UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Bojana Lango

Razvoj grafičnega uporabniškega vmesnika za spremljanje elektrokardiografskih signalov v realnem času

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

Mentor: prof. dr. Franc Jager

Ljubljana, 2011

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00173/2011
Datum: 03.10.2011

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **BOJANA LANGO**

Naslov: **RAZVOJ GRAFIČNEGA UPORABNIŠKEGA VMESNIKA ZA SPREMLJANJE ELEKTROKARDIOGRAFSKIH SIGNALOV V REALNEM ČASU**

DEVELOPMENT OF GRAPHIC USER INTERFACE FOR MONITORING ELECTROCARDIOGRAPHIC SIGNALS IN REAL TIME

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Z uporabo spiralnega modela načrtovanja uporabniških vmesnikov načrtujte, implementirajte in vrednotite grafični uporabniški vmesnik, ki omogoča izrisovanje elektrokardiografskih krivulj bolnikov v kliničnem okolju in v realnem času ter omogoča ažurno javljanje alarmov v primeru kritičnih stanj miokadra. Vmesnik naj omogoča tudi pregledovanje preteklih alarmov in dodajanje, ažuriranje ter brisanje bolnikov v podatkovni bazi bolnikov. Ob načrtovanju upoštevajte splošno sprejete principe in navodila načrtovanja uporabniških vmesnikov. Vsako itercijo načrtovanja hevristično vrednotite, nato pa izvršite testiranje uporabnikov v dejanskem kliničnem okolju. Za razvoj grafičnega uporabniškega vmesnika uporabite programsko okolje Qt 3.x, razvijte in testirajte pa ga v grafičnem okolju KDE operacijskega sistema LINUX.

Mentor:

prof. dr. Franc Jager

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

I Z J A V A O A V T O R S T V U

diplomskega dela

Spodaj podpisana Bojana Lango z vpisno številko 63010168 sem avtorica diplomskega dela z naslovom:

Razvoj grafičnega uporabniškega vmesnika za spremljanje elektrokardiografskih signalov v realnem času

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelala samostojno pod mentorstvom prof. dr. Franca Jagra
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne Podpis avtorice:

Zahvala

Za vso pomoč in strokovno vodstvo se zahvaljujem svojemu mentorju prof. dr. Francu Jagru.

Sodelavcem nekdanjega podjetja Konel se zahvaljujem za potrpežljivost in vse tehnično znanje, ki so ga nesebično delili z mano. Podjetju Telsima d.o.o. pa iskrena hvala za finančno podporo ob zaključku študija.

Posebna zahvala gre moji družini za vso spodbudo in navdušenje.

