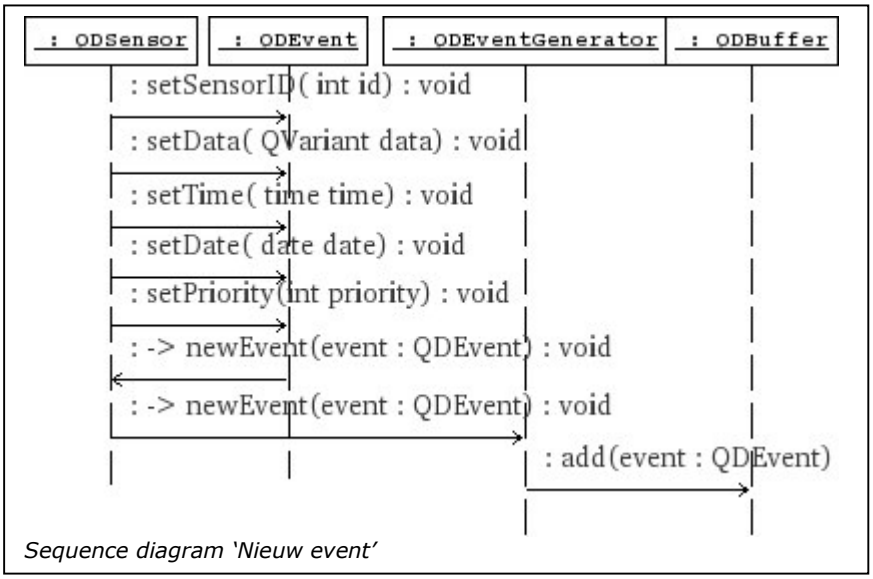
Ook binnen de klasse QDEvent is een stuk abstractie aanwezig. Het dataveld kan niet alleen maar getallen of tekst aan maar kan vrijwel alle vormen van data aan, inclusief plaatjes en videobeelden.

5.4 Sequence diagram

Een sequencediagram geeft een overzicht van de communicatie tussen de verschillende klassen. Een sequencediagram heeft een directe relatie met het klassendiagram. Ze worden ook tegelijkertijd gemaakt. Als de klassennamen duidelijk zijn moeten er een manier bedacht worden voor de communicatie die terug te vinden is in het sequencediagram. De namen die voor deze communicatie gebruikt worden komen terug in het klassendiagram.

De werking van een sequencediagram is vrij simpel. De klasse helemaal links in het diagram begint met informatie op te vragen of te geven waarbij elke stap een pijl is met de aangeroepen functie. Daarna gaat de informatie naar de andere klassen toe om uiteindelijk rechts onderin te eindigen.

Dit wil ik graag illustreren met het volgende voorbeeld:
Dit diagram is een overzicht van de communicatie die optreedt als een sensor een nieuw event genereert. Het event wordt vervolgens doorgegeven en in de juiste buffer opgeslagen.



Sequence diagram 'Nieuw event'

N.B. Er is express voor gekozen om de extra stap via QDEventGenerator in te bouwen. Op deze manier is het ook mogelijk om indien nodig zonder een buffer te werken.

Uit het sequencediagram is te zien welke klassen contact met elkaar hebben. Dit is handig om te weten als er een klasse aangepast moet worden. Dan is direct duidelijk welke klassen er gecontroleerd moeten worden op een correcte werking nadat de aanpassing heeft plaatsgevonden.

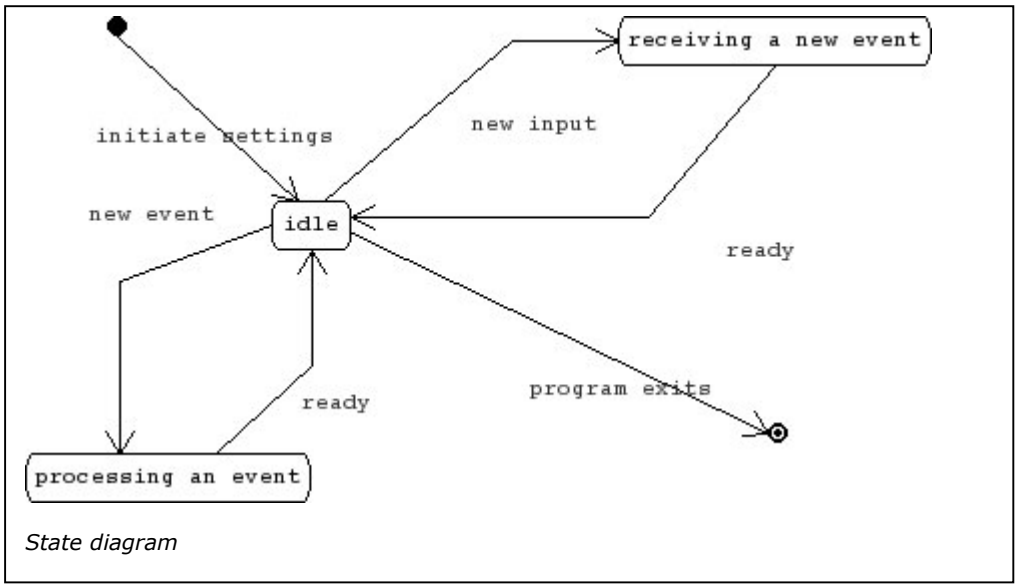
6. Basisontwerp

In dit hoofdstuk bespreek ik het maken van het basisontwerp. Het basisontwerp bestaat uit State diagrams, een database ontwerp en de beschrijving van de toekomstige werkomgeving.

6.1 State diagram

De laatste stap is het maken van een State diagram of toestandsdiagram. Een toestandsdiagram beschrijft de verschillende toestanden waarin een (deel van een) programma zich kan bevinden. Voorbeelden van een toestand zijn "ontvangen van data" en "verwerken van data".
Uit een toestandsdiagram blijkt welke toestanden er voor kunnen komen maar nog belangrijker is van welke toestand er naar welke toestand gegaan kan worden. Zeker in complexe situaties kan dat een hoop duidelijkheid geven.

In het volgende diagram wordt duidelijk hoe het 'informatie-vergaar' deel van de applicatie werkt.



State diagram

Omdat het hele proces event gedreven is , is de normale staat 'idle'. Als er nieuwe invoer is wordt deze verwerkt en als er een nieuw event is dat verwerkt moet worden (om bijvoorbeeld terug te sturen naar een apparaat) dan wordt dit afgehandeld.

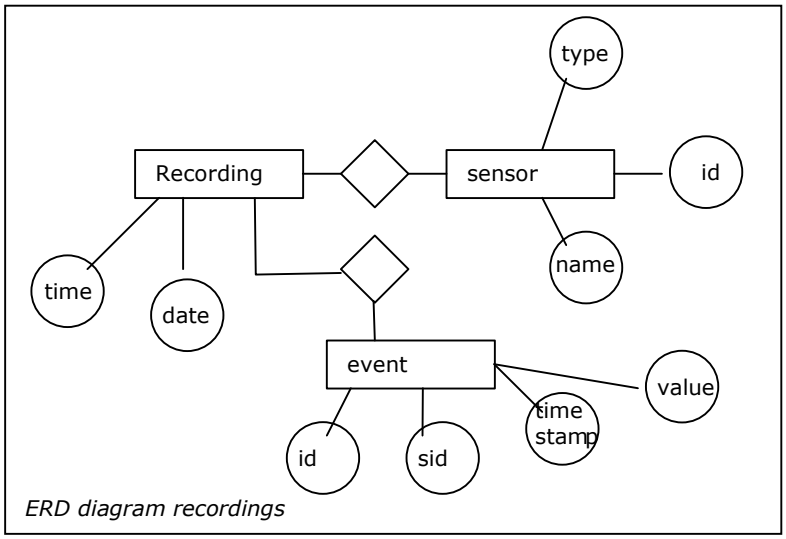
6.2 Database ontwerp

Binnen elke applicatie is er wel iets wat opgeslagen moet worden. Dit kunnen configuratiegegevens zijn, meetwaarden of uitkomsten van berekeningen. Zo ook in deze applicatie. Elk type sensor heeft zijn eigen waarden die nodig zijn voor het functioneren.

Zo ook deze applicatie. Meetwaarden die worden ontvangen moeten worden opgeslagen in een formaat dat handig is te versturen en terug te lezen. Ik heb hier gekozen voor een XML file. Dit zorgt dat ook externe applicaties de mogelijkheid hebben om deze informatie in te lezen en te verwerken. Daarnaast is het makkelijk uit te bereiden als er veranderingen komen in de manier van opslaan of de data behoefte.

Omdat dit opgeslagen moet worden in een file is er een ERD diagram gemaakt dat er als volgt uitziet:

In één recording kunnen één of meerdere sensoren voorkomen. Elk van deze sensoren kan nul of meerdere events hebben die allemaal in de recording opgenomen zijn.



ERD diagram recordings

```
- <iFast>
    <time value="16:21:24" />
    <date value="Mon May 10 2004" />
  - <sensors>
      <sensor id="0" name="Serial sensor 1" type="serial" />
      <sensor id="1" name="Serial Sensors 2" type="serial" />
    </sensors>
  - <events>
      <e id="1" sid="0" ts="58884625">59</e>
      <e id="2" sid="1" ts="58884655">57</e>
    </events>
  </iFast>
```

XML representatie van het ERD diagram

Het aantal events dat in één file past is theoretisch gezien niet gelimiteerd. Het is alleen niet aan te raden de bestanden groter te laten worden dan 100MB. Als files zo groot worden, worden ze onhandelbaar en lastig in te lezen.

7. Realisatie

Het realiseren van de applicatie nadat deze op papier werkend gemaakt is bestaat uit verschillende stappen. Eerst wordt er een schermontwerp gemaakt. Daarna wordt de basisfunctionaliteit geïmplementeerd waarna het hele systeem getest en bekeken wordt door de eindgebruikers. Aan de hand van commentaar en opmerkingen wordt de applicatie aangepast.

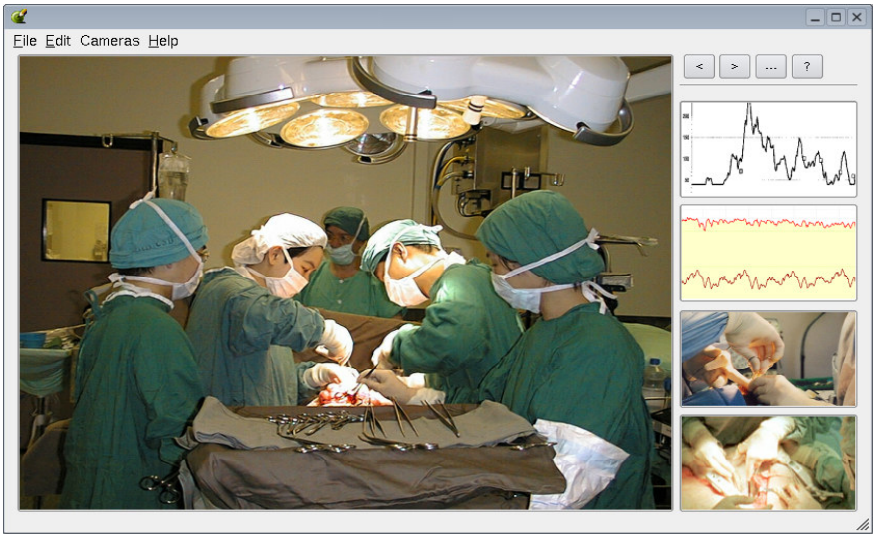
7.1 Schermontwerp

Het eerste grafische ontwerp maken is een vrij lastige klus. Je weet nog niet precies wat de gebruikers voor interface willen en je moet dus zelf wat bedenken aan de hand van voorbeelden van programma's van concurrenten. Ik heb daarom eerst met behulp van een tekenprogramma wat schetsen zitten maken. Het resultaat hiervan lijkt op de echte applicatie maar is niet functioneel. Zo'n schets wordt een mockup genoemd. In het schermontwerp wordt vastgelegd **wat** er weergegeven wordt en het accent ligt minder op **hoe** dat gebeurd.

Bij het schermontwerp moet er rekening gehouden worden met de volgende eisen die aan de interface gesteld worden:

ID	Userinterface eis	Prioriteit
U1	Applicatietaal moet engels zijn	Basic
U4	Leuke icons	Luxe
U6	Applicatie moet te vertalen zijn naar andere talen	Luxe
U7	Alle informatie die getoond moet worden moet een eigen plek hebben	Basic

Omdat de applicatie eerst voor Healthcare gebruikt zal gaan worden en daarna pas in een andere branche heb ik gekozen om een interface te ontwerpen die aansluit bij bestaande medische applicaties en hardware. Het idee is dat je via een touchscreen display of een muis op de grafieken aan de rechterkant kunt duwen. Dan wisselt deze van plaats met de grote grafiek aan de linkerkant. Grafieken en videobeelden zijn van gelijke orde en kunnen dus allemaal vergroot en verkleind worden.

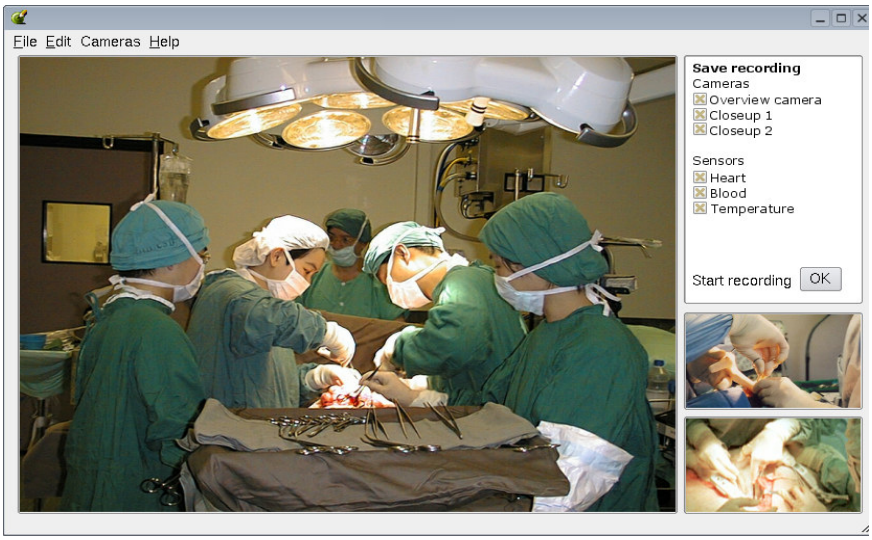


Eerste ontwerp interface overzichtscherm

Uit gespreken met de medisch specialisten bij het AMC ziekenhuis bleek dat de informatie tijdens een operatie sterk wisselt. Afhankelijk van het stadium van de operatie

zijn er steeds andere gegevens nodig om te controleren of de patiënt in goede staat verkeert. Dit bracht mij op het idee om een aantal kleine en één grote grafiek te maken waartussen makkelijk geswitcht kan worden.

In het eerste ontwerp zat ook het idee om zonder popup vensters te werken. Een popup venster is een venster dat voor de applicatie geplaatst wordt en gebruikt wordt om een vraag te stellen of informatie te geven.



Eerste ontwerp interface recording scherm

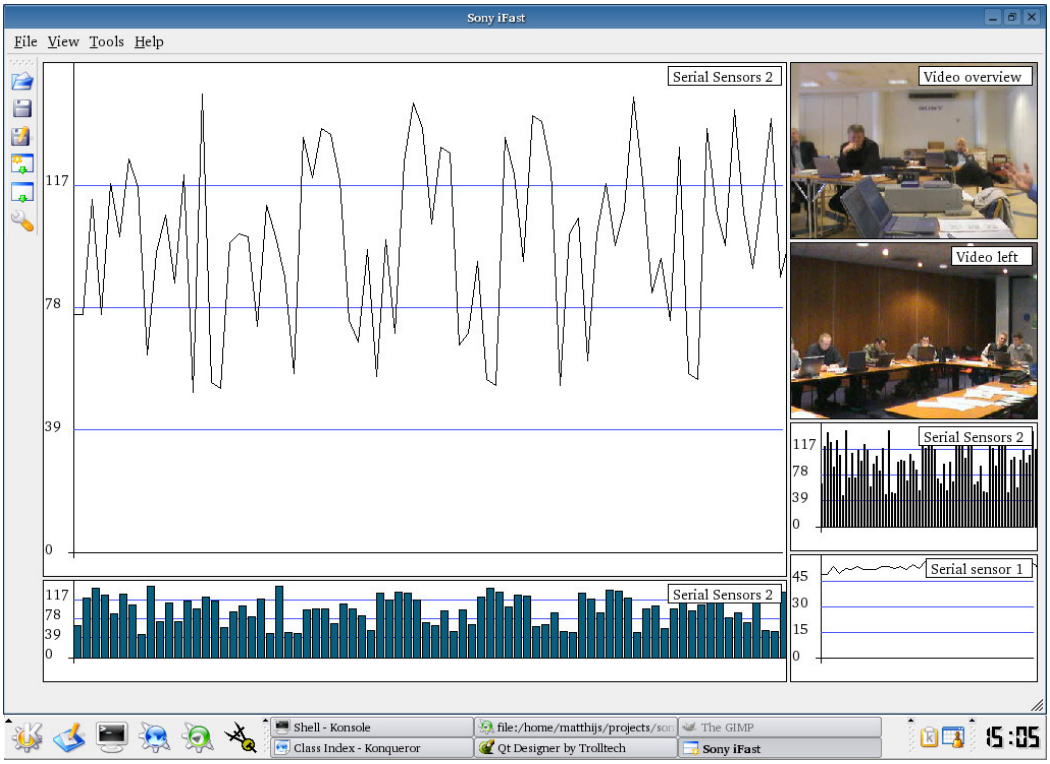
Voor een gevorderde gebruiker is het gebruik van popup schermen aan te raden. Doordat het een apart scherm is wordt duidelijk afgebakend wat bij elkaar hoort. Een beginnende gebruiker kan echter makkelijk 'verdwaald' raken in alle schermen.

De vragen worden nu niet via een los scherm maar in de bestaande interface gesteld. Dit heeft als voordeel dat de gebruiker altijd weet waar hij is in de applicatie. Het heeft als nadeel dat het erg beperkt is omdat er weinig ruimte beschikbaar is. Dit kan opgevangen worden door ook de plek van de grote grafiek beschikbaar te stellen.

7.2 Tweede grafische ontwerp

Aan de hand van de mockups heb ik een aantal gesprekken gehad met Sony medewerkers die daar hun mening over gegeven hebben. Deze informatie is verwerkt in een tweede grafisch ontwerp wat uiteindelijk geïmplementeerd is.

Het overzichtsscherm, of de standaard lay-out, is iets veranderd. Onder de grote grafiek aan de linkerkant is een extra grafiek gekomen. Dit heeft als reden dat veel grafieken en videobeelden een breedte/hoogte verhouding hebben van 3:4. Dit zou betekenen dat er een deel van het scherm niet gebruikt wordt wat niet zinvol is.

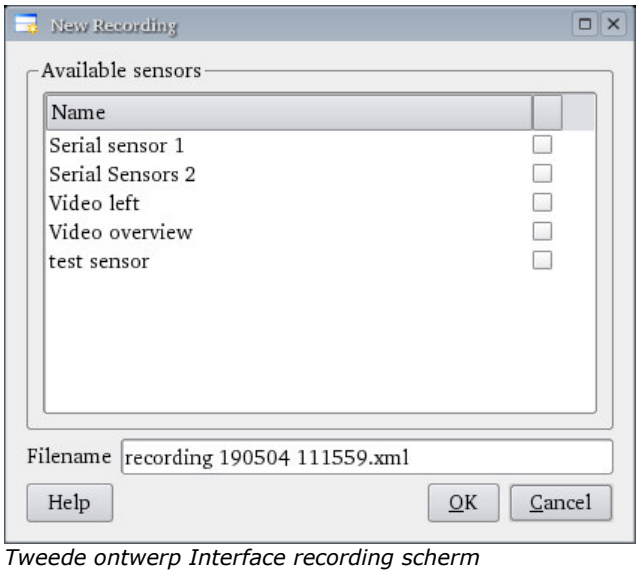


Tweede ontwerp interface overzichtscherm

Het extra veld dat hiermee gecreëerd is gedraagt zich anders dan andere velden. Zo kan het alleen grafieken weergeven en geen videobeelden. Dit heeft te maken met de beeldverhoudingen die niet gerealiseerd kunnen worden. Ook kan de grafiek niet van plaats gewisseld worden met een andere grafiek.

Ook het idee om geen popup scherm te gebruiken is gesneuveld. Vooral het gebrek aan ruimte was een erg groot probleem in sommige configuratievensters.