Damjanu

Kazalo

1 POVZETEK 1

2 ABSTRACT 3

3 NAMEN DELA 5

3.1 ZGRADBA DIPLOMSKEGA DELA 5

4 FIZIOLOŠKO OZADJE 7

4.1 ZGRADBA IN DELOVANJE SRCA 7

4.1.1 Zgradba srca 7

4.1.2 Delovanje srca 8

4.1.3 Prevodni sistem srca 8

4.2 ELEKTROKARDIOGRAM 9

4.2.1 Kaj je elektrokardiogram 9

4.2.2 Značilnosti krivulje EKG 10

5 ORODJA IN KNJIŽNICE 13

5.1 OKOLJE QT IN TIPIČNE PODOBE 13

5.1.1 Qt Designer 13

5.1.2 QT Assistant 15

5.1.3 Osnovne podobe in gradniki okolja Qt 16

5.1.4 Orodje qmake 18

5.2 KNJIŽNICA STANDARD TEMPLATE LIBRARY 19

5.3 PAMETNI KAZALNIKI 20

6 NAČRTOVANJE GRAFIČNIH VMESNIKOV 23

6.1 PRINCIPI NAČRTOVANJA UPORABNIŠKIH VMESNIKOV 24

6.2 NAVODILA 26

7 OPIS OKOLJA 29

7.1 SIMULATOR MERILNIKA 30

7.2 DETEKCIJA KRITIČNIH STANJ IN PROŽENJE ALARMOV 31

8 REZULTATI DELA 35

8.1 PREDSTAVITEV GRAFIČNEGA VMESNIKA 35

8.2 BOLNIKI 36

8.3 MERILNIKI 40

8.4 AKTIVNE MERITVE 41

8.4.1 Podoba za izris krivulje 42

8.4.2 Podoba za prikaz alarmov *42*
8.4.3 Podoba z osnovnimi podatki o bolniku *44*
8.4.4 Aktivna meritev za posameznega bolnika *44*
8.4.5 Aktivne meritve za več bolnikov hkrati *46*
8.5 ZGODOVINA ALARMOV 47
8.6 ZGODOVINA MERITEV 49
8.7 VREDNOTENJE REZULTATOV DELA 52
8.7.1 Hevristično vrednotenje *52*
8.7.2 Testiranje uporabnikov *53*

9 SKLEPNE UGOTOVITVE 55

10 SMERNICE ZA NADALJNO DELO 57

10.1 AVTENTIKACIJA IN AVTORIZACIJA 57
10.2 VMESNIK ZA PAMETNE TELEFONE 57
10.3 ARHIV BOLNIKOV 57
10.4 BRANJE KARTICE ZDRAVSTVENEGA ZAVAROVANJA 57
10.5 RAZŠIRJENI FILTRI ZA IZPIS ZGODOVINE ALARMOV 58

11 LITERATURA 59

Kazalo slik

SLIKA 1: VZDOLŽNI PREREZ SRCA [11] 7
SLIKA 2: CELICA SRCA OB POLARIZACIJI IN DEPOLARIZACIJI [7] 9
SLIKA 3: PRIMER KRIVULJE EKG Z OZNAČENIMI SEGMENTI [13] 10
SLIKA 4: QT DESIGNER 3 13
SLIKA 5: OKNO ZA UREJANJE LASTNOSTI PODOB 14
SLIKA 6: RAZISKOVALEC PODOB 15
SLIKA 7: QT ASSISTANT 16
SLIKA 8: OSNOVNO OKNO 17
SLIKA 9: PODOBA Z ZAVIHKI 17
SLIKA 10: TABELA 18
SLIKA 11: RAČUNALNIŠKA SKICA OKNA 23
SLIKA 12: DIAGRAM SISTEMA ZA MERJENJE EKG 29
SLIKA 13: OSNOVNO OKNO GRAFIČNEGA VMESNIKA 35
SLIKA 14: SEZNAM BOLNIKOV 37
SLIKA 15: UREJANJE PODATKOV O BOLNIKU 39
SLIKA 16: SEZNAM MERILNIKOV 41
SLIKA 17: PODOBA ZA IZRIS KRIVULJE 42
SLIKA 18: PODOBA ZA PRIKAZ ALARMOV V ČASU, KO NI NOBENEGA ALARMA 43
SLIKA 19: PODOBA ZA PRIKAZ ALARMOV V ČASU AKTIVNEGA ALARMA SREDNJE PRIORITETE 43
SLIKA 20: PODOBA Z OSNOVNIMI PODATKI O BOLNIKU V ČASU AKTIVNE MERITVE 44
SLIKA 21: AKTIVNA MERITEV ZA ENEGA BOLNIKA 45
SLIKA 22: PRIKAZ AKTIVNIH MERITEV ZA VEČ BOLNIKOV HKRATI 46
SLIKA 23: ZGODOVINA ALARMOV 47
SLIKA 24: IZRIS PRETEKLE MERITVE 49
SLIKA 25: DIALOG ZA IZBIRO KRIVULJ ZA PRIKAZ PRETEKLE MERITVE 50
SLIKA 26: DIALOG ZA IZBIRO PRETEKLE MERITVE 51

1 Povzetek

Tema diplomskega dela je razvoj grafičnega uporabniškega vmesnika za spremljanje elektrokardiografskih signalov v realnem času. Naša naloga je bilo načrtovanje, implementacija in vrednotenje grafičnega vmesnika.