Als alternatief is er gekozen om toch popup schermen te gebruiken maar deze op een standaardmanier in te richten zodat de gebruiker altijd weet waar het scherm voor dient. In een configuratiedialoog is er ook altijd een button “Help” aanwezig die direct verwijst naar een relevante help pagina.



Tweede ontwerp Interface recording scherm

7.3 Usability

Nadat er een schermontwerp gemaakt is waarin aangegeven is wat er precies getoond gaat worden is het tijd om na te denken over de manier waarop dit getoond gaat worden. De term die hiervoor vaak gebruikt wordt is '*usability*'. De mate van usability of bruikbaarheid geeft aan hoe makkelijk een applicatie te gebruiken is voor beginners en hoe intuitief een applicatie werkt. Vooral het eenvoudig configureren van een applicatie verbetert de bruikbaarheid van een applicatie buitengewoon.

Bij het verbeteren van de usability moet er gekeken worden naar de volgende punten:

ID	Userinterface eis	Prioriteit
U2	Te gebruiken door beginnende computergebruikers	Basic
U3	Consistente opbouw van schermen	Basic
U5	Systeem moet zowel met alleen een muis als alleen een keyboard te bedienen zijn	Luxe

Binnen mijn eigen applicatie heb ik nagedacht over hoe ik de applicatie bruikbaar kan maken voor de beginnende gebruiker. Dit vergt nogal wat denkwerk en inlevingsvermogen omdat je je eigen werk zo goed kent dat alles logisch is. Een gebruiker kan daar echter heel anders over nadenken. Om niet mezelf maar de gebruiker de keuze te geven wat voor configuratie-items hij/zij wil zien heb ik het op een andere manier opgelost.

In het framework dat ik gebruik voor het ontwikkeltraject zitten verschillende programma's waaronder een grafische interfaceontwerp programma. Met behulp van dit programma kan er op een eenvoudige wijze een interface ontworpen worden en kan deze geëxporteerd worden naar een XML bestand. Het programma kan vervolgens tijdens het opstarten deze XML file inlezen en op basis van de inhoud een interface weergeven.Als er dus een wijziging in de interface is hoeft de applicatie zelf niet aangepast te worden. Het grote voordeel hiervan is dat een beheerder met behulp van dit programma alle configuratieopties kan verwijderen die de gebruiker niet nodig heeft. Op deze manier is er een minimum aan configuratieopties aanwezig waardoor de gebruiker makkelijker met de applicatie om kan gaan. Voor configuratieopties die niet aanwezig zijn is een standaard waarde gedefinieerd zodat de applicatie wel normaal blijft functioneren.

Met behulp van deze mogelijkheid is de applicatie weer een stuk generieker te maken doordat de gebruiker/beheerder de interface zelf aan kan passen.

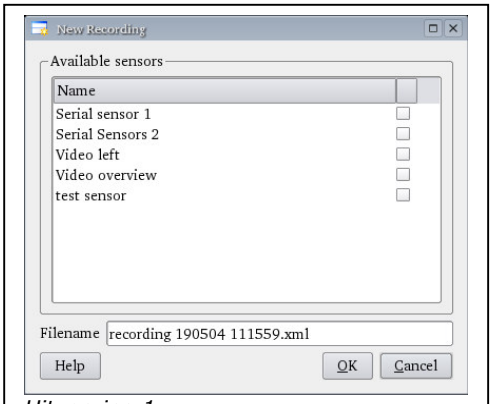
Rechts is een stukje uit zo'n XML file getoond. Hierin wordt de soort, naam, plaats en bijbehorende tekst gedefinieerd. Dit is gewoon leesbaar en zelfs met de hand aan te passen (indien nodig).

```
<widget class="QCheckBox">
  - <property name="name">
      <cstring>compressOutput</cstring>
    </property>
  - <property name="geometry">
    - <rect>
        <x>11</x>
        <y>273</y>
        <width>305</width>
        <height>23</height>
      </rect>
    </property>
  - <property name="text">
      <string>Compress output</string>
    </property>
</widget>
```

Fraament uit een XML file

Een ander voorbeeld is het simpel kunnen vertalen van het programma. Door de bijschriften en meldingen te vervangen door een andere tekst is de applicatie makkelijk van het engels naar het nederlands, duits of frans te vertalen.

Om de techniek en achterliggende gedachte nog iets te verduidelijken een kort voorbeeld aan de hand van de dialoog "Create a new recording". Deze dialoog zoekt alle gegevens bij elkaar die nodig zijn om een nieuwe recording aan te maken en stelt vragen die daarbij relevant zijn.

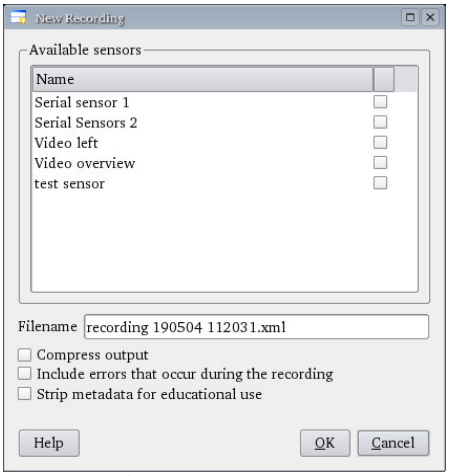


Uitvoering 1

Uitvoering 1

De basis versie kan alle informatie verzamelen die nodig is om een simpele recording te starten. Er kan gekozen worden welke sensoren er wel of niet opgenomen worden en in welke bestand ze opgeslagen worden.

Deze versie is voor de beginnende of gewone gebruiker die verder geen extra configuratieopties nodig heeft.

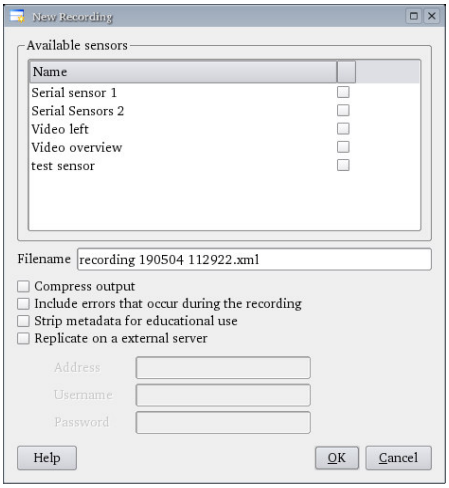


Uitvoering 2

Uitvoering 2

Deze dialoog kan alles wat de basisversie ook kan maar biedt een aantal extra configuratie opties die handig kunnen zijn tijdens het gebruik ervan. Deze dialoog is bedoeld voor de gewone of gevorderde gebruiker. Er wordt uitgegaan van enige kennis van het programma.

Uitvoering 3
Deze dialoog kan alles wat de basis en gevorderde versie ook kan maar biedt nog een extra mogelijkheid die een normale gebruiker waarschijnlijk nooit zal gebruiken maar voor een gevorderde gebruiker of systeembeheerder wel degelijk nuttig kan zijn. Er wordt van uitgegaan dat men redelijk wat kennis van het programma heeft.



Uitvoering 3

Uitleg
De drie dialogen die getoond zijn, zijn allemaal ongeveer hetzelfde en dienen voor dezelfde actie. Maar door een opties wel of niet te tonen is de applicatie bruikbaar door beginnende gebruikers en voelen gevorderde gebruikers of experts zich niet gelimiteerd in hun mogelijkheden.

7.4 Documentatie

Tijdens het schrijven van een programma is het nodig om te documenteren wat er precies gebeurt binnen een functie en wat een functie überhaupt doet. Zeker na een jaar weet vrijwel niemand meer wat een bepaalde functie doet. Omdat een applicatie onderhoudbaar moet blijven ben ik vanaf het begin af aan bezig geweest met documenteren. Zowel de werking van de functie als de functie zelf wordt beschreven. Hier is een standaard manier voor. Als je gebruik maakt van deze manier dan is het mogelijk productdocumentatie te genereren aan de hand van de files met de definities. Hiervan een voorbeeld.

De functie bool isRecording() van de klasse QDRecordingOverview bekijkt alle aanwezige recordings en checkt of één van deze recordings nog bezig is met opnemen. Dit is handig om te weten als je de applicatie af wil sluiten. De definitie van de functie is opgenomen in `recordingoverview.h`. De uitwerking van de functie staat in `recordingoverview.cpp`.

`recordingoverview.h`

```
/**
* Call this function to see if there is an active recording that is currently recording
* @return amount of recordings
*/
bool isRecording();
```

`recordingoverview.cpp`

```
bool QDRecordingOverview::isRecording()
{
    QDRecording * tmp;
    // cycle trough all the available recordings and see if one is recording
    for ( tmp = _list.first();  tmp; tmp = _list.next() )
    {
            if ( tmp->state() == QDRecording::Recording)
                    return TRUE;
    }
    return FALSE;
}
```

Documentatie

bool **isRecording** ()

Call this function to see if there is a recording that is currently recording

Returns: TRUE if there is a recording which is recording at the moment, FALSE otherwise

De informatie uit de header file met hierin de definities wordt verzameld door een programma genaamd KDoc dat aan de hand van deze informatie een aantal HTML pagina's genereert die samen de productdocumentatie vormen.

7.5 Het programmeren zelf

Na het maken van de schermontwerpen en de eventuele aanpassingen hierop en het bedenken van een documentatie strategie kon het echte programmeren beginnen. Ik heb dit zo gestructureerd mogelijk gedaan dus nooit aan één of twee klassen tegelijk werken, geen tijdelijke hacks gebruiken en een vorm van versiecontrole gebruiken.

Bij het programmeren van de applicatie moet er rekening gehouden worden met de volgende punten:

ID	Algemene eisen	Prioriteit
A1	De applicatie moet gemaakt worden in C++	Basic
A2	De applicatie moet kunnen draaien onder Windows, OS X en Linux	Luxe
A3	De applicatie moet met een aantal verschillende DBMS-en werken	Basic
A5	De applicatie moet zo generiek mogelijk opgezet worden voor toepassingen in andere branches	Basic

Het programmeren is eigelijk het steeds herhalen van de volgende procedure.

1) Programmeer de nieuwe klasse
2) Test de nieuwe klasse
3) Verbind de klasse met de andere klassen die al klaar zijn
4) Test de verbinding

Deze procedure kan oneindig herhaald worden voor elke nieuwe klasse die gemaakt wordt. Het proces van het maken van een klasse is goed te beschrijven aan de hand van een voorbeeld: de klasse QDRecordingOverview. De klasse QDRecordingDialog heeft als functie de benodigde variabelen bij elkaar te zoeken.

In het klassendiagram is al gedefinieerd wat de publieke functies zijn dus die mogen niet veranderd worden (zonder het klassendiagram aan te passen). Deze kunnen zonder meer ingevuld worden. Ik heb hieronder de code van één functie helemaal uitgewerkt

In de code wordt een XML file zoals in hoofdstuk 7.3 beschreven is, ingelezen en verwerkt. Vervolgens worden er, indien ze bestaan, acties aan de nieuwe buttons gekoppeld. Sommige elementen zijn belangrijker dan anderen. Zonder een okay button is het hele scherm waardeloos omdat er nooit data verwerkt kan worden zonder op de okay button te duwen.

recordingdialog.h

```
class QDRecordingDialog : public QObject
{
public:
        /**
          * Constructs a new QDRecordingDialog
        * @param parent Pointer to the parent
        * @param name Name of the object
          */
        QDRecordingDialog(QObject *parent = 0, const char * name = 0);
        /**
          * Creates and show the dialog
          */
        int show();
```

recordingdialog.cpp

```
QDRecordingDialog::QDRecordingDialog(QObject *parent, const char * name) : QObject(parent, name)
{
_dialog = (QDialog * )QWidgetFactory::create("ui/newrecordingdialog.ui", this);
// if there was an error creating the dialog '_dialog' will be 0
if (_dialog)
{
        QLineEdit * _filename  = (QLineEdit *) _dialog->child("filename",    "QLineEdit");
        if( _filename)
                connect (_filename, SIGNAL( textChanged ( const QString & ) ), this, SLOT( fileChanged() ) );
        QPushButton * _okay    = (QPushButton *) _dialog->child("ok",      "QPushButton");
        QPushButton * _cancel = (QPushButton *) _dialog->child("cancel", "QPushButton");
        QPushButton * _help    = (QPushButton *) _dialog->child("help",    "QPushButton");
        if( _help)
                connect(_help, SIGNAL( clicked() ), this,     SLOT( help() ) );
        if( _okay )
        {
                connect(_okay, SIGNAL( clicked() ), this,     SLOT( okay() ) );
                connect(_okay, SIGNAL( clicked() ), _dialog, SLOT( accept() ) );
        }
        else
                // you need an okay button otherwise the dialog is useless
                QDErrorHandler::addError("QDRecordingDialog","No Okay button", QDErrorHandler::Warning);

        if( _cancel)
                connect(_cancel,       SIGNAL( clicked() ), _dialog, SLOT(reject () ) );
        setValues();
}
else
        QDErrorHandler::addError("QDRecordingDialog","Error loading interface file", QDErrorHandler::Warning);
}
```

QT maakt gebruik van een eventsysteem dat bestaat uit signals en slots. Een object verstuurt een signal dat doorgegeven wordt naar één of meerdere objecten die hieraan verbonden zijn.

Dit werkt als volgt:

myapplication.cpp

```
QDEventGenerator * _generator = new QDEventGenerator(qApp, "Event Generator");
QDisplay * _display = new QDDisplay(this, "Graphical User Interface");
……
connect( _display, SIGNAL( startSensor(int) ), _generator, SLOT( startSensor(int) ) );
```

Eerst worden alle belangrijke klassen gedefinieerd. Daarna worden in het hoofdprogramma de signals aan de slots verbonden. De objecten weten dus onderling niets van elkaar, alleen in het hoofdprogramma is er een overzicht wie met wie communiceert. Dit heeft als voordeel dat het ene component makkelijk met het andere component gecombineerd kan worden. Ook zijn componenten veel beter herbruikbaar doordat ze volledig op zichzelf staan.

Andere klassen

Tijdens het maken van de applicatie zijn er een aantal hulpklassen gemaakt die niet relevant zijn voor de inhoud van het programma maar wel voor de werking. Een voorbeeld hiervan is de klasse QDConfigFile. Deze klasse slaat allerlei configuratie-items op in een XML file en maakt deze beschikbaar voor de rest van de applicatie. Hier worden gebruikersnamen, wachtwoorden etc in opgeslagen. De data wordt opgeslagen aan de hand van een variabele naam en een waarde. Elke variabele naam kan maar één keer voorkomen.

Een van de belangrijkste functies in deze klasse is het zoeken. Er zijn drie mogelijkheden om te zoeken. Zoeken naar een string, naar een getal of een boolean waarde.

configfile.h

```
QString find(QString key, QString defaultValue = 0 );
bool    findBool(QString, bool    defaultValue = 0 );
int     findInt(QString,  int     defaultValue = 0 );
```

configfile.cpp

```
QString QDConfigFile::find(QString key, QString defaultValue)
{
        QDConfigList::iterator it;
        for ( it = _configList.begin(); it != _configList.end(); ++it )
        if ( (*it).key() == key)
           return (*it).value().toString();

        return defaultValue;
}
```

In deze functie wordt er eerst door de hele lijst gegaan om te kijken of er een waarde is die bij de variabele naam hoort. Als die gevonden wordt dan wordt die waarde terug geven. Indien de waarde niet is gevonden wordt de defaultValue terug gegeven.

In de klasse wordt geen vorm van sortering gebruikt. Er zullen zelden meer dan honderd configuratie-items aanwezig zijn dus de performancewinst is slechts minimaal. Daarnaast worden de configuratie-items in het algemeen alleen tijdens het opstarten opgevraagd en verder tijdens het draaien van de applicatie niet meer. De extra performance weegt dus niet op tegen het extra werk.

8. Testen

Nadat (een deel van de) implementatie af is kan er begonnen worden met testen. Dit kan op twee verschillende methoden. Blackbox en whitebox testen.

Met 'Blackbox testen' ga je er vanuit dat je niets weet van hoe het stuk code in elkaar zit maar test je gewoon of het werkt. Of de response volgens de specificaties is en of het gedrag verwacht is.
Bij 'Whitebox testen' weet je juist wel wat er in het stuk code gebeurt. Je loopt regel voor regel de code door om te kijken of er geen bugs inzitten. Je controleert dan of variabelen netjes gedigitaliseerd worden, of er nooit gedeeld door 0 wordt of dat er buffer overflows op kunnen treden.

Al deze testen doe je aan de hand van een testplan. Hierin beschrijf je hoe je verschillende onderdelen wilt gaan testen, wat voor testsets je hierbij gaat gebruiken en wat hier de uitkomsten van waren. Hier komt een lijstje met problemen uit. Problemen die bij elkaar horen worden gegroepeerd tot bugs.

8.1 Whitebox testen

Whitebox testen is de lastigste vorm van testen. Je moet naar code kijken die je zelf geschreven hebt en daar je eigen fouten in zien te ontdekken. Het is het makkelijkst om een aantal regels op te stellen waar code aan moet voldoen.

- Variabelen moeten altijd geinitialiseerd worden
- Integers nooit initialiseren met een 0 waarde
- Waarden die van buiten het systeem komen altijd controleren op lengte en geldigheid
- Strings die worden geconverteerd naar integers checken op 0 waarden

Whitebox testen levert in het algemeen problemen op die direct opgelost kunnen worden. Als er een fout ontdekt wordt is het altijd duidelijk waar deze zit

8.2 Blackbox testen

Blackbox testen is makkelijker dan Whitebox testen maar er zijn wel meer dingen waar rekening mee gehouden moet worden.

- Voldoet het gedrag aan de verwachtingen
- Is het gedrag consistent
- Worden er duidelijke foutmeldingen en vragen gesteld

Blackbox testen levert in het algemeen problemen op die niet direct opgelost kunnen worden. Omdat de interne werking niet bekend is moet er aan de hand van de symptomen ongeveer uitgezocht worden waar het probleem zich bevindt.

8.3 Testrapport

Deze twee methodes van testen moeten vervolgens in de praktijk gebracht worden. Hoe dit allemaal is gebeurt is vastgelegd in het testplan. De resultaten hiervan zijn veel belangrijker en staan hier onder vermeld.

Nr	Testcase	Resultaat	Correct
1	Er wordt een nieuw event gegenereerd	Het event wordt opgeslagen in de buffer	✓
2	Verander de instellingen van een sensor	De sensor herlaadt zelf zijn instellingen en blijft functioneren	✓
3	Er wordt een nieuwe lay-out geladen	De juiste beelden worden getoond	✓
4	Maak een nieuwe recording aan	De recording begint en springt automatisch over naar een volgende file bij de maximale grote	✓
5	Lees een recording in en probeer deze af te spelen	Video events worden niet weergegeven	✗
6	Disconnect een sensor en zie hoe het programma hierop reageert	Er wordt een foutmelding gegenereerd in de centrale event logger	✓
7	Genereer een nieuw data event en kijk wat er gebeurt	Indien nodig wordt de bijbehorende grafiek opnieuw getekend	✓
8	Zet een diagram stil en kijk dan of deze op de één of andere manier beweegt	Dit gebeurd in vrijwel alle gevallen behalve het resizen van een window	✗
9	Stop een actieve recording	Het is mogelijk een actieve recording te stoppen, deze weggooien zodat deze niet meer in de lijst staat kan echter niet	✗
10	Exporteer de errors naar CSV of XML	Alles wordt netjes opgeslagen in een XML file	✓

De resultaten uit de whitebox test

Nr	Filename	Gevonden fout
1	Alle files	Er wordt regelmatig gebruik gemaakt van een variabele ‘tmp’ of ‘tmp2’. Dit is onduidelijk en moet veranderen
2	Alle files	Er worden regelmatig hele lange commando’s gebruikt (functie().functie2().functie3().functie4()) Dit is erg onoverzichtelijk en moet zo veel mogelijk vermeden worden.
3	Graphicalrepresentation.cpp	Niet overal wordt op ‘delen door 0’ gecontroleerd
4	Serialconnection.cpp	De ontvangen input wordt niet gecontroleerd op een buffer overflow
5	Cameraconnection.cpp	De manier van implementatie moet eigelijk anders zodat er echt streaming ondersteund wordt
6	Sensoroverview.cpp	Nog geen gebruik gemaakt van de centrale error logger

Bij elkaar zijn er niet vreselijk veel fouten boven water gekomen. Dit komt deels doordat ik van te voren al regelmatig de sourcecode heb doorgelopen op fouten en deze direct aangepast heben anderzijds doordat ik niet objectief ben. Ik heb de code zelf geschreven en het is dus lasting om hier goed commentaar op te geven of fouten uit te halen.