Razvili smo grafični uporabniški vmesnik, ki omogoča pregledovanje, dodajanje, ažuriranje in brisanje bolnikov v bazo bolnikov ter dodeljevanje merilnikov bolnikom. Omogoča izrisovanje elektrokardiografskih krivulj v realnem času in v primeru kritičnih stanj ažurno javljanje alarmov. Za čim boljšo opaznost smo pri prikazu alarmov uporabili izrazite žive barve in utripajoče ozadje. Zaključene meritve so dosegljive za pregledovanje in tiskanje. Hrani zgodovino poteklih alarmov in omogoča njihovo pregledovanje.

Razvoj je potekal v več iteracijah spiralnega modela načrtovanja uporabniških vmesnikov. Pri svojem delu smo se opirali na principe in navodila za razvoj uporabniških vmesnikov, zlasti na 10 Nielsenovih principov načrtovanja uporabniških vmesnikov. Zaključeni sklop funkcionalnosti vsake iteracije smo vrednotili hevristično v povezavi s principi načrtovanja grafičnih vmesnikov. Na izdelanem grafičnem uporabniškem vmesniku je potekalo testiranje uporabnikov v dejanskem kliničnem okolju v Splošni bolnišnici Dr. Franca Derganca Nova Gorica. Grafični uporabniški vmesnik smo izpopolnjevali v sodelovanju z bolnišničnim osebjem.

Grafični uporabniški vmesnik smo razvili v programskem okolju Qt 3.x. Grafični uporabniški vmesnik ni odvisen od operacijskega sistema, razvijali in testirali pa smo ga v grafičnem okolju KDE operacijskega sistema LINUX.

Ključne besede:

grafični uporabniški vmesnik, elektrokardiografija, alarmi, načrtovanje, principi in navodila

2 Abstract

The topic of this thesis is development of a graphical user interface for real-time monitoring of electrocardiographic signals. Our main goal was to design, implement and evaluate the graphical user interface.

We developed a graphical user interface which allows us to browse, add, edit and delete patients in the patient's database. It also allows us to assign the measurement devices to the patients. It supports real-time displaying of electrocardiographic curves and alarm notifications in case of patient's critical states. For best visibility we used bright vivid colors and flashing background. Past measurements can be viewed and printed. Past alarms are stored and accessible for examination.

Development took place in several iterations of the spiral user interface design model. In our work we followed basic principles and guidelines for user interface design, especially 10 Nielsen's design principles. Subset of functionalities for each iteration has been validated against usability heuristics. The developed graphical user interface was submitted to user testing in a real clinical environment in Dr. Franc Derganec general hospital in Nova Gorica. We updated the graphical user interface in collaboration with the medical staff.

The graphical user interface was developed in Qt 3.x programming environment. It runs on several platforms and operating systems and was developed and tested in KDE graphical environment of LINUX OS.

Key words:

Graphical user interface, electrocardiography, alarms, design, principles and guidelines

3 Namen dela

Pri motnjah in nepravilnostih v delovanju srca je pravočasno ukrepanje pogosto ključnega pomena. Zaradi tega je smiselna želja po konstantnem nadzoru nad srčno aktivnostjo bolnikov in prikazom stanja v realnem času.

Sistem, ki bi omogočal pregled nad več bolniki hkrati in izpisoval rezultate meritev EKG v realnem času, bi predstavljal dragoceno orodje za osebje v bolnišnici.

Odločili smo se razviti grafični vmesnik, ki bi znal v realnem času izrisovati krivuljo EKG in tudi delno analizirati izmerjene vrednosti. Prepoznati mora nepravilnosti in jih klasificirati ter javljati alarme. Ker niso vsa odstopanja enako ključnega pomena za življenje bolnika, mora vmesnik znati prepoznati obliko nepravilnosti in tudi s stopnjo alarma opozoriti, kako nujno je ukrepanje. Glede na medicinsko zgodovino bolnika mora obstajati možnost, da se mejne vrednosti za proženje alarmov postavijo za vsakega bolnika posebej preko enostavne konfiguracijske datoteke.