9. Implementatie AMC

Nadat de applicatie klaar en getest was, was het tijd om het te testen in een productieomgeving. Vanwege de medische insteek gebeurde dit bij het Academisch Medisch Centrum in Amsterdam. In dit hoofdstuk bespreek ik de aanleiding, werkzaamheden en uitkomsten van dit stuk van mijn afstudeerperiode

9.1 Situatiebeschrijving

In het AMC is sinds het begin van dit jaar een zogenaamde Traumakamer in gebruik genomen. Dit is een soort van onderzoeksruimte waar patiënten die met een ziekenauto of traumahelikopter binnen gebracht worden direct onderzocht worden. In de ruimte is een CT scanner en een röntgenapparaat aanwezig en zonodig kan er direct geopereerd worden.

Zodra de patiënt binnengebracht is, wordt hij/zij aangesloten op allerlei soorten meetapparatuur. Zeker met zware verkeersslachtoffers is het nodig om hartslag, bloeddruk en de samenstelling van het bloed te controleren.

De apparatuur die hiervoor gebruikt wordt werkt allemaal stand alone dus alle data is verdwenen zodra de patiënt de ruimte verlaat. Omdat het AMC deze data wil gaan gebruiken voor evaluatie besprekingen moet hier verandering in komen. Hiervoor zou het programma gebruikt kunnen worden dat ik gemaakt heb.

9.2 Doel

Er moet onderzoek gedaan worden naar de mogelijkheid om een koppeling te maken tussen de medische apparatuur en het programma dat ik gemaakt heb. Indien mogelijk moet daar ook de basis voor gelegd worden. Als uitkomst moet er een rapport komen waarin voor elk onderzocht apparaat een korte analyse is opgenomen en een overzicht van de mogelijkheden voor het maken van een desbetreffende koppeling.

9.3 Apparatuur

Binnen het AMC waren er verschillende apparaten die op het systeem aangesloten moesten worden. Deze worden in de traumakamer gebruikt maar staan ook opgesteld in bijvoorbeeld de Intensive Care of operatiekamers.

Op het lijstje met apparatuur staan de volgende vijf apparaten:

Meetapparatuur
- Philips H3046A, Monitor (ECG, SP)2, NIBD, Druk, CO2, Temp etc)
- Adaptive Support Ventilation, Raphael Color (firma Hamilton Medical) serienummer CH7403

Infuuspompen
- Perfusor FM, Braun
- Infusomaat P, Braun
- D-100 Level one systeem 1000

Van geen van deze apparaten is iets meer te vinden dan een productbeschrijving van maximaal drie kantjes. Ik heb dus alles met de gebruikershandleidingen uit moeten zoeken die in de doos bij het apparaat zelf zaten. Hierdoor heb ik vrijwel al mijn onderzoek in de drie weken bij het AMC moeten doen en kon ik weinig tot niets van te voren voorbereiden.

9.4 Problemen

Tijdens dit deel van mijn afstuderen ben ik enorm veel problemen tegen gekomen. Ik kwam in een compleet andere omgeving dan dat ik gewend was wat resulteerde in allerlei misverstanden. Binnen de medische wereld heeft alles een net andere naam. Een monitor is bijvoorbeeld geen computer beeldscherm maar een apparaat waar je een patiënt op aansluit en waar dan bijvoorbeeld bloeddruk en hartslag vanaf te lezen is. Ook is de manier van werken heel anders. Normaliter gebruik je zoveel mogelijk één standaard, maar in een ziekenhuis heeft elk apparaat en leverancier zijn eigen manier van communicatie. Zelfs apparatuur van één leverancier werkt niet gegarandeerd samen.

Het was voor mij dus ook erg moeilijk om überhaupt te kunnen communiceren met de apparatuur. Om hun marktaandeel niet prijs te hoeven geven zijn veel protocolbeschrijvingen die gebruikt worden niet publiekelijk beschikbaar. Er kan via een vertegenwoordiger wel naar gevraagd worden blijkt uit sommige handleidingen maar bij geen van de leveranciers heb ik tot nu toe iets los gekregen. De handleidingen die bij de producten zelf zaten waren erg uitgebreid over het gebruik en instellen van de apparatuur maar vermelden vrijwel niets over mogelijkheden om te communiceren met een externe bron.

Het laatste en misschien wel meest vervelende probleem tijdens mijn tijd in het AMC is de bureaucratie die er heerst. Als er iets moest gebeuren dan moest dat altijd met een manager overlegd worden. Dit was extra

vervelend omdat deze mensen er vaak niet waren. Omdat het een ziekenhuis is werken mensen allemaal andere uren en moest ik soms twee dagen wachten op een antwoord. Dit is erg vervelend als je ergens maar drie weken bent.

Ook werd mijn aanwezigheid niet door iedereen gewaardeerd. Ik was daar geplaatst op verzoek van de algemene Manager Traumazorg en niet door een technische manager. De technische afdeling, waarvan ik mijn antwoorden moest krijgen, stelde dit niet echt op prijs waardoor de samenwerking niet altijd even vloeiend liep.

9.5 Oplossingen

Niet voor elk probleem heb ik een oplossing kunnen vinden, daarom is dit deel van mijn afstuderen ook niet helemaal gelukt zoals ik en de mensen bij Sony dit graag gezien hadden. De onduidelijkheid rondom de apparatuur zorgde dat ik niet echt kon testen of experimenteren of een bepaalde methode van communiceren werkte.

Voor andere problemen heb ik wel een oplossing kunnen vinden. Ik ben een aantal keren met de Manager Traumazorg bij een traumageval geweest waar ze me alle apparatuur aanwees en wat er mee gebeurde. Hierdoor snapte ik de werking van bepaalde apparatuur een stuk beter en weet ik nu ook wanneer wat gebruikt wordt. Hierdoor krijg je een stukje inzicht wat helpt bij het onderzoeken.

Het probleem met de bureaucratie heb ik zoveel mogelijk genegeerd. Ik was daar toch vooral te gast en vond het niet nuttig om daar een groot probleem van te maken. Ik heb het wel met een begeleider van het AMC besproken die toegaf dat het zo was maar dat er niets aan te doen was. Hier ben ik dus maar vanuit gegaan en heb ook de technische afdeling zo veel mogelijk ontzien door vragen direct aan de fabrikanten/leveranciers te stellen in plaats van aan hen.

10. Evaluatie

In dit hoofdstuk ga ik mijn eigen afstudeerperiode evalueren. Ik maak daarbij onderscheid tussen de proces en productevaluatie. Verder geef ik aan wat ik er precies van geleerd heb en hoe ik hoe ik daar nu tegen aan kijk.

10.1 Product evaluatie

Het ontwerp
Nadat je het ontwerp af hebt en bent begonnen met programmeren bedenk je vaak dat je dingen op een andere manier had moeten modeleren omdat dit net handiger uitgekomen was. Ook in dit geval waren er wat kleine dingen die ik graag anders had gedaan maar in het algemeen ben ik erg tevreden over het ontwerp. Er zit genoeg abstractie in om alle eisen die aan het programma gesteld zijn in te willigen en toch ruimte te houden voor uitbereiding.

Het programma
Ik ben zelf erg tevreden over het programma. Het voldoet in grote lijnen aan de verwachtingen die ik er zelf van had en ook de collega's van Sony zijn er over te spreken. Ik heb er zelf een heel stuk creativiteit in kunnen verwerken en vond het heel leuk om er gedurende een tijd elke dag aan te werken. Ik denk zeker dat het programma ook in een echte omgeving ingezet gaat worden wat een behoorlijk compliment is richting mij.

Helaas is niet alles helemaal afgekomen. De integratie in een productieomgeving is nog niet wat het zijn moet. Hier ga ik de tijd tussen het inleveren van dit verslag en de eindzitting aan werken. Ik hoop dan hier nog iets over te vertellen.

De documentatie
De documentatie is altijd een lastig punt in een softwareontwikkelproces maar ik ben blij dat ik een manier heb gevonden om deze documentatie te laten genereren aan de hand van commentaar in de header files. Dit maakte het maken een stuk makkelijker waardoor ik er ook meer aandacht aan besteed heb.

Slot
Ik ben tevreden over de producten die ik opgeleverd heb. Ik heb hier een stukje creativiteit in kunnen leggen en heb er met plezier aan kunnen werken.

10.2 Proces evaluatie

Methode
Ik heb SDM gekozen omdat het de enige methode is die we uitgebreid behandeld hebben. Misschien had ik toch de stap moeten zetten en eens naar IAD kijken. Veel dingen die ik gedaan heb, passen binnen een IAD traject. Vooral het feit dat je bij voortschrijdend inzicht of andere aanpassingen vrijwel al je documenten opnieuw moet maken is erg vervelend. SDM geeft wel duidelijk aan wat er moet gebeuren en is dus redelijk makkelijk in het gebruik.

Technieken
Ik ben zelf erg tevreden met de keus voor UML. Ondanks de gebreken vind ik het een goede manier om de toekomstige situatie te schetsen. Daarnaast is het met bijvoorbeeld een Sequence Diagram mogelijk een leek op een simpele manier de samenwerking tussen de verschillende componenten te laten zien.

Planning
Aan het begin van het afstudeertraject heb ik netjes een planning gemaakt en daar heb ik me zoveel mogelijk aan gehouden. Het bleek dat ik het programmeren zelf niet helemaal goed had ingeschat. Hier ben ik twee weken langer mee bezig geweest dan dat aan het begin was ingepland. Dit kon ik echter opvangen met de twee weken uitloop die ik wel ingepland had. Ook heb ik wat voorbereidende werkzaamheden uit kunnen voeren in de tijd die ingepland stond voor het maken van de definitiestudie en het basisontwerp.

Slot
Ik ben zelf tevreden over het proces en ik denk dat ik zeker bezig geweest ben op het niveau van een vierdejaars HBO student. Ik heb het nodige geleerd en bewezen dat ik de technieken en methoden die ik de afgelopen vier jaar geleerd heb kan gebruiken en toepassen binnen een reële situatie. Verder heb ik de nodige praktijkervaring opgedaan wat een goede aanvulling is op mijn C.V.

10.3 Wat heb ik geleerd

Ook al is het doel van het afstuderen het bewijzen dat ik de dingen die ik de afgelopen vier jaar geleerd hebt in de praktijk kan brengen vind ik het voor mezelf toch belangrijk om eens op een rijtje te zetten welke andere dingen ik de afgelopen periode geleerd heb.

Het belangrijkste wat ik de afgelopen paar maanden geleerd heb is het werken in een professionele omgeving en wat daar bij komt kijken. Sony heeft ongeveer 180.000 werknemers dus alles werkt anders dan in een klein bedrijf. Werknemers zijn vooral gericht op hun eigen productgroep en zien niet altijd het grote geheel. Dit is een vreemde gewaarwording maar na een gesprek met mijn bedrijfsmentor kwam ik er achter dat dit in vrijwel elk groot bedrijf zo is.

Wat ik zelf ook leuk vond waren de collega's bij Sony. Omdat het een organisatie is die door heel Europa en het Midden Oosten werkt komen de werknemers ook overal vandaan. Hierdoor doe je een hoop extra ervaring mee op, al is het alleen maar het continue Engels praten.

11. Bronvermelding

Tijdens mijn afstudeerperiode heb ik verschillende bronnen gebruikt. Hieronder volgt een lijstje van boeken, websites en handleidingen die ik gebruikt hebt. Deze lijst is niet helemaal compleet. Om het overzicht te behouden heb ik alleen de belangrijkste sites genoemd.

Boeken

- Praktisch UML — Jos warmer & Anneke kleppe
- C++ GUI programming with QT3 — Bruce Pseren
- Programming with QT — Matthias Dalheimer
- XML voor het dagelijks gebruik — Michael Seeboerger
- Database systemen voor de praktijk — VanDenBulcke

Handleidingen

- Philips monitor handleiding (medisch)
- Hamilton beademingsapparatuur handleiding (medisch)
- Styleguide programmeereisen Sony

Sites

- QT manual — http://doc.trolltech.com/3.2/
- Google — http://www.google.nl
- Hamilton medical — http://www.hamilton-medical.com/
- Philips medical — http://www.medical.philips.com/

12. Termenlijst

In dit verslag zijn op verschillende plaatsen termen gebruikt die hier nader toegelicht worden. Indien nodig is er een website bij vermeld waar verdere informatie gevonden kan worden.

Term	Omschrijving
Codebase	Alle code waaruit de applicatie bestaat, inclusief documentatie en iconen.
ERD	Enitity Relation Model. Een manier om de relatie tussen verschillende data elementen aan te geven. Hierin kunnen verplichtingen in worden vastgelegd en of er meerdere relaties tegelijk kunnen bestaan.
Mockup	Een mockup is een getekende userinterface. Dit kan gebeuren met een tekenprogramma, met een speciaal grafisch ontwikkelprogramma of gewoon met de hand. Aan de hand van een mockup kan er gebrainstormd worden en kunnen er makkelijk een verandering doorgevoerd worden.
Object georiënteerd programmeren	Bij object georiënteerd programmeren ligt de nadruk op de data. Een object geeft een hoeveelheid data aan, gegroepeerd met alle functies die werken op de data. Programmeertalen die op deze manier (kunnen) werken zijn Java en C++.
QT	QT is een grafische toolkit die functioneert bovenop C++. Naast widgets zoals buttons, invoervelden en popup menu's ondersteunt het ook een standaard manier van netwerktoegang. QT is beschikbaar voor Windows, OS X en Linux. Applicaties kunnen eenvoudig uitgebracht worden voor meerdere platformen door ze simpelweg opnieuw te compileren. Er is dus maar één codebase en geen drie. Voor verdere informatie zie http://www.trolltech.com/products/qt/
UML	Unified modelling language. Ondanks de naam is UML geen taal maar een techniek. Met behulp van deze techniek is het mogelijk de toekomstige situatie van een systeem te tekenen en definiëren. UML bevat modellen om de toestand van een object, de eigenschappen van een object en de communicatie van objecten onderling weer te geven. Voor verdere informatie zie http://www.uml.org/

Widget	Kleinste onderdeel waaruit een grafische omgeving wordt opgebouwd. Dit kan een button zijn, een checkbox, een dropdown menu etc. Dit is een term die binnen QT gebruikt wordt. In de MFC documentatie wordt dit een window genoemd
XML	XML staat voor Extensive Markup language. Met behulp van XML kan er op een gestructureerde manier data worden opgeslagen in een file. Deze data kan vervolgens door elk ander programma dat XML data aan kan ingelezen en verwerkt worden. Voor verdere informatie zie http://www.xml.com/

Bijlagen

Tijdens het afstudeertraject zijn er allerlei producten gemaakt die relevant zijn voor het proces. Deze zijn allemaal achteraan toegevoegd. Aangezien dit losse producten zijn en niets met dit verslag te maken hebben heeft elk product zijn eigen voorkant en is de originele bladzijdennummering in stand gehouden.

Als bijlage zijn opgenomen in deze volgorde:

- Opdrachtomschrijving
- Plan van aanpak
- Definitiestudie
- Basisontwerp
- Testplan
- Gebruikershandleiding
- Technische handleiding
- Overzicht medische apparatuur

Opdrachtomschrijving

Sony traject

Auteur
M.C. Van Dorp
Datum
24 Maart
Versie
1.2

1. Inleiding

Sony Medical, als onderdeel van Sony Europe, levert en plaatst systemen voor de gezondheidszorg. Het betreft vooral systemen voor het presenteren van patiëntendata en sinds kort ook videobewaking. De afdeling levert apparatuur en software aan onder andere ziekenhuizen en onderzoekcentra.

2. Probleemstelling

Bij elke implementatie krijgt men verschillende soorten input te verwerken, voorbeelden hiervan zijn seriële input, gebruikers input of informatie vanuit digitale of analoge contacten. Het probleem dat zich elke keer weer voordoet is dat er geen goede applicatie is om al deze verschillende soorten input te combineren en te representeren.

3. Doelstelling van de opdracht

Op het moment is er een applicatie, iFast genaamd, die door een relatie van Sony gemaakt wordt en mogelijk geschikt is. Sony heeft van de relatie de source code van het programma gekregen en kan dit nu aan gaan passen. Binnen Sony zijn hiervoor verschillende ideeën waarvan de haalbaarheid nog bekeken moet worden.

Het doel van de afstudeeropdracht is het bestaande programma te herzien, te documenteren en uit te bereiden.
Er moet functionaliteit toegevoegd worden die duidelijk wordt door interviews met Sony medewerkers en klanten van Sony.
Daarnaast moet er een gebruiksvriendelijke interface gemaakt worden die gebruikt kan worden zonder dat hier een keyboard voor nodig is.

Als er een versie klaar voor gebruik is kan deze waarschijnlijk getest worden binnen het AMC ziekenhuis in Amsterdam. Het ziekenhuis wil graag de trauma kamers beter gaan monitoren en hoopt dat met deze applicatie te kunnen doen.

4. Aanvulling op de opdracht

Binnen de ziekenhuizen zijn een aantal protocollen actief waar rekening mee gehouden moet worden. De twee belangrijkste zijn HL7 en de DICOM standaard. HL7 wordt gebruikt om patiënt gegevens zoals naam, leeftijd etc uit te wisselen tussen systemen. DICOM is een relatief nieuwe standaard die wordt gebruikt om medische gegevens uit te wisselen. Dit zijn dus testresultaten, X-Ray scans of MRI scans.
Vooral DICOM is iets wat door Sony zelf gepromoot wordt.

Om de applicatie binnen een bestaande omgeving te kunnen gebruiken moet er op termijn een koppeling gemaakt worden. Hiervoor moeten basis functionaliteiten voor beschikbaar zijn. Hier moet onderzoek naar gedaan worden
N.B. Het maken van de koppelingen zelf valt buiten de scope van de opdracht.

5. Planning

Dit is een globale planning voor het hele traject. De planning beslaat 18 weken waarin zowel het product als het afstudeerverslag af dienen te zijn.

Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Inrichten werkplek																		
Inwerken en documenteren iFast																		
Maken Definitiestudie																		
Maken Basisontwerp																		
Maken mockups voor overleg																		
Programmeren systeem, inclusief testen																		
Implementatie bij bijv. AMC																		
Uitloop																		
Onderzoek doen naar DICOM																		
Werken aan afstudeerverslag																		

Plan van aanpak

Sony traject

Auteur
M.C. Van Dorp
Datum
1 Maart
Versie
1.0

Voorwoord

Dit document is gemaakt in het kader van mijn afstudeerperiode en is bedoeld voor de begeleidende docenten, de afstudeerbegeleider en als opstart van het afstudeertraject. Dit document valt niet onder de mijlpaalproducten voor de opdrachtgever en daarom zal dit document ook alleen in het Nederlands beschikbaar zijn.

Inhoudsopgave

1 Opdrachtomschrijving

Sony Healthcare, als onderdeel van Sony Europe, levert en plaatst systemen voor de gezondheidszorg. Het betreft vooral systemen voor het presenteren van patiëntendata in combinatie met beelden en sinds kort ook videobewaking. De afdeling levert apparatuur en software aan onder andere ziekenhuizen en onderzoekcentra.

Bij elke implementatie krijgt men verschillende soorten input te verwerken, voorbeelden hiervan zijn seriële input, gebruikers input of informatie vanuit digitale of analoge contacten. Het probleem dat zich elke keer weer voordoet is dat er geen goede applicatie is om al deze verschillende soorten input te combineren en te representeren. Er zijn wel besta
andere applicaties maar deze zijn behoorlijk prijzig en bieden voor hun prijs te weinig functionaliteit. Deze kostprijs zorgt dat bepaalde trajecten niet doorgaan.

Op het moment is er een applicatie, iFast genaamd, die door een relatie van Sony gemaakt wordt en mogelijk geschikt is. Hier wordt alleen niet aan verder gewerkt. Sony heeft van de relatie de sourcecode van het programma gekregen en kan dit nu aan gaan passen.
Het doel van de opdracht is het bestaande programma te herzien, uit te bereiden en te documenteren. Hiertoe zal eerst duidelijk gemaakt worden wat de functionaliteit van het programma moet zijn. Het geheel moet dan beschikbaar worden als een soort plugin voor Realshot Manager, een programma wat al door Sony zelf ontwikkeld wordt. Dit programma verzorgt de weergave van de camera's.
Zodra het programma klaar is om in productie gebruikt te gaan worden zal er een implementatie traject gezocht worden waarbij het programma ingezet kan worden.

2 Afbakening opdracht

Het programma, of de plug-in, zal een beperkt aantal sensoren en output manieren ondersteunen. Dit is omdat elke nieuwe soort sensor onderzoek- en implementatietijd kost en deze er niet direct is. Later kunnen er altijd nog mogelijkheden toegevoegd worden maar dat valt buiten de afstudeeropdracht.

3 Randvoorwaarden

Aan software ontwikkeling, zeker binnen een groot bedrijf als Sony, zijn allerlei regels en voorwaarden verbonden. Er zijn regels over hoe code opgeleverd moet worden, welke programmeertaal er gebruikt dient te worden, hoe er gerapporteerd en gedocumenteerd dient te worden, speciale regels over deadlines etc. Aan al deze eisen moet voldaan worden. Hier is binnen Sony een document voor, het is dus niet nodig al deze eisen hier voluit te noemen.

4 Risicofactoren

Binnen elk project bestaan verschillende risico's die de voortgang kunnen belemmeren. Het is verstandig om vooraf een aantal van deze risico's te bekijken en te bepalen hoe daarmee om te gaan. Risico's moeten wel realistisch zijn dus dingen als aardbevingen en overstromingen worden niet opgenomen

Risico
Doordat er te weinig medewerking is vanuit de software ontwikkeling afdeling loopt het traject vertraging op waardoor deadlines niet gehaald worden.
Oplossing
Er moet een soort van contract gemaakt worden tussen het projectteam en de software ontwikkeling afdeling waarin vastgelegd wordt hoe communicatie dient te lopen, wat voor termijnen wel en niet acceptabel zijn etc.

Risico
Het project is onverzien veel complexer dan dat dit aan het begin ingeschat was, hierdoor is er teveel werk en komt de afstudeeropdracht niet op tijd af.
Oplossing
Als er eenmaal begonnen is aan het project kan er aan de hand van voortschrijdend inzicht bepaalt worden of het project goed is ingeschat. Als dat niet zo is dan kan tot de zesde week van de periode de opdrachtomschrijving aangepast worden zodat deze wel reëel haalbaar is.

5 Projectorganisatie

Er is niet echt sprake van een projectgroep maar de personen die (in)direct aan het traject meewerken zijn:

Direct
- Matthijs van Dorp (Afstudeerder, verantwoordelijk voor documenteren, modelleren, programmeren en interviewen)
- Martin Diependaal (afstudeerbegeleider, technisch specialist, verantwoordelijk voor communicatie met andere afdelingen)
- Gerard Bes (praktijk specialist, verantwoordelijk voor contact met klanten)

Indirect
- Henk Demper (contact software ontwikkeling)
- Naam onbekend (IT Manager ziekenhuis AMC)

In principe zullen de werkzaamheden verricht worden door Matthijs van Dorp, de overige personen op de lijst zijn beschikbaar voor interviews, technisch advies of gaan mee op bezoek bij een klant van Sony.

6 Wijze van rapporteren

Tijdens de periode dat er aan het product gewerkt wordt zal er regelmatig gerapporteerd moeten worden aan verschillende personen.

Afstudeerbegeleider:
Er wordt wekelijks een bespreking gehouden met de afstudeerbegeleider waarin de stand van zaken behandeld wordt. Er zal aandacht gegeven worden aan de resultaten van die week. Ook zal er dan de mogelijkheid zijn (mogelijke) problemen te bespreken.

Afdeling software ontwikkeling:
Omdat deze afdeling in de toekomst het programma moeten gaan onderhouden zal deze afdeling regelmatig op de hoogte gehouden moeten worden van de status van het product. De eerste keer dat dit gebeurd is vlak voor er begonnen gaat worden met programmeren, daarna zal er elke twee weken een rapport gestuurd worden met hierin de status en eventuele toelichtingen.

7 Benodigde mensen/middelen

Om het project succesvol af te kunnen ronden zijn er mensen en middelen nodig. Hieronder volgt een overzicht van wat er minimaal nodig is. Hier kunnen later nog extra dingen aan toegevoegd worden.

Rapporten:
- Beschrijving plugin management Realshot Manager
- Beschrijving programmeervoorwaarden

Software:
- Complete IDE voor ontwikkelingstoepassingen
- Realshot Manager

Hardware:
- Computer
- Terminal Server (InReach)
- Diverse sensoren voor testdoeleinden

8 Planning

Om het project te kunnen beheersen en iets van deadlines te kunnen realiseren is er een planning gemaakt. De planning is gericht op het afstudeertraject. Hierdoor zijn zaken als werken aan het afstudeerverslag opgenomen in de planning. De totale periode die beschikbaar is, is 18 weken.

Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Inrichten werkplek																		
Inwerken en documenteren iFast																		
Maken Definitiestudie																		
Maken Basisontwerp																		
Maken mockups voor overleg																		
Programmeren systeem, inclusief testen																		
Implementatie bij bijv. AMC																		
Uitloop																		
Onderzoek doen naar DICOM																		
Werken aan afstudeerverslag																		

9 Beschrijving mijlpaalproducten

Binnen het project zijn de volgende mijlpalen te onderscheiden.
Deze dienen in deze volgorde opgeleverd te worden.

- Plan van aanpak *
- Definitiestudie *
- Detail ontwerp *
- Gebruikers handleiding
- Technische handleiding
- Een werkend systeem

Mijlpalen die met een * gemarkeerd zijn, zijn mijlpalen die alleen voor het proces van belang zijn. Deze zullen dan ook alleen in het Nederlands beschikbaar zijn. Overige documenten worden in het engels opgeleverd.

10 Kosten/batenanalyse

Om het product af te krijgen moeten er kosten gemaakt worden. Deze kosten bestaan uit de volgende onderdelen

- Loonkosten werknemers
- Hardware kosten voor testen
- Promotiekosten

Op het moment is er geen exact bedrag te noemen wat dit totaal gaat kosten, dit is dan ook buiten de scope van dit document.

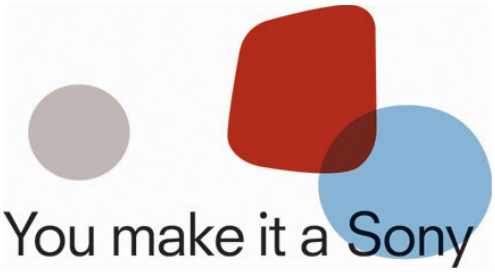
Als het product af is en in productie gaat zal Sony hier inkomsten door verkrijgen

- Licentiekosten
- Hogere cameraverkopen
- Betere “reputatie”

Ook dit is niet direct in aantallen of bedragen uit te drukken. Het is de bedoeling dat de baten de kosten overstijgen, anders is het project niet levensvatbaar.

Definitiestudie

Sony traject



Auteur
M.C. Van Dorp
Datum
25 Mei 2004
Versie
1.2

Voorwoord

Dit document is gemaakt in het kader van mijn afstudeerperiode en is bedoeld voor de begeleidende docenten, de afstudeerbegeleider en als opstart van het afstudeertraject. Dit document valt niet onder de mijlpaalproducten voor de opdrachtgever en daarom zal dit document ook alleen in het Nederlands beschikbaar zijn. Er wordt uitgegaan dat modelleertechnieken die binnen UML gebruikt worden bekend zijn.

N.B. Binnen deze definitiestudie zal er vooral ingegaan worden op de medische sector. Resultaten en het eindproduct kunnen echter ook gebruikt worden in elke andere sector.

Inhoudsopgave

1. Inleiding

Data, zeker in ziekenhuizen staat nooit op zich zelf. Data of gegevens die gemeten worden hebben altijd verband met andere metingen, een persoon of een bepaalde situatie. In ingewikkelde omgevingen kunnen deze verbanden op den duur zo complex worden dat de relatie niet meer duidelijk is. Om toch het overzicht te behouden en op een simpele, eenduidige manier te representeren en te verwerken wordt er steeds meer geautomatiseerd. Chirurgen krijgen de gegevens van patiënten allang niet meer op papier aangeleverd maar op beeldschermen die overal in een operatiekamer aanwezig zijn.
Sony Healthcare, als onderdeel van Sony Europe denkt dat hier een grote markt ligt en wil hier actief in gaan investeren in onder meer softwareontwikkeling en het uitwerken van concepten.

Het systeem waarvoor deze definitiestudie geschreven wordt is onderdeel van het plan en zal een ondersteunende rol krijgen. In deze *definitiestudie* zal er gekeken gaan worden naar de huidige situatie van dit soort automatisering binnen ziekenhuizen, bestaande pakketten en zal er een aantal functionele eisen opgesteld worden. Daarnaast zal er een overzicht van mogelijke veranderingen gemaakt worden.

2. Beschrijving huidige situatie

Op het moment is er bij Sony nog geen applicatie/systeem wat de functionaliteit bevat die geëist wordt van dit systeem. Er is dan ook niet echt een huidige situatie. Om toch een goed beeld te krijgen van hoe het er nu aan toe gaat volgt er een situatieschets van hoe nu bepaalde scenario's worden opgelost. In de het hoofdstuk "Verandering" worden deze scenario's nog een keer besproken maar dan op de nieuwe manier.
Ook hier is er gekozen om scenario's uit de gezondheidszorg te nemen omdat het hier als eerste toegepast zal gaan worden.

2.1 Behandeling na operatie

Een patiënt komt na een operatie binnen op een vercoeverkamer. Daar blijft hij/zij liggen totdat hij bij gekomen is en zijn hartslag/broeddruk etc in orde is. Iemand van de verpleging loopt regelmatig langs en controleert dit bij elke patiënt.

2.2 Operen en doceren gecombineerd

op het moment wordt dit niet gebruikt.

2.3 Monitoren eerste hulp

Een patiënt wordt binnen gebracht en wordt direct aangesloten op allerlei soorten apparatuur. Deze informatie wordt verder niet opgeslagen, af en toe wordt er wel een aantekening op een formulier gemaakt op basis van gegevens uit deze apparatuur.

3. Verandering

Als het nieuwe systeem ingevoerd is zullen er verschillende dingen anders gaan lopen. De scenario's zoals deze in hoofdstuk twee geschetst zijn worden nu nog een keer bekeken, maar nu op de nieuwe manier.

3.1 Behandeling na operatie

Een patiënt komt na een operatie binnen op een verkoeverkamer. Daar wordt hij/zij aangesloten aan verschillende apparaten. Hierdoor is de bloeddruk en hartslag continue te meten. Deze waarden, samen met videobeelden van de zaal, worden op een wand in de zusterpost geprojecteerd. Als iemand bij komt wordt er handmatig nog een laatste controle gedaan en wordt de patiënt verplaatst.

3.2 Opereren en doceren tegelijkertijd

In een speciaal ontwikkelde operatiekamer die voorzien is van allerlei camera's wordt een operatie uitgevoerd. De camerabeelden gaan naar een klaslokaal dat één verdieping hoger ligt. De videobeelden worden op een soort video muur gepresenteerd. Naast videobeelden komt er ook weer allerlei data op te staan zoals hartslag, bloeddruk en andere relevante meetgegevens. Via een microfoon kunnen ze direct vragen stellen aan de opererende chirurg. Naast het direct tonen worden de beelden ook opgeslagen en kunnen deze gebruikt worden voor andere doeleinden.

3.3 Monitoren eerste hulp

Patiënten worden uit de ambulance op een brancard gelegd. Daar worden ze direct op het systeem aangesloten zodat er altijd zichtbaar is hoe een patiënt er voor staat. Als de meetwaarden boven voor gedefinieerde waarden komen gaat er een soort van alarm en moet er actie ondernomen worden. De gegevens worden opgeslagen zodat het proces later geëvalueerd kan worden.

4. Ontwikkelomgeving

Om aan het project te kunnen werken heft elk projectlid een eigen pc nodig. Op deze pc moet office geïnstalleerd zijn voor het maken van presentaties, documenten en verslagen. Afhankelijk van de taken komt hier een aantal software pakketen bij.
Voorbeelden hiervan zijn

- QT Professional
- KDevelop
- GDB
- Umbrello

Naast deze software eisen moet er, al dan niet op locatie, een testomgeving gecreëerd worden waarin het programma getest, gedebugt en verder ontwikkeld kan worden.
Deze omgeving moet zo veel mogelijk gelijk zijn aan de realiteit dus met verschillende soorten sensoren, veranderende eisen etc.

5. Acceptatieprocedure en wijze van invoeren

Het systeem zal op termijn in productie genomen moeten gaan worden. Dit zal niet zonder slag of stoot gaan aangezien er allerlei kwaliteitseisen zullen zijn waaraan het programma moet voldoen.

De acceptatieprocedure zal bestaan uit een aantal fasen waarin bepaalde werkzaamheden uitgevoerd worden.

Tijdslijn	Programmeren	Testen	Debuggen
Week 1			
Week 2			
Week 3			
Week 4			
Week 5			
Week 6			
Week 7			
Week 8			
Week 9			

In deze week wordt hieraan gewerkt
Release

Na vier weken wordt er een eerste interne release gemaakt. Dit geeft de medewerkers tijd om er naar te kijken en er hun mening over te geven. Ook is dit de eerste stap voor interne publiciteit. Aan de hand van resultaten en commentaar wordt er verder gewerkt aan een tweede interne release. Als hier geen problemen mee zijn dan kan het programma ook gepromoot gaan worden door middel van een externe release.

6. Kosten / baten analyse

Net als elk ander project kost dit pakket iets om het in te voeren. Naast deze kosten zijn er ook baten. Deze moeten goed in verhouding met elkaar zijn anders heeft het invoeren weinig zin en is het bedrijf zonder dit pakket zelfs beter af.

6.1 Kosten

Om het product af te krijgen moeten er kosten gemaakt worden. Deze kosten bestaan uit de volgende onderdelen

- Loonkosten werknemers
- Hardware kosten voor testen
- Promotiekosten

Op het moment is er geen exact bedrag te noemen wat dit totaal gaat kosten, dit is dan ook buiten de scope van dit document.

6.2 Baten

Als het product af is en in productie gaat zal Sony hier inkomsten door verkrijgen

- Licentiekosten
- Hogere cameraverkopen
- Betere “reputatie”

Ook dit is niet direct in aantallen of bedragen uit te drukken.

7. Systeembeschrijving

Voordat er begonnen wordt aan het modelleren moet er een ondubbelzinnige omschrijving van het systeem komen. Deze volledige omschrijving bevind zich in bijlage 1.
Een fragment hieruit is als volgt:

> *Het systeem zoals dat gemaakt gaat worden sluit aan bij een ander systeem dat al in staat is beeld weer te geven. Door de toevoeging van het nieuwe systeem dat als onderdeel van het oudere systeem gaat werken wordt het mogelijk om zowel beeld als bijbehorende data weer te geven en op te slaan voor verder onderzoek. De data die weergegeven wordt is door het systeem eerder uit verschillende databronnen gehaald en mogelijk bewerkt. De data kan op verschillende manieren weergegeven worden namelijk: puur een waarde, een grafiek of veranderend beeld. De waarden die uit het systeem komen zijn vooral bedoelt om naderhand terug te kijken en mogen alleen voor monitoring gebruikt worden. Aan de waarden mogen dus geen consequenties verbonden worden.*

Aan de hand van deze systeembeschrijving kan er een toekomstige situatie gedefinieerd worden.

8. Toekomstige Situatie

Om de toekomstige situatie duidelijk te krijgen gaan we deze modelleren. We maken hier gebruik van UML(Unified Modelling Language). Aan de hand van deze modellen is het een stuk inzichtelijker hoe het programma gaat werken.

In dit hoofdstuk gaan we het systeem, zoals dit in hoofdstuk zeven geschetst is modelleren. We doen dat in vier stappen:

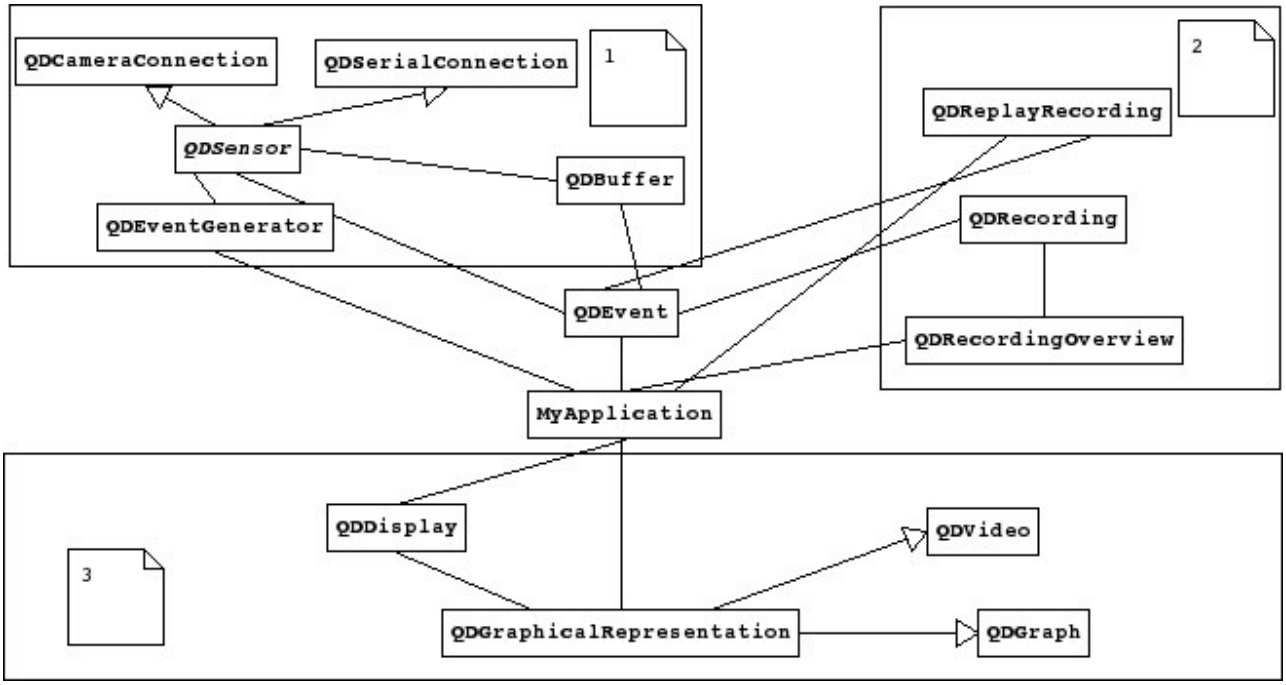
- Klassendiagram
- Use Cases diagram
- Sequence diagram
- State diagram

Elk van deze stappen wordt afzonderlijk toegelicht en besproken.

Het toestandsdiagram valt niet binnen de definitiestudie maar het basisontwerp. Deze is hier dan ook nog niet opgenomen.
De diagrammen zijn zo globaal mogelijk gehouden. Elke nieuwe soort sensor of grafiek die toegevoegd wordt heeft zijn eigen klasse. Het is ondoenlijk om elke keer alle diagrammen aan te passen. Daarom is er een “meta type” diagram en sensor gemodelleerd. Elke nieuwe sensor zal dus alle eigenschappen van de groep “sensors” krijgen, waarbij de technische implementatie verder niet zichtbaar hoef te zijn voor de rest van het systeem.

8.1 Algemeen klassendiagram

Een klassendiagram wordt gebruikt om de verschillende informatie die aanwezig/nodig is inzichtelijk te maken. Naast de informatie zelf wordt erin aangegeven wat er met de informatie, of een object zoals dat heet, gedaan kan worden.



Algemeen klassediagram

Het diagram is opgedeeld in vier verschillende stukken. Drie hiervan zijn losse onderdelen die min of meer onafhankelijk van elkaar functioneren. Het laatste deel bestaat uit een aantal klassen die alles bij elkaar brengen of een algemene functie hebben.

8.1.1 Deel 1

Het stuk aangeduid met het nummer één zorgt dat de informatie bij elkaar gebracht wordt. Via verschillende sensor implementaties (overerving van QDSensor) worden er QDEvents gemaakt en deze worden in een buffer geplaatst.

8.1.2 Deel 2

Het stuk aangeduid met nummer twee zorgt dat events die eenmaal aangemaakt zijn opgeslagen kunnen worden in een bepaald formaat. Via QDReplayRecording is het mogelijk deze recordings weer terug te spelen op dezelfde manier als dat nu gebeurd in deel r als dat nu gebeurd in deel één.

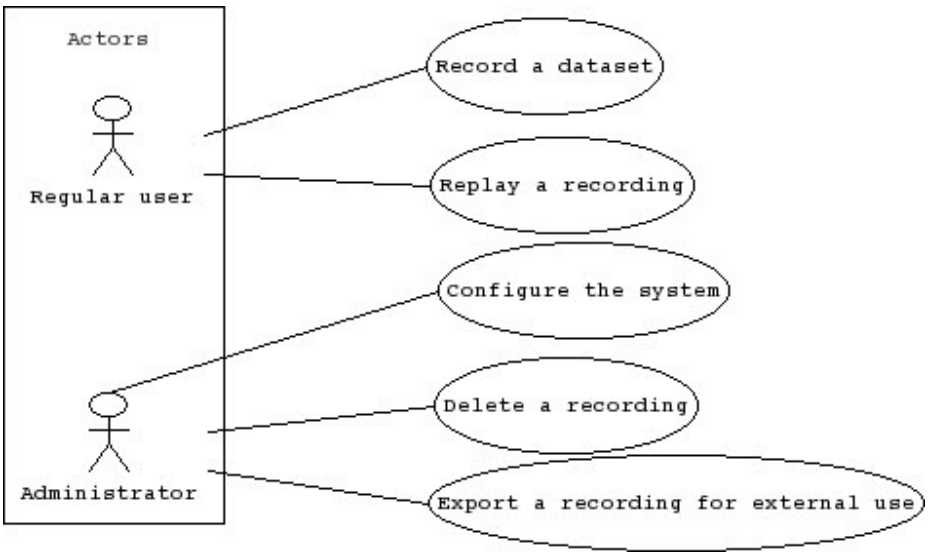
8.1.3 Deel 3
Het stuk aangeduid met nummer drie zorgt dat events die binnen komen grafisch weergegeven kunnen worden. Dit kan in de vorm van een video feed of een grafiek.