Zaradi želje po prenosljivosti in odzivnosti smo kot najprimernejše okolje za razvoj izbrali QT/C++. Okolje QT deluje na večini najbolj razširjenih operacijskih sistemov, kot so Windows, Linux, Unix, MacOS. Z vidika prenosljivosti bi bila primerna izbira tudi programski jezik Java, vendar se javanska koda interpretira in izvaja bistveno počasneje v primerjavi s QT/C++. [2]

Želeli smo razviti uporaben in estetski grafični uporabniški vmesnik, zato smo pri svojem delu upoštevali mentalni model uporabnika. Skušali smo se držati navodil za načrtovanje in gradnjo grafičnih vmesnikov. Končni izdelek smo želeli hevristično oceniti s pomočjo Mandelovih in Nielsenovih principov načrtovanja.

Seveda bi bilo nesmiselno pričakovati, da bi računalniško podprt sistem nadomestil znanje ekspertov kardiologov. To nikakor ni namen te diplomske naloge. Želeli pa smo razviti orodje, ki bi lajšalo njihovo delo s sprotnim pregledom nad trenutnim stanjem bolnikov in opozarjalo, kadar se njihova srčna aktivnost spremeni.

3.1 Zgradba diplomskega dela

Diplomska naloga se začne s kratko predstavitvijo zgradbe in delovanja srca. Razloženi so osnovni pojmi elektrokardiografije in osnovne značilnosti EKG krivulje. Opis fiziološkega ozadja pomaga k boljšemu razumevanju problemske domene.

Sledi predstavitev orodij in knjižnic, ki smo jih pri svojem delu uporabljali. Veliko pozornosti smo namenili okolju QT, orodju QT Designer in tipičnim podobam okolja QT, ki smo jih uporabili za razvoj grafičnega vmesnika.

Kot uvod v načrtovanje grafičnega vmesnika je razložen spiralni model načrtovanja programske opreme. Sledijo našteti nekateri principi in navodila za načrtovanje uporabniških vmesnikov.

Glavni del diplomske naloge se začne z grobim opisom okolja. Predstavljen je simulator merilnika, ki smo ga uporabljali pri svojem delu, kadar pravih merilnikov ni bilo na voljo. Razložena je arhitektura celotnega sistema, katerega del je naš grafični vmesnik. Opisano je, kako poteka detekcija kritičnih stanj in javljanje alarmov.

Jedro diplomske naloge je predstavitev izdelanega grafičnega vmesnika. Vsebuje natančen opis izgleda in funkcionalnosti vseh elementov grafičnega vmesnika. Poleg tega pojasnjuje, katere principe smo pri načrtovanju upoštevali. Za posamezne dele grafičnega vmesnika utemeljimo, katera navodila načrtovanja so upoštevana in zakaj. Sledi primer hevurističnega vrednotenja rešitve in opis testiranja s strani testnih uporabnikov.

Sklepni del obsega kritično oceno in vrednotenje opravljenega dela ter smernice za nadaljno delo. Ob samem razvoju so se porodile različne ideje, kako bi trenutno različico grafičnega vmesnika lahko izboljšali in dopolnili za bolj učinkovito uporabo. Te so pojasnjene v zaključnem poglavju s smernicami za nadaljno delo.