8.1.4 Hoofdprogramma
Het laatste stuk vormt de verbinding tussen de drie andere delen. Vanuit MyApplication wordt de communicatie tussen de onderdelen georganiseerd en gecontroleerd. De klasse QDEvent vormt een centraal onderdeel in de applicatie en valt dus binnen geen van de andere drie categorieën.

Dit is slechts het algemene klassendiagram waarin de onderlinge relaties tussen de verschillende objecten naar voren komt. Normailiter worden in een klassendiagram ook alle functies en variabelen die elke klas heeft getoond. Dit zijn er echter zo veel dat dit het diagram onoverzichtelijk maakt. Om toch een totaaloverzicht van alle klassen te hebben zijn in de bijlagen detail klassendiagrammen opgenomen van de losse onderdelen.

8.2 Algemene Use Cases

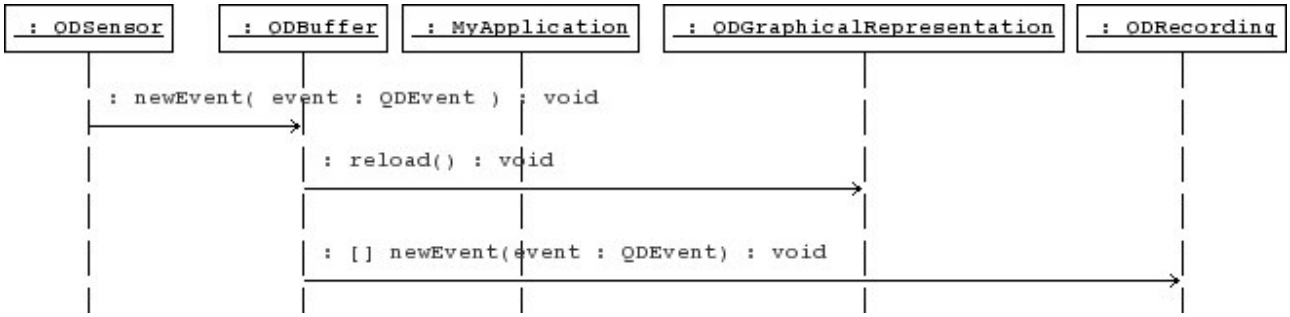
Een Use Case diagram wordt gebruikt om interactie met de gebruiker vast te leggen. Voor iedereen situatie zoals de gebruiker die tegen kan komen wordt een use case gedefinieerd. Dit is wel zo globaal mogelijk, er is dus een use case "configureren" en niet "camera configureren", "sensor 1 configureren, "sensor 2 configureren". Dit is gedaan om het overzichtelijk te maken. In een use case diagram wordt onderscheidt gemaakt tussen verschillende soorten gebruikers. Een normale gebruiker zal minder use cases hebben dan een beheerder of iemand met extra privileges.



Use Case Diagram

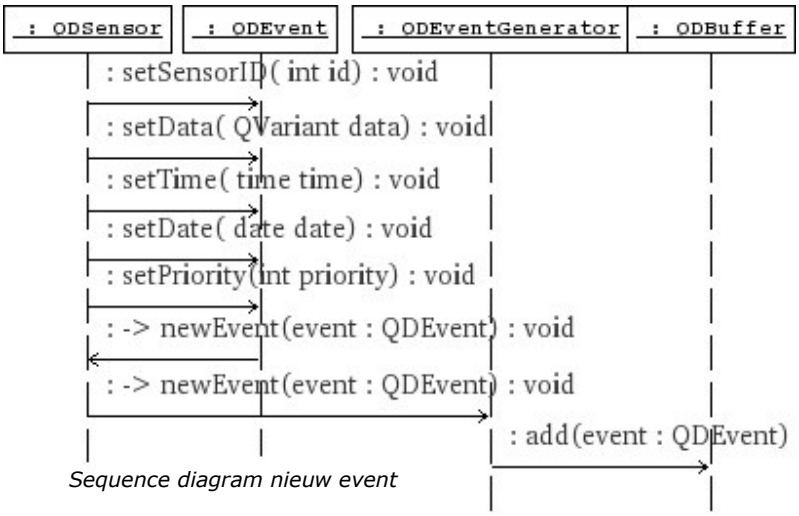
8.3 Algemeen Sequence diagram

Een Sequence diagram geeft de interne werking weer van een programma en is dan ook veel minder van belang voor de eindgebruiker, maar des te meer voor de ontwikkelaars. In een sequence diagram worden informatiestromen tussen klassen weergegeven. Zo is er duidelijk wat welke klasse doet, wie er mee communiceert en of er niet teveel dan wel te weinig informatie opgeslagen wordt.



Sequence diagram nieuw event

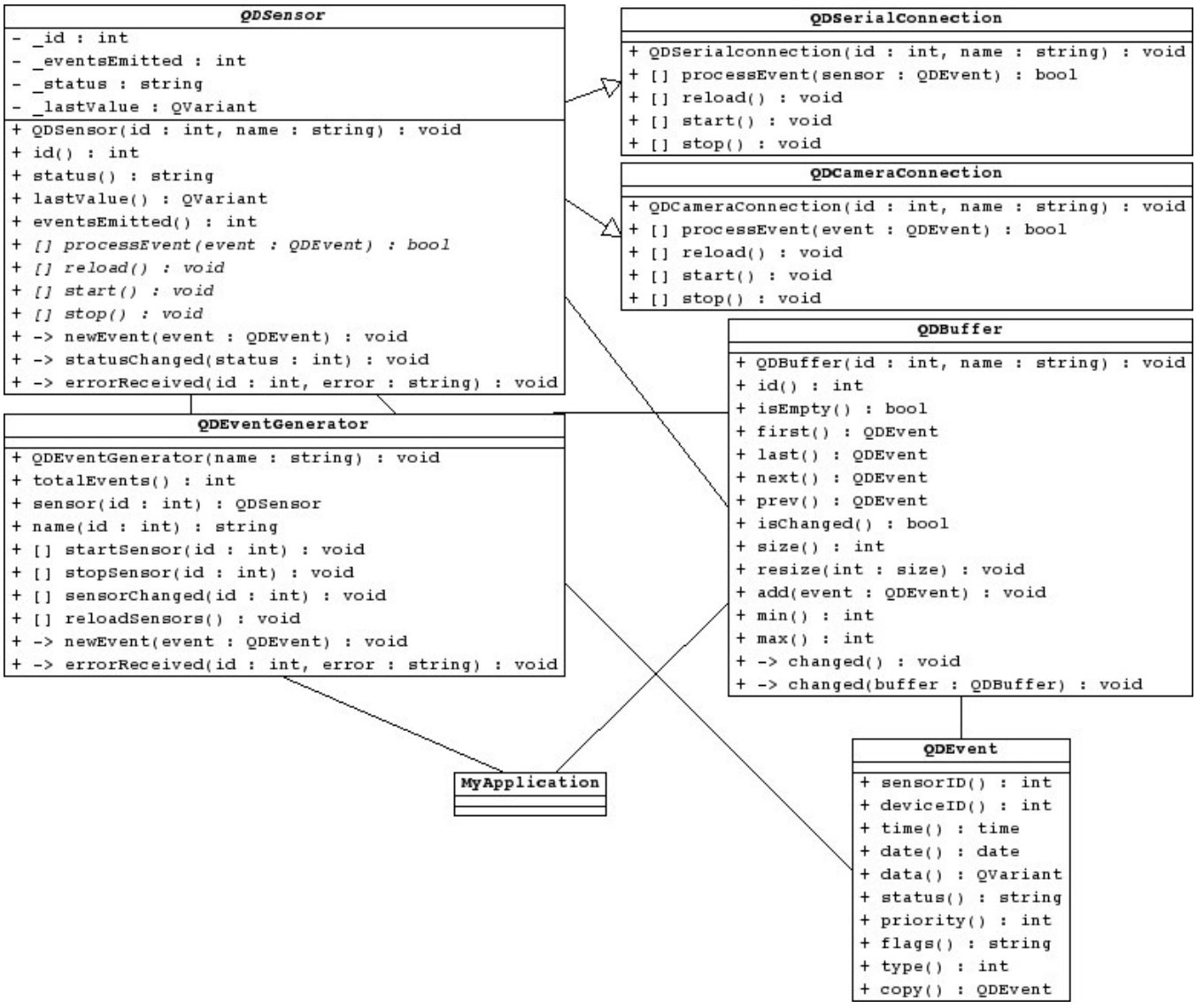
Het vorige sequence diagram geeft in een globale manier weer welke klassen op wat voor manier met elkaar communiceren en welke informatie er wordt weer gegeven.
Dit is een sterk vereenvoudigde versie bedoelt om een indruk te geven over de manier van communicatie. Het diagram aan de rechterkant is de uitgewerkte versie waarin veel beter te zien is hoe het eerste stuk van de communicatie loopt.



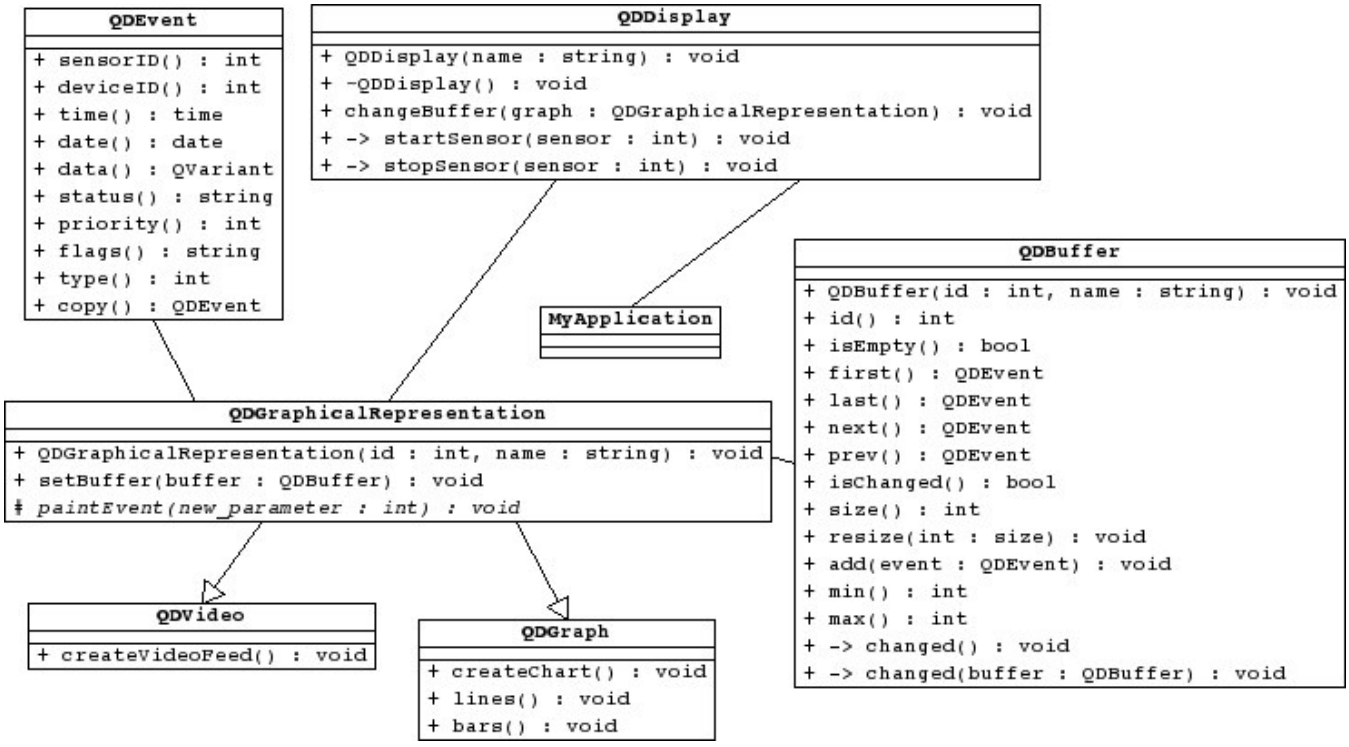
Sequence diagram nieuw event

Bijlagen

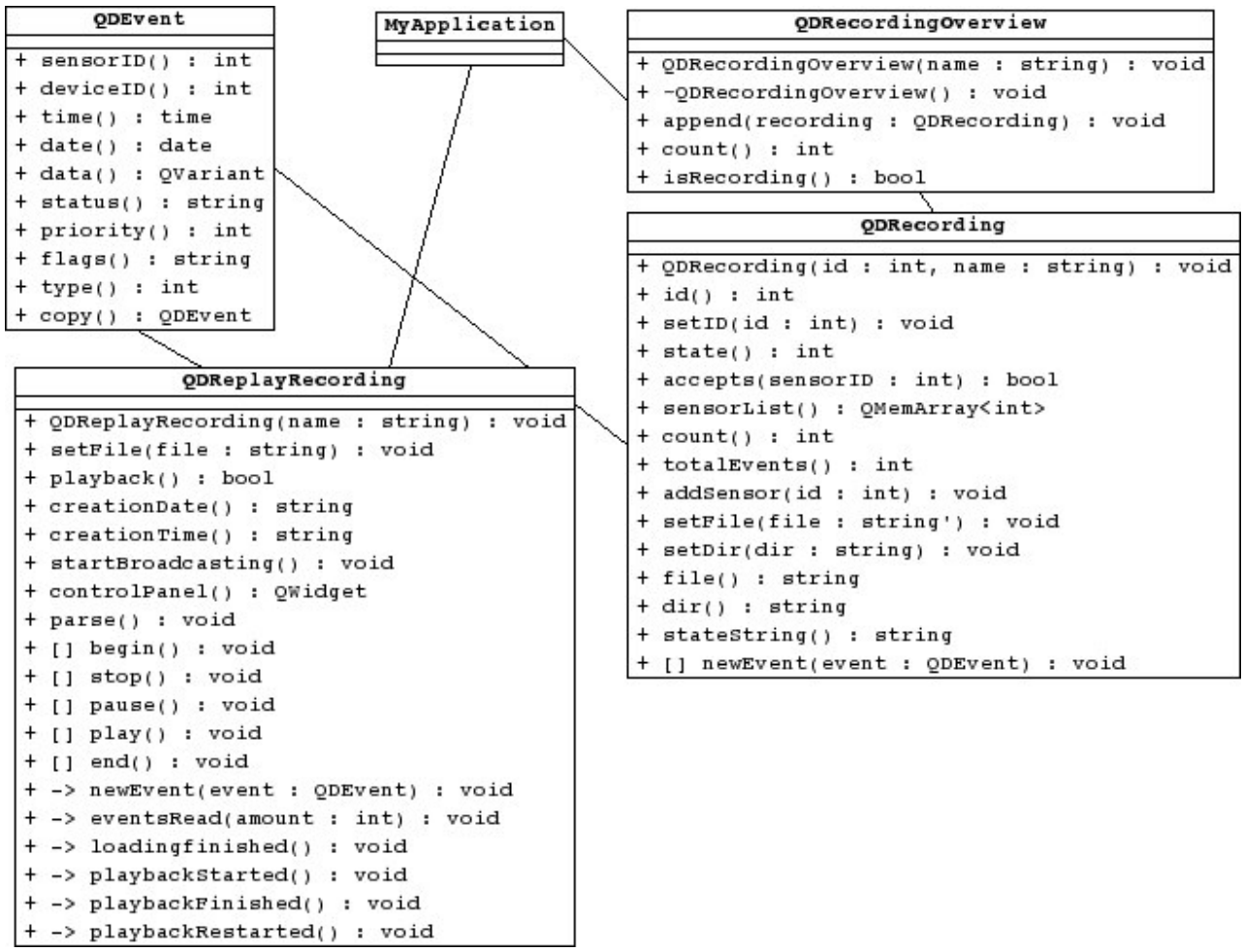
Hierna volgen een aantal diagrammen die om het overzicht te kunnen behouden niet in het document zelf maar als extra bijlage toegevoegd zijn.



Klasse diagram Informatie vergaring



Klassendiagram Display



Klassendiagram Recording

Ontwerp

Sony traject

Auteur
M.C. Van Dorp
Datum
28 Mei 2004
Versie
1.2

Inhoudsopgave

1. Inleiding

In dit *ontwerp* zullen we het in de definitiestudie besproken systeem, verder ontwerpen, zodat het klaar is om gerealiseerd te worden. Zonder ontwerp kan het programmeren in principe niet beginnen.
De functiestructuur zal beschreven worden aan de hand van toestandsdiagrammen en de gegevensstructuur zal beschreven worden aan de hand van databasemodellen. Ook zal er een indruk komen van de schermopbouw en daarmee met de lay-out van de applicatie. Als laatste wordt aangegeven hoe wij het gehele systeem gaan testen. Het testen scheiden wij daar in white- en blackbox testing, die beide later beschreven zullen worden.

2. State diagrams

Een programma dan wel object heeft tijdens de executie van het programma een bepaalde toestand. Deze toestand kan tijdens het uitvoeren wijzigen. Deze toestand wordt weergegeven in een zogenaamd *State Diagram.* In dat diagram wordt inzichtelijk gemaakt hoe de toestanden kunnen verlopen vanuit een begintoestand tot de eindtoestand.

We zullen deze toestanden beschrijven aan de hand van de indeling die in de definitiestudie gemaakt is in de drie verschillende onderdelen.

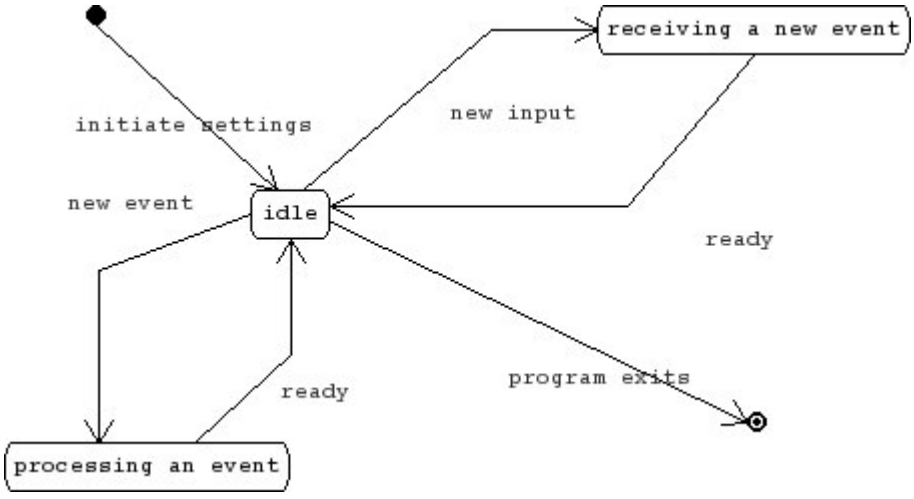
Deel 1
Het stuk aangeduid met het nummer één zorgt dat de informatie bij elkaar gebracht wordt. Via verschillende sensor implementaties (overerving van QDSensor) worden er QDEvents gemaakt en deze worden in een buffer geplaatst.

Deel 2
Het stuk aangeduid met nummer twee zorgt dat events die eenmaal aangemaakt zijn opgeslagen kunnen worden in een bepaald formaat. Via QDReplayRecording is het mogelijk deze recordings weer terug te spelen op dezelfde manier als dat nu gebeurd in deel r als dat nu gebeurd in deel één.

Deel 3
Het stuk aangeduid met nummer drie zorgt dat events die binnen komen grafisch weergegeven kunnen worden. Dit kan in de vorm van een video feed of een grafiek.

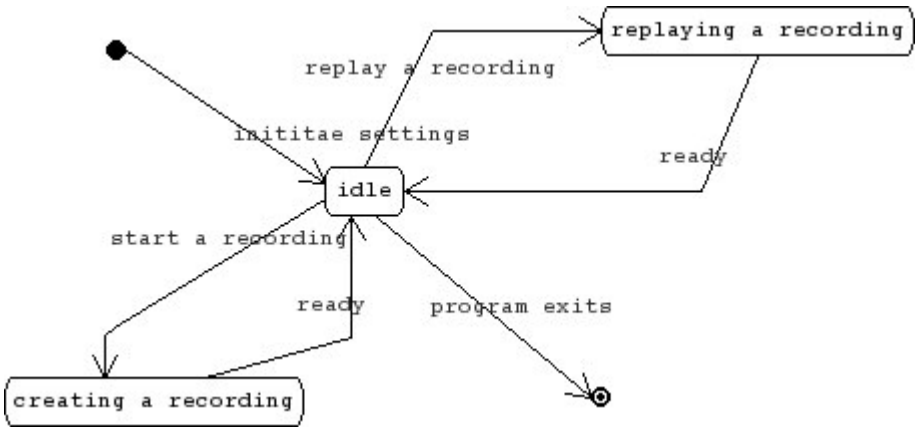
2.1 State diagram 'deel 1'

Het volgende diagram beschrijf de drie toestanden waarin deel 1 zich in kan bevinden. Normaliter is de module in een rust of idle stand, die afgewisseld kan worden met een nieuw event genereren of een nieuw event verwerken.



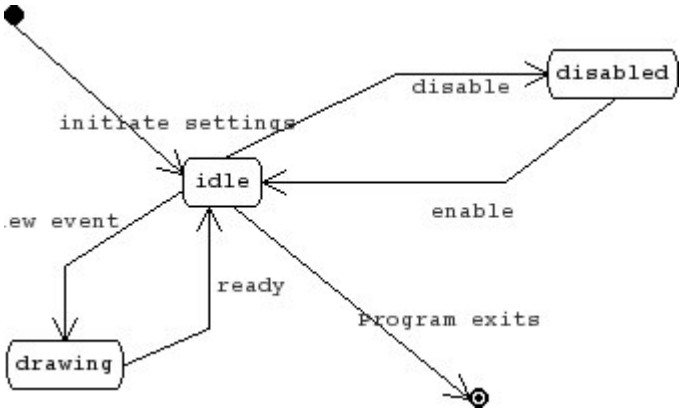
2.2 State diagram 'deel 2'

Het volgende diagram beschrijf de drie toestanden waarin deel 2 zich in kan bevinden. Normaliter is de module in een rust of idle stand, die afgewisseld kan worden met een een bestaande recording terug spelen of een nieuwe recording aanmaken.



2.2 State diagram 'deel 3'

Het volgende diagram beschrijf de drie toestanden waarin deel 2 zich in kan bevinden. Normaliter is de module in een rust of idle stand, die afgewisseld kan worden met een 'disabled' toestand waar niet gereageerd wordt op nieuwe events en een 'drawing' toestand waarin de grafieken en video-feeds geupdate worden.



3. Toekomstige werkomgeving

In dit hoofdstuk wordt beschreven wat er in de organisatie waar deze applicatie geïmplementeerd gaat worden aanwezig moet zijn om de applicatie optimaal te benutten. Dit is enerzijds een stuk hardware maar ook procedures en taakbeschrijvingen moeten hiervoor aangepast worden.

N.B. In deze beschrijving ga ik uit van een medische omgeving. Een gedeelte hiervan is toepasbaar op elke andere situatie. Een deel zeer zeker niet.

3.1 Hardware

Om de applicatie in te voeren zijn er verschillende hardware onderdelen nodig. Allereerst is dat een server waarop iFast draait. Deze server moet voorzien zijn van voldoende opslagcapaciteit. Daarnaast moet er verschillende meetapparatuur aanwezig zijn waaruit iFast informatie kan halen. Zonder deze hardware is iFast nutteloos.

3.2 Werkwijzen

Er zijn verschillende mensen die met de applicatie moeten gaan werken. Vanuit de Use cases die in de definitiestudie zijn gemaakt is al duidelijk dat er twee verschillende groepen personen zijn die met de applicatie moeten werken.

- Gewone gebruikers
- Beheerders

Nu zijn de gewone gebruikers vooral druk bezig met de meetwaarden aflezen van de apparatuur zelf. De beheerders sluiten de apparatuur aan en testen deze. Daarna is hun werk gedaan.

De gewone gebruikers zullen in de toekomst de recordings starten en afsluiten, recordings terug spelen en basis instellingen veranderen. De beheerders zullen bestaande recordings kunnen weggooien en de wat geavanceerdere instellingen veranderen.

3.3 Taakbeschrijvingen

De werkwijzen veranderen en dus de taakomschrijvingen ook.
De gewone gebruikers zullen een soort cursus moeten volgen om met de applicatie om te kunnen gaan. De beheerders zullen een extra stuk kennis op moeten doen over de applicatie zelf, maar ook over netwerken en andere computertechnologieën.

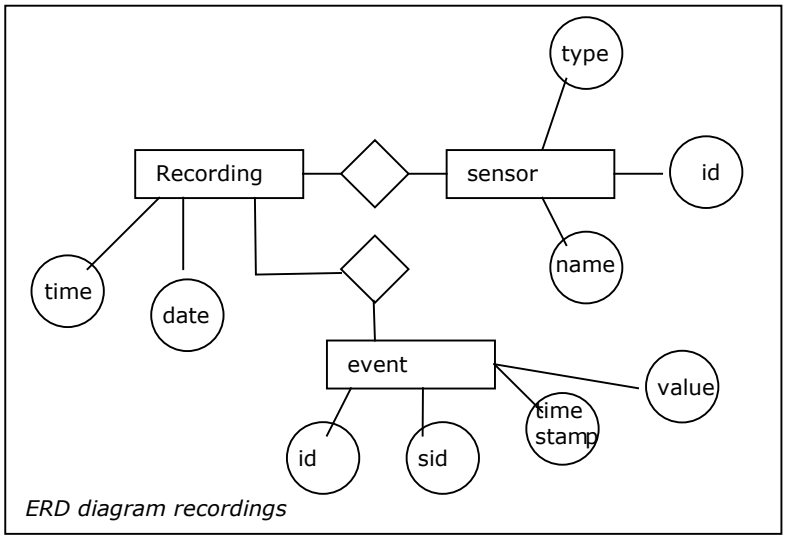
4. Relationeel Representatiemodel

Binnen elke applicatie is er wel iets wat opgeslagen moet worden. Dit kunnen configuratiegegevens zijn, meetwaarden of uitkomsten van berekeningen. Zo ook in deze applicatie. Elk type sensor heeft zijn eigen waarden die nodig zijn voor het functioneren.

Zo ook deze applicatie. Meetwaarden die worden ontvangen moeten worden opgeslagen in een formaat dat handig is te versturen en terug te lezen. Ik heb hier gekozen voor een XML file. Dit zorgt dat ook externe applicaties de mogelijkheid hebben om deze informatie in te lezen en te verwerken. Daarnaast is het makkelijk uit te bereiden als er veranderingen komen in de manier van opslaan of de data behoefte.

Omdat dit opgeslagen moet worden in een file is er een ERD diagram gemaakt dat er als volgt uitziet:

In één recording kunnen één of meerdere sensoren voorkomen. Elk van deze sensoren kan nul of meerdere events hebben die allemaal in de recording opgenomen zijn.



ERD diagram recordings

```
- <iFast>
    <time value="16:21:24" />
    <date value="Mon May 10 2004" />
  - <sensors>
      <sensor id="0" name="Serial sensor 1" type="serial" />
      <sensor id="1" name="Serial Sensors 2" type="serial" />
    </sensors>
  - <events>
      <e id="1" sid="0" ts="58884625">59</e>
      <e id="2" sid="1" ts="58884655">57</e>
    </events>
  </iFast>
```

XML representatie van het ERD diagram

Het aantal events dat in één file past is theoretisch gezien niet gelimiteerd. Het is alleen niet aan te raden de bestanden groter te laten worden dan 100MB. Als files zo groot worden, worden ze onhandelbaar en lastig in te lezen.

5. Schermontwerp

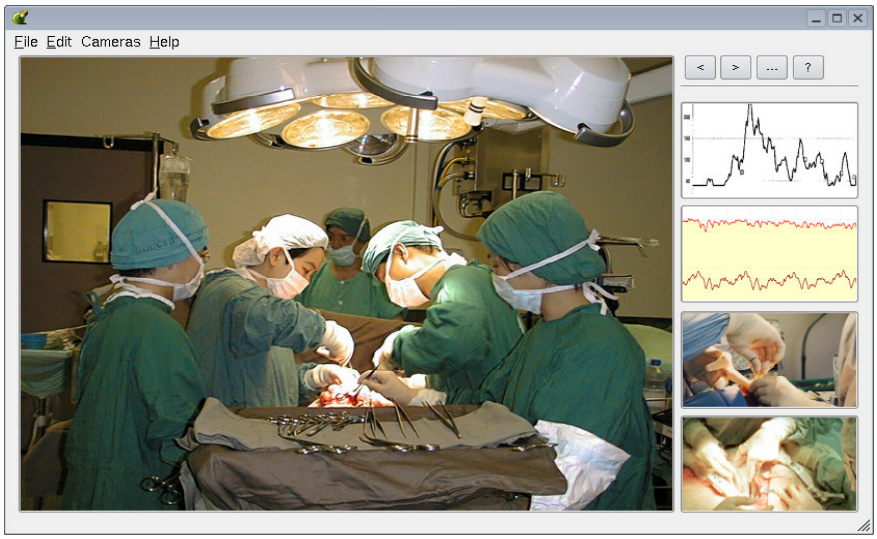
Het eerste grafische ontwerp maken is een vrij lastige klus. Je weet nog niet precies wat de gebruikers voor interface willen en je moet dus zelf wat bedenken aan de hand van voorbeelden van programma's van concurrenten. Ik heb daarom eerst met behulp van een tekenprogramma wat schetsen zitten maken. Het resultaat hiervan lijkt op de echte applicatie maar is niet functioneel. Zo'n schets wordt een mockup genoemd. In het schermontwerp wordt vastgelegd **wat** er weergegeven wordt en het accent ligt minder op **hoe** dat gebeurd.

Bij het schermontwerp moet er rekening gehouden worden met de volgende eisen die aan de interface gesteld worden:

ID	Userinterface eis	Prioriteit
U1	Applicatietaal moet engels zijn	Basic
U4	Leuke icons	Luxe
U6	Applicatie moet te vertalen zijn naar andere talen	Luxe
U7	Alle informatie die getoond moet worden moet een eigen plek hebben	Basic

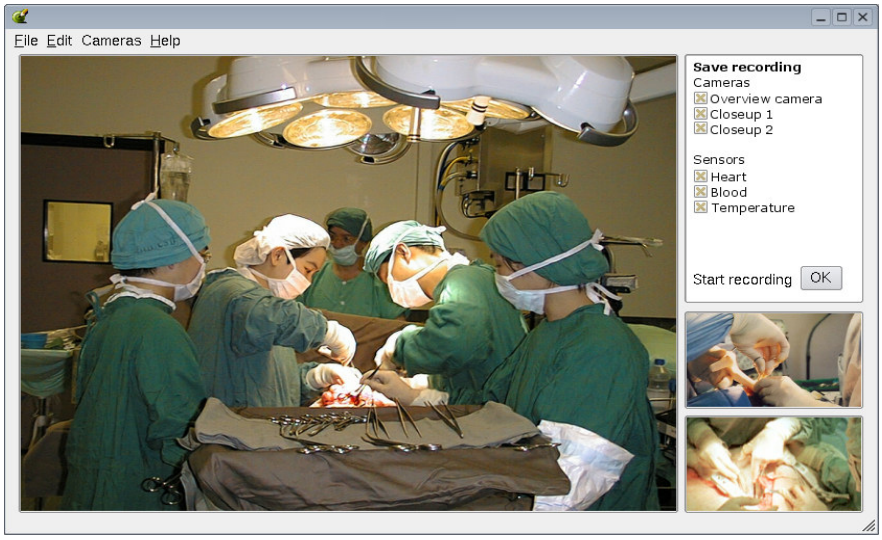
Omdat de applicatie eerst voor Healthcare gebruikt zal gaan worden en daarna pas in een andere branche heb ik gekozen om een interface te ontwerpen die aansluit bij bestaande medische applicaties en hardware.

Het idee is dat je via een touchscreen display of een muis op de grafieken aan de rechterkant kunt duwen. Dan wisselt deze van plaats met de grote grafiek aan de linkerkant. Grafieken en videobeelden zijn van gelijke orde en kunnen dus allemaal vergroot en verkleind worden.



Eerste ontwerp interface overzichtscherm

In het eerste ontwerp zat ook het idee om zonder popup vensters te werken. Een popup venster is een venster dat voor de applicatie geplaatst wordt en gebruikt wordt om een vraag te stellen of informatie te geven. Voor een gevorderde gebruiker is het gebruik van popup schermen aan

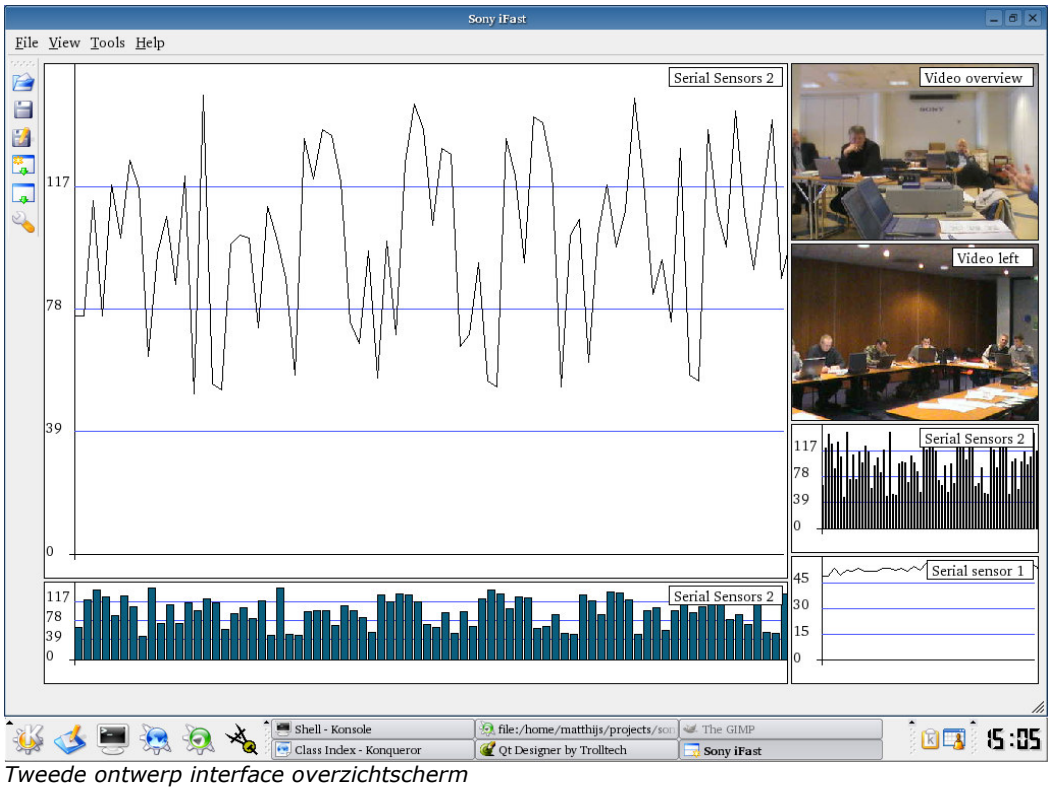


te raden. Doordat het een apart scherm is wordt duidelijk afgebakend wat bij elkaar hoort. Een beginnende gebruiker kan echter makkelijk 'verdwaald' raken in alle schermen.

De vragen worden nu niet via een los scherm maar in de bestaande interface gesteld. Dit heeft als voordeel dat de gebruiker altijd weet waar hij is in de applicatie. Het heeft als nadeel dat het erg beperkt is omdat er weinig ruimte beschikbaar is. Dit kan opgevangen worden door ook de plek van de grote grafiek beschikbaar te stellen.

Tweede grafische ontwerp

Het overzichtsscherm, of de standaard lay-out, is iets veranderd. Onder de grote grafiek aan de linkerkant is een extra grafiek gekomen. Dit heeft als reden dat veel grafieken en videobeelden een breedte/hoogte verhouding hebben van 3:4. Dit zou betekenen dat er een deel van het scherm niet gebruikt wordt wat niet zinvol is.



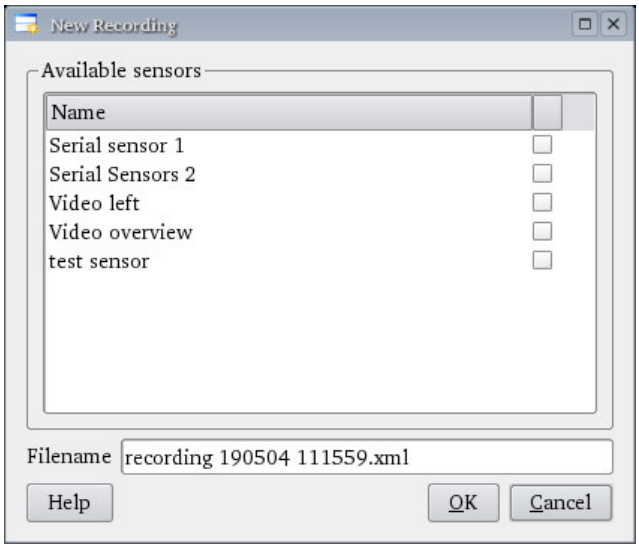
Tweede ontwerp interface overzichtscherm

Het extra veld dat hiermee gecreëerd is gedraagt zich anders dan andere velden. Zo kan het alleen grafieken weergeven en geen videobeelden.

Dit heeft te maken met de beeldverhoudingen die niet gerealiseerd kunnen worden. Ook kan de grafiek niet van plaats gewisseld worden met een andere grafiek.

Ook het idee om geen popup scherm te gebruiken is gesneuveld. Vooral het gebrek aan ruimte was een erg groot probleem in sommige configuratievensters.

Als alternatief is er gekozen om toch popup schermen te gebruiken maar deze op een standaardmanier in te richten zodat de gebruiker altijd weet waar het scherm voor dient. In een configuratiedialoog is er ook altijd een button "Help" aanwezig die direct verwijst naar een relevante help pagina.



Tweede ontwerp Interface recording scherm

6. Testen

Voor het testen van de applicatie moet een duidelijk gestructureerd testplan gemaakt worden met overzichtelijke testcases. Dit wordt uitgesplitst in de eerder behandelde white- en blackbox test vormen. Na het uitvoeren van de testcases worden de resultaten gedocumenteerd en verder verwerkt tot een probleemlijst. Vanuit deze probleemlijst kan een pugilist gemaakt worden die terug gestuurd kan worden naar de ontwikkelaars.

6.1 Blackbox testen

Blackbox testen is simpeler. Om een goede blackbox test uit te kunnen voeren moeten er testsets gemaakt worden die de volgende regels controleren.

- Voldoet het gedrag aan de verwachtingen
- Is het gedrag consistent
- Worden er duidelijke foutmeldingen en vragen gesteld

Er zijn oneindig veel testsets te maken, de kunst is echter om juist die tests uit te voeren die net wel of net niet binnen de toegestane waarden liggen.

Als er een inconsistentie of probleem gevonden is dan moet dit genoteerd worden in de resultatenlijst.

6.2 Whitebox testen

Tijdens het whitebox testen wordt de source code van de applicatie nagekeken op fouten. Dit kunnen fouten in de werking van een (deel) van een klasse, onvoldoende of foutief commentaar of mogelijkheden tot ongeautoriseerde toegang. Deze vorm van testen moet vrij gestructureerd aangepakt worden omdat de fouten in het algemeen vrij klein zijn en niet opvallen. Hiervoor zijn er een aantal regels opgesteld waaraan de code moet voldoen.

- Variabelen moeten altijd geinitialiseerd worden
- Pointers altijd initialiseren met een 0 waarde
- Bij delen altijd checken op ‘delen door 0’
- Waarden die van buiten het systeem komen altijd controleren op lengte en geldigheid
- Strings die worden geconverteerd naar integers checken op 0 waarden

Als er in de code één of meer overtredingen van deze regels voorkomen dan moet dat genoteerd worden. Al deze dingen kunnen dan in één keer aangepakt worden. Hierna kan een nieuwe versie van de applicatie opgeleverd worden zonder al deze fouten.

Sony iFast

Testplan

Auteur
M.C. Van Dorp
Datum
1 Juni 2004
Versie
1.0

1. Inleiding

Voor het testen van de applicatie moet een duidelijk gestructureerd testplan gemaakt worden met overzichtelijke testcases. Dit wordt uitgesplitst in de eerder behandelde white- en blackbox test vormen. Na het uitvoeren van de testcases worden de resultaten gedocumenteerd en verder verwerkt tot een probleemlijst. Vanuit deze probleemlijst kan een pugilist gemaakt worden die terug gestuurd kan worden naar de ontwikkelaars.