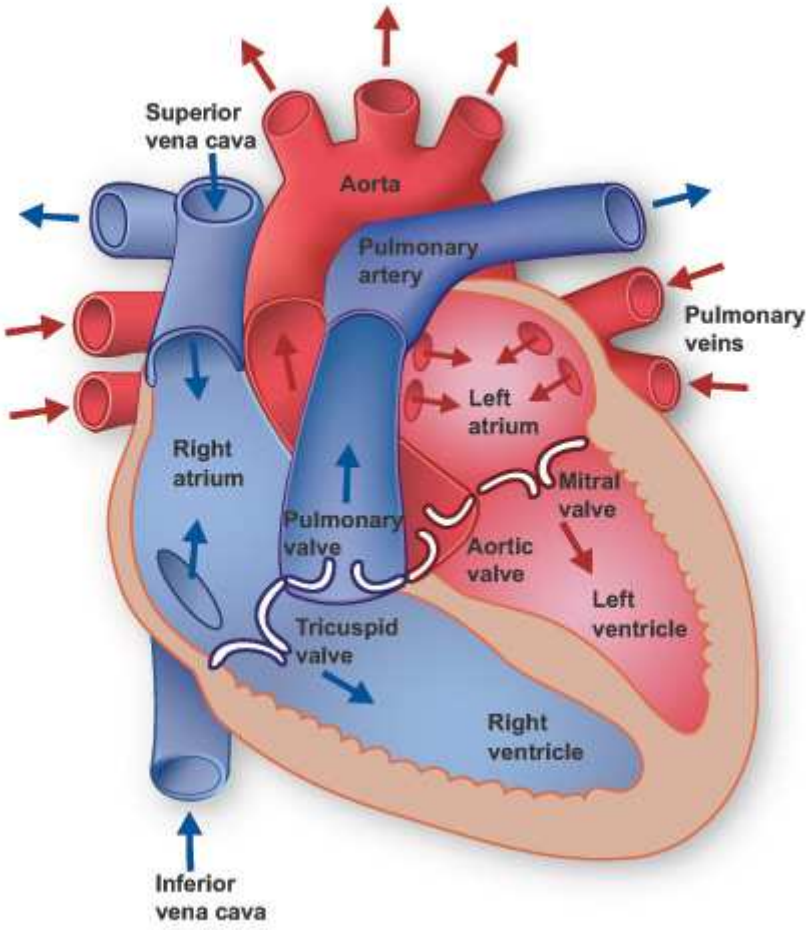
4 Fiziološko ozadje

4.1 Zgradba in delovanje srca

4.1.1 Zgradba srca

Srce leži v sredini prsnega koša, nekoliko levo med levim in desnim pljučnim krilom. Je votel organ. Stena srca je sestavljena iz treh plasti [1]:

- posrčnica (endokard) je najbolj notranja plast, ki pokriva nortanjost srčnih votlin in zaklopke;
- srčna mišica (miokard);
- osrčnik (perikard) je trdna vezivna plast, ki obdaja celotno srce.



Slika 1: Vzdolžni prerez srca [11]

Srčna mišica je po zgradbi podobna progastim skeletnim mišicam, na primer bicepsu. Po samem delovanju pa je bolj podobna gladkim mišicam notranjih organov, saj njeno delovanje ni odvisno od naše volje. Slika 1 prikazuje zgradbo srca.

Notranjost srca delimo na levo in desno polovico, ki sta med sabo ločeni z močno mišično steno (pretin ali septum). Vsaka od polovic je sestavljena iz dveh delov, ki sta ločena z zaklopko. Zgornja votlina se imenuje preddvor (atrij), spodnja votlina pa prekat (ventrikel). Srce je torej sestavljeno iz dveh preddvorov, v katera doteka kri preko telesnih in pljučnih dovodnic (ven), ter dvek prekatov, iz katerih kri zapušča srce v pljučno arterijo in veliko telesno odvodnico (aorto). V srcu se nahajajo štiri srčne zaklopke: po ena med vsakim prekatom in preddvorom in še dve, ki ločujeta prekata od velikih žil. Delujejo kot nekakšni ventili, ki se odpirajo in zapirajo. Njihova naloga je usmerjanje toka krvi skozi srce.

4.1.2 Delovanje srca

Srce deluje kot usklajena dvojna črpalka. V desni preddvor priteka s kisikom revna kri, se potisne v desni prekat, od tam pa gre v pljuča. S kisikom nasičena kri se vrne v srce skozi pljučne vene v levi preddvor. Od tam teče v levi prekat, ki jo preko velike telesne odvodnice potiska po arterijah celotnega telesa. Zaradi tega je mičica levega prekata najdebeljša in najmočnejša.

V grobem lahko razdelimo delovanje srca na dve stopnji: raztezanje (diastola), ko se srce polni s krvjo, in krčenje (sistola), ko se prekata skrčita in potisneta kri v žile. [1][4]

4.1.3 Prevodni sistem srca

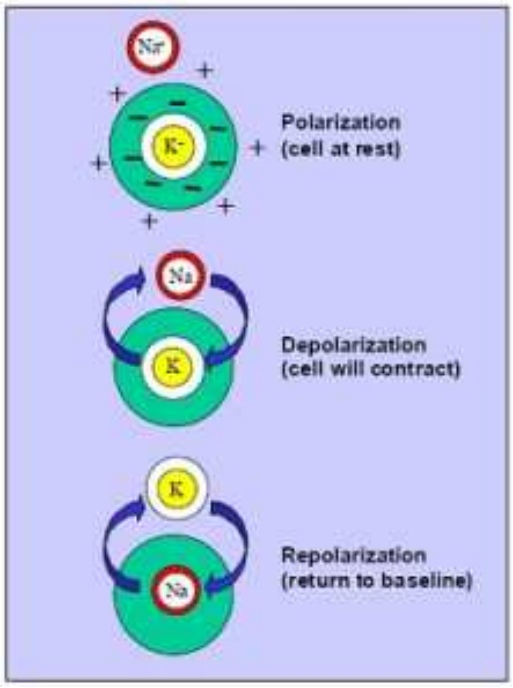
Za učinkovito delovanje srca se morajo njegovi posamezni deli krčiti v natančno določenem zaporedju. V samem srcu se nahaja specifični živčno-mišični sistem, ki je sposoben tvoriti dražljaje, ki povzročajo krčenje srčne mišice.

Tik pod izlivom zgornje velike vene se nahaja t.i. sinusni vozel, ki sproža krčenje preddvorov in s tem iztisk krvi v prekata. Vzburjenje se zakasnjeno preko preddvornoprekatnega vozla (AV vozla), Hisovega snopa in Purkinijevih niti, ki so vpletene v mišice prekatov, prenese v prekata, ki se skrčita nekoliko kasneje kot preddvora.

V mirovanju je ovojnica srčne celice električno negativna. Če namestimo eno elektrodo na površino celice miokarda, drugo pa nekoliko stran, se zaradi upornosti celične stene izmeri potencial enak 0. Če pa bi z elektrodo predrli celično steno, bi izmerili v notranjosti celice negativen potencial (potencial mirovanja).

Na notranji strani celične membrane je visoka koncentracija K^+ ionov, na zunanji strani pa Na^+ ionov. S signalom, ki povzroči začetek **depolarizacije**, se spremeni prepustnost celične membrane za Na+ ione. Ti hitro vstopajo v notranjost celice, zato celični potencial hitro naraste v pozitivni smeri. Mišičje se skrči. Za tem se potencial celic spet postopno vrača v stanje mirovanja. Govorimo o **repolarizaciji**. Koncentracija K^+ ionov znotraj celice se poveča, medtem ko Na^+ ioni izstopajo iz celice. Mišičje se spet sprošča. [7]

Slika 2 predstavlja poenostavljeno celico v polarizaciji in depolarizaciji.



Slika 2: Celica srca ob polarizaciji in depolarizaciji [7]

4.2 Elektrokardiogram

4.2.1 Kaj je elektrokardiogram

Elektrokardiogram (EKG) je grafični zapis električnih potencialov srčne mišice. Merimo ga z elektrokardiografom. Ta preko eletrod, ki so nameščene na različnih položajih telesa, meri napetost.