2. Blackbox testen

Blackbox testen is simpeler

Om een goede blackbox test uit te kunnen voeren moeten er testsets gemaakt worden die de volgende regels controleren.

- Voldoet het gedrag aan de verwachtingen
- Is het gedrag consistent
- Worden er duidelijke foutmeldingen en vragen gesteld

Er zijn oneindig veel testsets te maken, de kunst is echter om juist die tests uit te voeren die net wel of net niet binnen de toegestane waarden liggen.

Als er een inconsistentie of probleem gevonden is dan moet dit genoteerd worden in de resultatenlijst.

3. Whitebox testen

Tijdens het whitebox testen wordt de source code van de applicatie nagekeken op fouten. Dit kunnen fouten in de werking van een (deel) van een klasse, onvoldoende of foutief commentaar of mogelijkheden tot ongeautoriseerde toegang. Deze vorm van testen moet vrij gestructureerd aangepakt worden omdat de fouten in het algemeen vrij klein zijn en niet opvallen. Hiervoor zijn er een aantal regels opgesteld waaraan de code moet voldoen.

- Variabelen moeten altijd geinitialiseerd worden
- Pointers altijd initialiseren met een 0 waarde
- Bij delen altijd checken op ‘delen door 0’
- Waarden die van buiten het systeem komen altijd controleren op lengte en geldigheid
- Strings die worden geconverteerd naar integers checken op 0 waarden

Als er in de code één of meer overtredingen van deze regels voorkomen dan moet dat genoteerd worden. Al deze dingen kunnen dan in één keer aangepakt worden. Hierna kan een nieuwe versie van de applicatie opgeleverd worden zonder al deze fouten.

4. Testresultaten

Voor het blackbox testen zijn er een aantal testsets gemaakt die uitgevoerd zijn. In de tabel hieronder is te zien hoe dit is afgelopen

De resultaten uit de blackbox test

Nr	Testcase	Resultaat	Correct
1	Er wordt een nieuw event gegenereerd	Het event wordt opgeslagen in de buffer	✓
2	Verander de instellingen van een sensor	De sensor herlaad zelf zijn instellingen en blijft functioneren	✓
3	Er wordt een nieuwe lay-out geladen	De juiste beelden worden getoond	✓
4	Maak een nieuwe recording aan	De recording springt automatisch over naar een volgende file bij de maximale grote	✓
5	Lees een recording in en probeer deze af te spelen	Video events worden niet weergegeven	✗
6	Disconnect een sensor en zie hoe het programma hierop reageert	Er wordt een foutmelding gegenereerd in de centrale event logger	✓
7	Genereer een nieuw data event en kijk wat er gebeurd	Indien nodig wordt de bijbehorende grafiek opnieuw getekend	✓
8	Zet een diagram stil en kijk dan of deze op de één of andere manier beweegt	Dit gebeurd in vrijwel alle gevallen behalve het resizen van een window	✗
9	Stop een actieve recording	Het is mogelijk een actieve recording te stoppen, deze weggooien zodat deze niet meer in de lijst staat kan echter niet	✗
10	Exporteer de errors naar CSV of XML	Alles wordt netjes opgeslagen in een XML file	✓

De resultaten uit de whitebox test

Nr	Filename	Gevonden fout
1	Alle files	Er wordt regelmatig gebruikt van een variabele ‘tmp’ of ‘tmp2’. Dit is onduidelijk en moet veranderen
2	Alle files	Er worden regelmatig hele lange commando’s gebruikt (functie().functie2().functie3().functie4()) Dit is erg onoverzichtelijk en moet zo veel mogelijk vermeden worden.
3	Graphicalrepresentation.cpp	Niet overal wordt op ‘delen door 0’ gecontroleerd
4	Serialconnection.cpp	De ontvangen input wordt niet gecontroleerd op een buffer overflow
5	Cameraconnection.cpp	De manier van implementatie moet eigelijk anders zodat er echt streaming ondersteund wordt
6	Sensoroverview.cpp	Nog geen gebruik gemaakt van de centrale error logger

Bij elkaar zijn er niet vreselijk veel fouten gevonden. Dit kan zijn doordat de maker alles getest heeft en dus veel dingen logisch lijken die het eigenlijk niet zijn. Dit zelfde kan gelden voor de source code.

User Manual

Sony traject

Writer
M.C. Van Dorp
Date
7 June 2004
Version
1.0

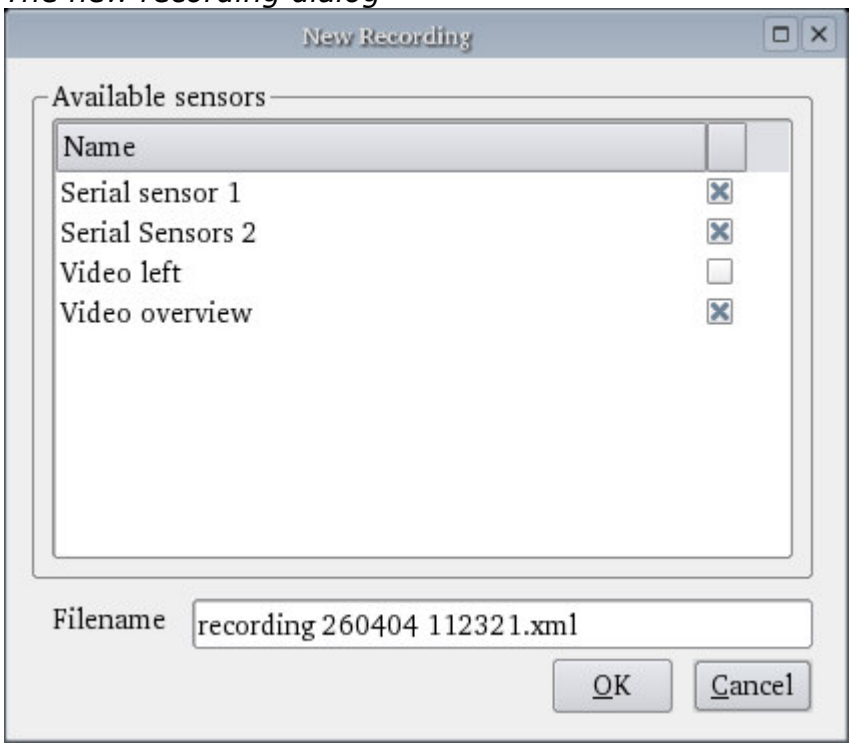
Table of contends

1. Recordings

Sony iFast can record events that are generated in an XML file. With the special viewer you can replay the recording for evolutional purposes
A recording can contain data from several sources, including metadata.

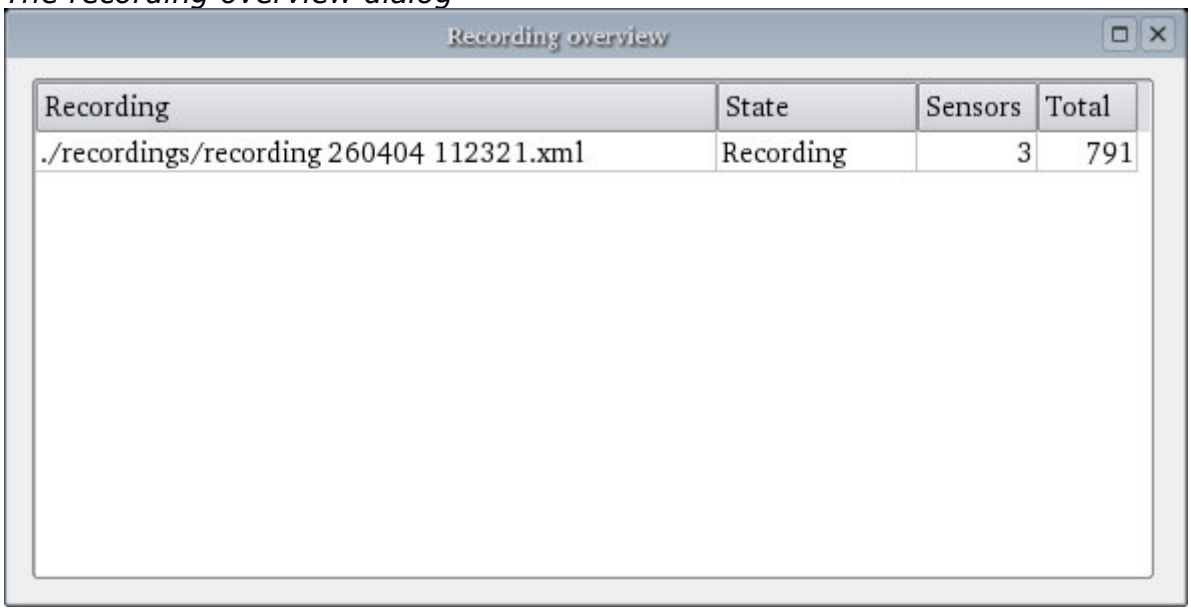
Starting a recording
This part describes the steps to start a recording
Step 1
Click on the "Start new recording" icon in the menu bar. A dialog will appear
Step 2
Click on a checkbox to add the sensor to the new recording
Note If 'disable logging' is disabled on a sensor the name will not show up in the list of available sensors
Step 3
Change the name of the file to a different name (if necessary), the directory the files are placed in, is defined in the preferences dialog
Step 4
If you selected all the sensors that you need you can click on the OK button to start the recording.

The new recording dialog



Managing recordings
While running iFast you can always check the current recording by clicking on the "Recording overview" icon in the menu bar. This will launch a dialog with a list of all the current recordings. For each recording the following values are specified: filename, state, amount of sensors and the total amount of events that are recorded.

The recording overview dialog



From here you can stop or resume recordings. You can do this by right clicking on the row with the desired recording.
Note Resuming a recording means overwriting the old file and start again. This behavior is undesirable and is subject to change
Note 2 This list will refresh every 2 seconds to give an up to date view

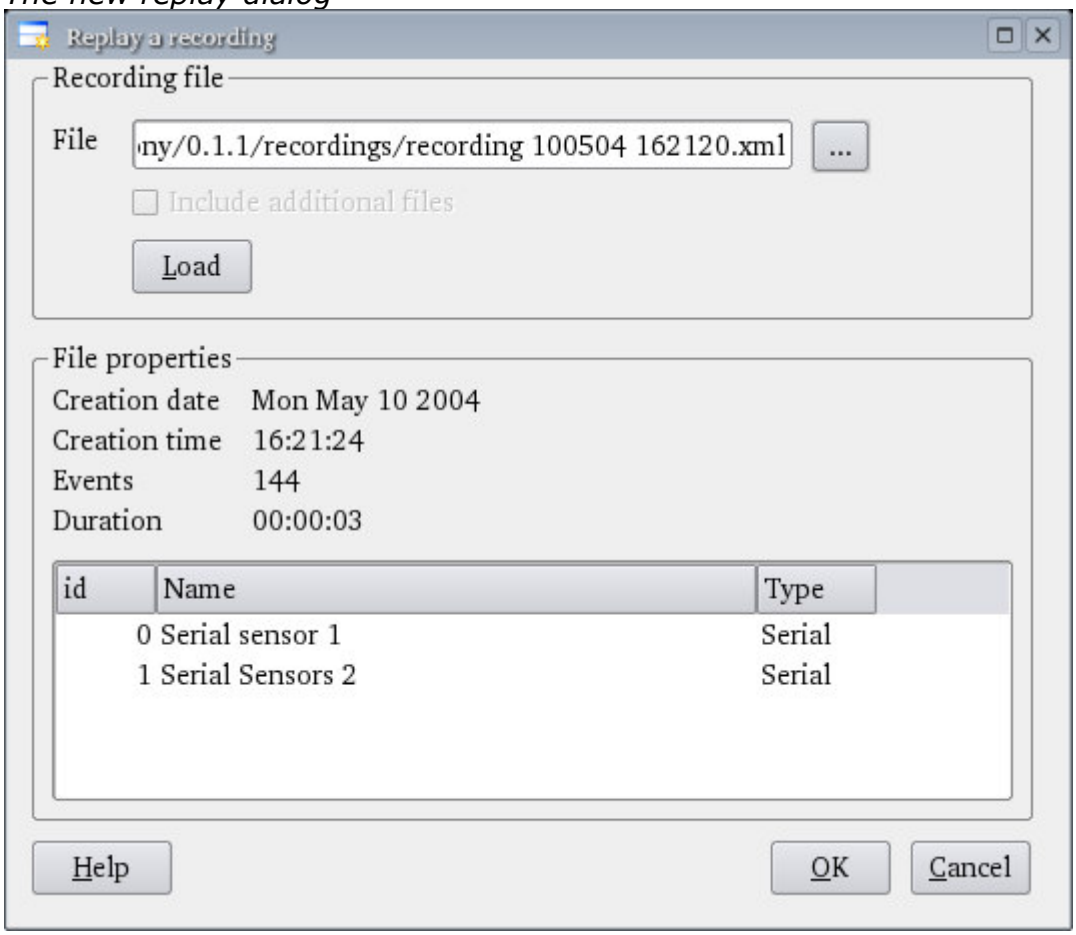
2. Replay a recording

Replaying a recording is one of the most useful features of iFast. You can take a recording, generated by you or someone else and load it.
The recording will start playing at the same speed it was recorded. With the control panel at the bottom right you can stop the replay or skip pars of it.
Note Starting a replay will stop all current recordings. It will also make the program ignore all the events generated during the replay.

Starting a replay
This part describes the steps to start a replay of a recording file.
Step 1
Click on the "Recording playback" icon in the menu bar. A dialog will appear
Step 2
Select the desired recording. The names (if not manually changed) represent the day and time the recording was made.
The file 'recording 100504 162120.xml' was created on 10/05/2004 at 16:21:20.
Step 3
Click on the 'Load' button. This will load the entire file onto memory and makes it available for the system. After it finishes loading, some basic information is displayed like duration of the recording, names of the sensors and the time and date it was created. In future releases this might be extended with a name of the creator or other valuable information
Step 4
If you have loaded the right file you can click on the 'OK' button, this will close the dialog and start the replay

Note It's not possible to replay more then one recording at the time. This behavior is by design.

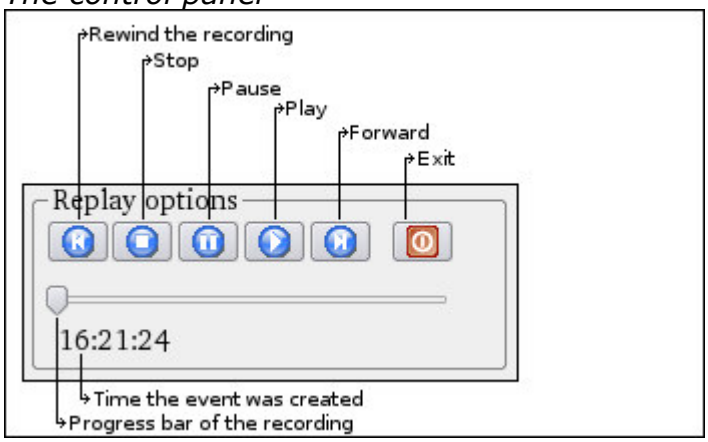
The new replay dialog



Using a replay

This part describes the actions during a replay action.
When you load a recording file into the system a small control panel appeared in the bottom right of the screen. There are a few buttons on it with the same functions as let say a CD player.

The control panel



For now the recordings are replayed at the same speed that they were recorded. In future releases there will be the possibility to replay at 0.5x, 2x, 4x and 8x the speed of the original recording.

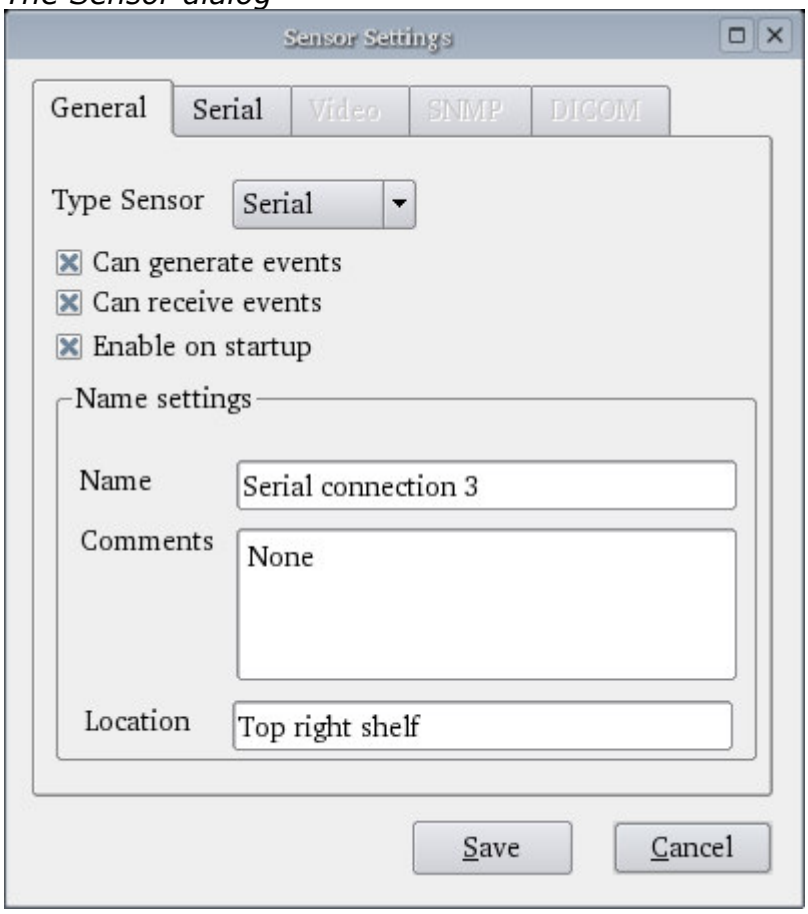
3. Frequently asked questions

In this program a sensors is defined as a (virtual) device that can generate events. With the default settings you can use serial sensors and Sony Security cameras.
Note It's possible to write custom plugins to add extra sensors.

Adding a new sensor
These are the steps to add a sensor
Step 1
Click on 'tools > Edit sensors' or press 'Control + E' to open the dialog with the available sensors
Step 2
Click on 'Add a new device' to open the dialog for configuring the new sensor
Step 3
Select the desired sensor type (default this supports serial and camera's) This will reveal a new tab where you can configure specific options
Step 4
Enter a name, description and a location
Step 5
If all the information is entered correctly you can click on OK to finish the process

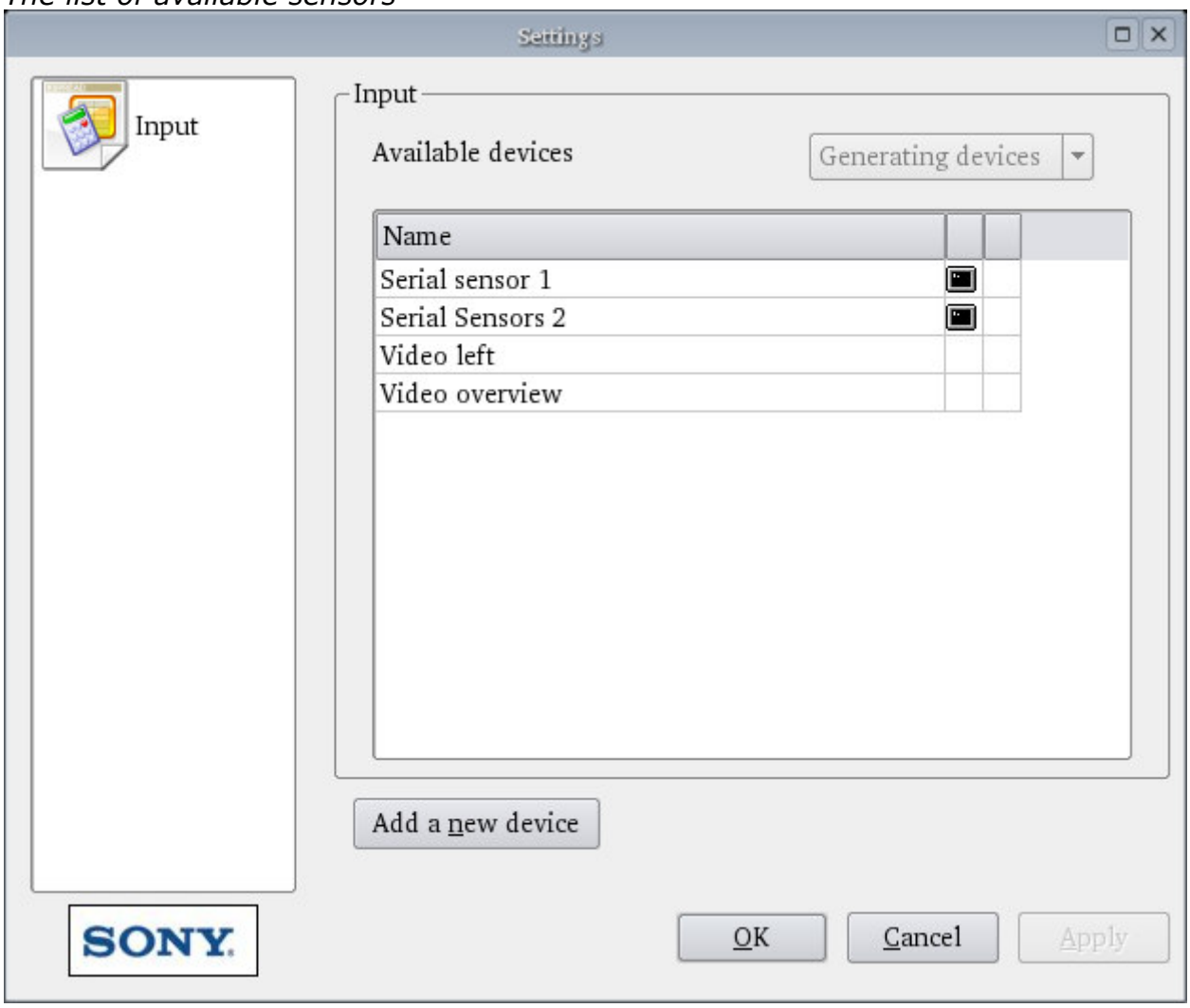
The new sensor is now initialized and available for use

The Sensor dialog



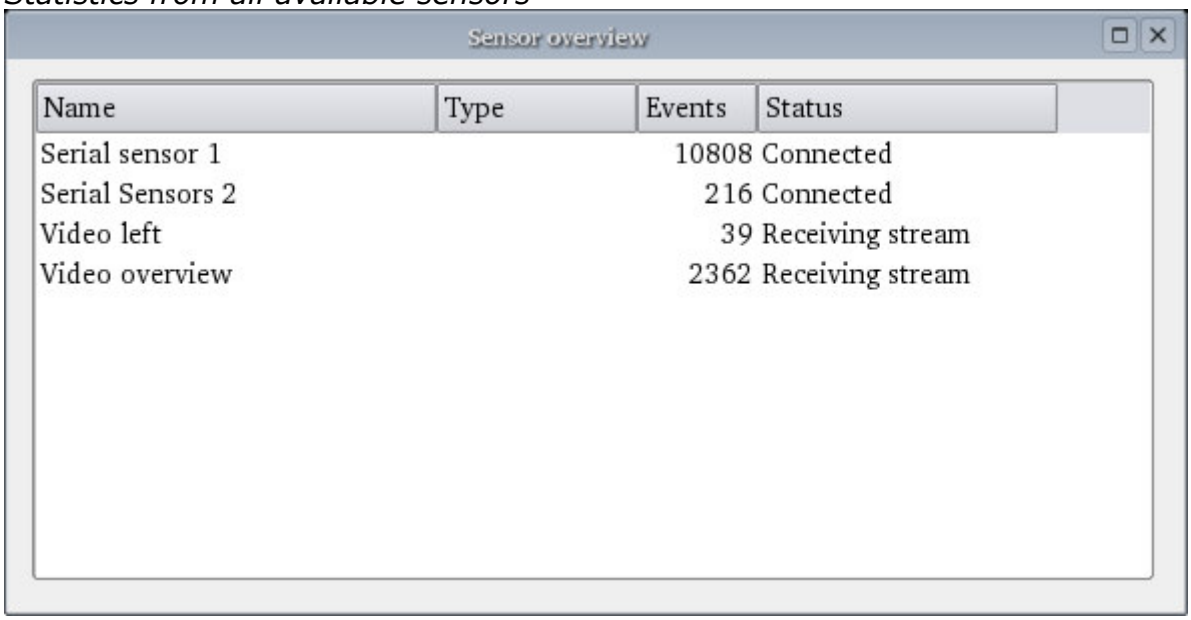
Managing sensors
You can edit all the available sensors via the 'Edit sensors' menu entry. Double click on a sensor name and you get the same dialog as for creating a new sensor.

The list of available sensors



For viewing a list of statistics from all the sensors you can click on 'view > Sensors'. A new dialog will appear with a list of the sensor names, types, amount of events generated and a status description.

Statistics from all available sensors



Note This list will refresh every 2 seconds to give an up to date view

4. Frequently Asked Questions

Here is a list of most encountered questions.

What is iFast?
iFast is en program that can receive different types of events, store them and represent them in a graphical way. It's functionality can be extended by adding new modules like remote managing or new types of sensors.

How do I start a recording
How recordings work, how to start one and a little technical background is described in the 'Recording' page in the manual.

How do I start a recording replay
How to start a recording replay is described in the 'Recording Replay' page in the manual.

Which display resolutions can I use?
The application is developed for an optimal resolution of 1024x768 pixels. The minimum size is 640x480 but this is not advised, as there are some layout problems. Theoretically it's usable on screens that don't use a 4/3 ratio so displaying on a plasma screen should be no problem.

For additional questions contact your sales contact.

5. Shorts cuts

For higher productivity and easier use there are a few shortcuts defined. These short cuts help you navigate trough the program without using your mouse. In future versions there will be more shortcuts so the program can be used without a mouse.

Key combination	Description
F1	Open the help browser
Control + O	Open a new display configuration
Control + S	Save the display configuration
Control + E	Open the edit screen for the available sensors
Control + R	Open the edit screen for general recording options
Control + P	Open the preferences screen
Alt + N	Start a new recording
Alt + O	Open the overview screen for all current recordings

Technical Manual

Sony traject

Writer
M.C. Van Dorp
Date
7 June 2004
Version
1.0

Introduction

This is the technical manual of the Sony iFast application created for Sony Europe. This document describes the program, how it was created and where to find the product documentation. It contains a listing of the different people who worked on the development.

Used products

There are a number of tools used during the development of this program.

- QT (www.trolltech.com/products/qt/) A multi-platform graphical toolkit based on C++
- KDoc (http://www.ph.unimelb.edu.au/~ssk/kde/kdoc/) A documentation generation tool used for auto-generating a part of the product documentation
- GIMP (www.gimp.org) Graphics program used for editing and creating icons.

People involved

There are a number of people involved in the design and creation of this program. They all work at the Professional services department of Sony Europe.

Martin Diependaal martin.diependaal@eu.sony.com
Matthijs van Dorp matthijs.vandorp@eu.sony.com
Gerard Bes gerard.bes@eu.sony.com

Design

There are a number of UML (http://www.uml.com) diagrams available concerning this product. The diagrams are made with Umbrello (http://uml.sourceforge.net/) , a free available UML editor and have the extension .xmi .

You can download them from the Sony network
http://43.194.180.188/files/

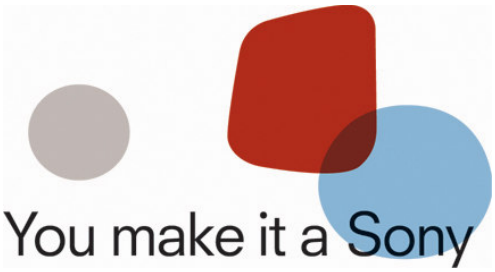
Documentations

There is an extensive product documentation including API documentation and examples online on the Sony network.

You can download it here
http://43.194.180.188/
http://43.194.180.188/iFast/

Procesbespreking

AMC traject



Auteur
M.C. Van Dorp
Datum
27 Mei 2004
Versie
1.1

Voorwoord
Dit document is gemaakt naar aanleiding van de periode dat ik extern geplaatst ben bij het AMC ziekenhuis in Amsterdam. In dit document bespreek ik mijn ervaringen, producten en resultaten die geboekt zijn. Daarnaast geef ik een oordeel over de samenwerking met mensen van het AMC.

Inhoudsopgave

1. Doel

Het doel van de opdracht is het onderzoeken of er de mogelijkheid is tot het maken van een link of koppeling tussen bestaande medische apparatuur en Sony iFast.

Via Sony Nederland is er een gelegenheid gecreëerd waarin ik voor drie weken extern bij het AMC ziekenhuis geplaatst werd om hier onderzoek te doen naar de verschillende apparatuur die hier aanwezig was en wat de verbindingsmogelijkheden waren.
Het gaat hier specifiek om apparatuur die in de nieuwe traumarooms geïnstalleerd is.
Er moet informatie uit deze apparatuur gehaald worden die op een gestructureerde manier opgeslagen kan worden en weer terug gespeeld kan worden voor evaluatiemogelijkheden.

2. Personen

Binnen het AMC heb ik met verschillende personen samen moeten werken. Een aantal van deze personen was direct betrokken met het traumateam, de andere personen maakten deel uit van het technisch ondersteunend team.

Matthijs van Dorp
Ikzelf. Vierdejaars student aan de Haagse Hogeschool die bij Sony Europe bezig is met afstuderen op een medisch onderwerp.

Ping Fung Kon Jin (onderzoeker)
Ping is een student op de Universiteit van Amsterdam die waarschijnlijk voor stage/afstuderen of onderzoek tijdelijk gestationeerd is binnen het AMC. Hij doet daar onderzoek naar automatiseringsmogelijkheden van de traumakamers en ziekenhuizen. Tijdens de drie weken dat ik bij het AMC werkte was hij zeker anderhalve week niet aanwezig.

Jan Hendriks (Instrumenteel bureau AMC)
Jan is een technisch medewerker bij het Instrumenteel bureau AMC wat onder andere alle techniek voor de operatiekamers verzorgt. Hij schijnt een expert te zijn op het gebied van de apparatuur die aangesloten moest worden maar kon me ondanks dat niet echt verder helpen. Hij had het vaak te druk en werkte onregelmatige tijden. Hij was niet echt blij met mijn komst. Op een vrij netjes mailtje dat ik hem stuurde kreeg ik een reply die voor zijn baas bedoelt was waarin hij vroeg of mijn situatie al opgelost was. Dit bleek over kostenposten te gaan die niet te plaatsen waren.

Aukje Kuntze (Manager Traumazorg)
Aukje is volgens haar kaartje de algemene manager traumazorg maar ik ben er niet echt achter wat ze nou echt doet. Ze is erg behulpzaam en weet een hoop van het gebruik van de apparatuur. Qua techniek weet ze echter niets. Ze heeft me een aantal rondleidingen door de traumakamers gegeven waardoor de lokale situatie wel ongeveer duidelijk is.

3. Taken

De taken die in het kader van dit project uitgevoerd moeten worden zijn:

- Onderzoeken apparatuur op mogelijkheden om een geschikte koppeling te maken
- Opdoen van praktische kennis en dit te combineren met theoretische ideeën.
- Weer aansluiting krijgen bij het AMC ziekenhuis

4. Uitkomsten

Tijdens de drie weken liep ik tegen meer problemen aan dan ik verwacht had. Apparatuur bleek een stuk vreemder in elkaar te zitten dan ik verwacht had, complete afdelingen werkten volkomen langs elkaar heen en leveranciers/producenten wisten vaak zelf geeneens antwoord op vragen die ik had. Ook was niet alle hardware voorradig zodat ik niet alles fysiek heb kunnen onderzoeken.

Door deze factoren waar van te voren niet echt rekening mee gehouden was is het project een beetje anders gaan lopen. Ik heb totaal maar twee van de vijf apparaten fysiek kunnen onderzoeken omdat de rest in gebruik of niet voorradig was. De andere drie apparaten heb ik alleen op papier, met behulp van de handleidingen die ik pas de laatste dag kreeg, kunnen onderzoeken. Hier is dus niet al te veel uitgekomen.

Verzoeken aan fabrikanten om protocolspecificaties hebben allemaal tot niets geleid. Of deze vraag aan de verkeerde persoon gesteld is of dat deze informatie überhaupt niet vrijgegeven wordt is niet te zeggen.

5. Producten

Vanuit het AMC lag de vraag er om naar vijf verschillende medische apparaten te kijken.
Het gaat hier om:

- Philips H3046A, Monitor (ECG, SP)2, NIBD, Druk, CO2, Temp etc)
- Hamilton Medical Adaptive Support Ventilation, Raphael Color
- B|Braun Perfusor FM
- B|Braun Infusomaat P
- Level one systeem D-100 1000

5.1 Philips Monitor

Product
Philips H3046A, Monitor (ECG, SP)2, NIBD, Druk, CO2, Temp etc)
serienummer
M3046a

Soorten communicatie
- RJ-45
- VGA aansluiting
- Alarm poort (waarschijnlijk maak breek contact)

Productbeschrijving
http://www.medical.philips.com/main/products/patient_monitoring/products/m3_m4/

> Designed for flexible continuous monitoring, the M3 and M4 monitors offer wireless capability for adult, pediatric and neonatal patient care. M Series monitors are equipped with a Multi-Measurement Server that includes a basic set of non-invasive parameters and input for invasive pressure or temperature. Optional Multi-Measurement Server extensions offer Mainstream or Microstream® CO_2.
>
> Whether at the bedside or in transport, M3 and M4 monitors provide seamless connectivity to the patient monitoring network and facilitate information management with central monitoring, bed-to-bed overview, and integration with clinical information systems. Universal admitting, discharge, and transfer at the bedside or central station allows for smooth transitions when admitting, discharging, or transferring patients.
>
> In transit, the monitor uses battery support and a built-in antenna and transceiver to provide continuous monitoring. The small, lightweight Multi-Measurement Server enables quick and convenient transfer of vital signs and trended data to another M3, M4, or IntelliVue patient monitor.

Eigen ervaring

Wat precies het verschil is tussen een M3 en M4 is niet echt duidelijk maar verder is het een redelijk goed te gebruiken apparaat. Door middel van een soort touchscreen is de interface te bedienen. De manual komt echter niet helemaal overheen met de menu items in het apparaat. De M4 die ik onderzocht heb had een software revisie E terwijl de manual die bijgeleverd was alleen voor software revisie D was. Ook de digitale handleiding op een bijgeleverde CD in de directory 'Software revision E' vermelde dat deze alleen bedoelt was voor gebruik met software versie D.

De RJ-45 aansluiting op de achterkant kan volledig benut worden. Er is echter geen configuratie mogelijkheid om een IP adres in te voeren. Dit moet gebeuren via een BOOTP (http://www.networksorcery.com/enp/protocol/bootp.htm) server ergens op het netwerk te draaien. Op de M4 draait dan zelf een soort van firewall. Alle tcp/ip poorten worden gefilterd waardoor het niet duidelijk is hoe er gecommuniceerd moet worden.

Mijn idee is dat dit komt doordat de configuratie ingesteld staat op 'lokaal gebruik' en geen externe verbindingen accepteert. Ik heb hierover een vraag gesteld aan de technische dienst van het AMC die daar tot op heden niet op gereageerd hebben.

Naast de RJ-45 poort zit er ook een VGA aansluiting op. Hier kan een externe monitor op worden aangesloten waarop hetzelfde beeld wordt getoond als op de M4 zelf. Dit beeld heeft een resolutie van 640x480 pixels.

Communicatie met leverancier

Omdat het eerst helemaal niet lukte om de M4 op het netwerk aan te sluiten heb ik telefonisch contact opgenomen met Jan Hendriks van het AMC Instrumenteel Bedrijf. Volgens hem gebruikte de M4 geen tcp/ip maar Philips SDN / Carenet en dat hij daar verder weinig over kon zeggen. Naar aanleiding van dit gesprek stuurde hij het volgende mailtje.

Matthijs,

Ik heb ook nog even gekeken, maar wij weten niet zo een twee drie een project waarbij data via het Philips SDN / CareNet naar een PC wordt gestuurd.
Misschien dat Philips je daar wat verder mee kan helpen. Ad Raemaekers (vertegenwoordiger) of Bas van Breugel (technische helpdesk O40-2787630) kunnen je misschien verder helpen.

Jan

Als reactie hierop heb ik de technische helpdesk van Philips gebeld die mij doorverbonden met een andere technische specialist omdat Bas van Breugel op vakantie was. Ik heb mijn probleem hier verteld. Daar kon die persoon geen direct antwoord op geven maar hij beloofde me bij de fabriek informatie op te gaan vragen en me hierover later terug te bellen.
Het ging hier om het specifieke probleem dat de M4 continue de melding 'not supported network' gaf zodra er een netwerkstekker ingestoken werd.

Drie dagen later werd ik terug gebeld met de mededeling dat er gebruikt gemaakt moest worden van een BOOTP server. Verder was dat de enige informatie hij zelf los kon krijgen dus als ik verder vragen had dan kon hij me daar niet mee helpen. Er wordt dus geen gebruik gemaakt van Philips SDN / Carenet.

Uitkomsten
Zonder protocolgegevens is het erg moeilijk om voor dit product een implementatie te maken. Dit zal moeten door het protocol te *reverse engineren* wat erg veel tijd gaat kosten. Dit moet gedaan worden met hulp van de technische dienst van het AMC die niet al te behulpvaardig is.

Het is in theorie mogelijk om een koppeling te maken.

5.2 Hamilton beademingsapparaat

Product
Adaptive Support Ventilation, Raphael Color (firma Hamilton Medical)
serienummer
CH7403

Soorten communicatie
- RS232
- Speciale aansluiting voor zusterpost en andere alarmen (maak/breek)

Productbeschrijving

http://www.hamilton-medical.com/products/raphael_color.html

> The RAPHAEL Color is the newest addition to HAMILTON MEDICAL's RAPHAEL ventilator family. With its color screen, the fully outfitted RAPHAEL Color is a cost-effective and compact solution for ICU patients from children through adults.
>
> The RAPHAEL Color's novel pneumatic design adapts itself to the patient's needs: the controller delivers the selected tidal volume at the lowest pressure possible, combining the benefits of pressure-controlled ventilation with a volume guarantee. Incorporating a biphasic ventilation concept, the RAPHAEL Color lets your patients breathe freely in all modes and phases.
>
> In addition to the conventional pressure and volume-controlled ventilation modes, the RAPHAEL Color offers the unique closed-loop mode, ASV (Adaptive Support Ventilation), as well as DuoPAP and APRV.
>
> With its comprehensive monitoring and graphics package, the RAPHAEL Color can display parameters as curves, trends, and loops. These are shown to their best advantage by the full-color screen.
>
> Despite the RAPHAEL Color's broad range of ventilation capabilities, operators find the ventilator, with its ergonomic user interface, easy to use.
>
> And although it's compact, the RAPHAEL Color is complete, with battery backup, nebulizer, Interface and oxygen monitoring. The small package fits into a tight corner of the ICU -- and into a small budget, too.

Eigen ervaring

De Raphael Color zit redelijk goed in elkaar. Er zit een duidelijke interface in die door vrijwel iedereen wel te gebruiken is. Aan de achterkant zit een RS232 aansluiting waarmee gecommuniceerd kan worden. Hier wordt een hand-shake methode voor gebruikt. Er is een applicatie van Hamilton zelf (Hamilton Datalogger) die informatie uit het apparaat kan halen, dat is mij zelf nog niet gelukt. Uit de handleiding blijkt dat de protocolbeschrijvingen opgevraagd kunnen worden bij Hamilton zelf.

Communicatie met leverancier

Ik heb via een contactpersoon (Jan Hendriks) binnen het AMC een E-mail gestuurd naar een vertegenwoordiger van Hamilton Medical waarin ik gevraagd heb naar de protocolbeschrijving. Hier heb ik tot heden geen antwoord op gekregen.

Email naar Hamilton Nederland

Geachte heer, mevrouw

Binnen het AMC hebben wij een "Raphael color" beademingsmachine staan waar we wat meer mee willen gaan doen. We zijn bezig met intern een simpele applicatie te ontwikkelen die de data uit de "Raphael color" opvangt en opslaat. Ongeveer vergelijkbaar met de datalogger die Hamilton Medical al levert, maar dan helemaal aan onze wensen aangepast

Voor de maken van deze applicatie hebben we de protocolgegevens van de Raphael color nodig. In de handleiding op pagina G-11 staat dat we daar voor terecht kunnen bij onze Hamilton Medical contactpersoon, in dit geval u.

Ik zou graag het volgende van u willen ontvangen
- Protocol gegevens voor de seriële communicatie tussen de Raphael Color en een PC in een digitaal formaat (pdf, doc, html)

- Een lijstje met mogelijke verschillen in protocol tussen
de Raphael color en andere soortgelijke producten die jullie verkopen

Ik hoop spoedig van u een reactie te krijgen

Met vriendelijke groet,
Matthijs van Dorp

Uitkomsten

Zonder protocolgegevens is het erg moeilijk om voor dit product een implementatie te maken. Dit zal moeten door het protocol te *reverse engineren* wat erg veel tijd gaat kosten. Dit wordt wel iets makkelijker omdat de er een bestaande implementatie is (Hamilton Datalogger) zodat het seriële verkeer gewoon gesnift kan worden.

Het is in theorie mogelijk om een koppeling te maken.