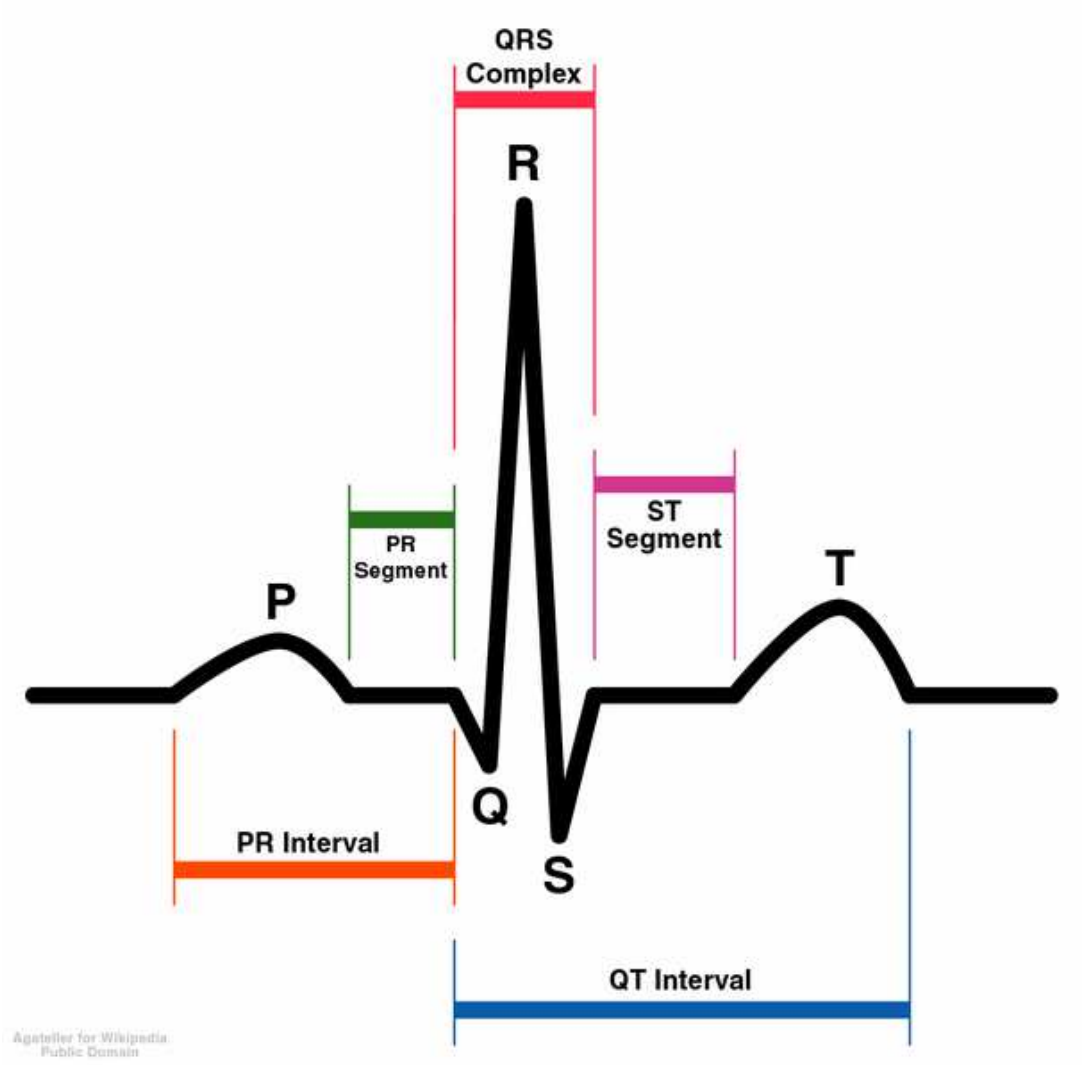
Običajno je na kožo pritrjenih 9 elektrod: na levo in desno zapestje, na levi gleženj ter šest elektrod na sprednjo in levo stran prsnega koša, tako da obkrožajo srce. EKG odda 12 odvodov ali zapisov, ki imajo posebne oznake:

- D1 - desna roka in leva roka
- D2 - desna roka in leva noga
- D3 - leva roka in leva noga
- AVR - desna roka
- AVL - leva roka
- AVF - leva noga
- V1, V2, V3, V4, V5 in V6

Odvodi D1, D2 in D3 do bipolni in merijo diferenco med dvema točkama. Preostali odvodi so enopolni. [7][12]

4.2.2 Značilnosti krivulje EKG

Slika 3 prikazuje primer krivulje EKG:



Slika 3: Primer krivulje EKG z označenimi segmenti [13]

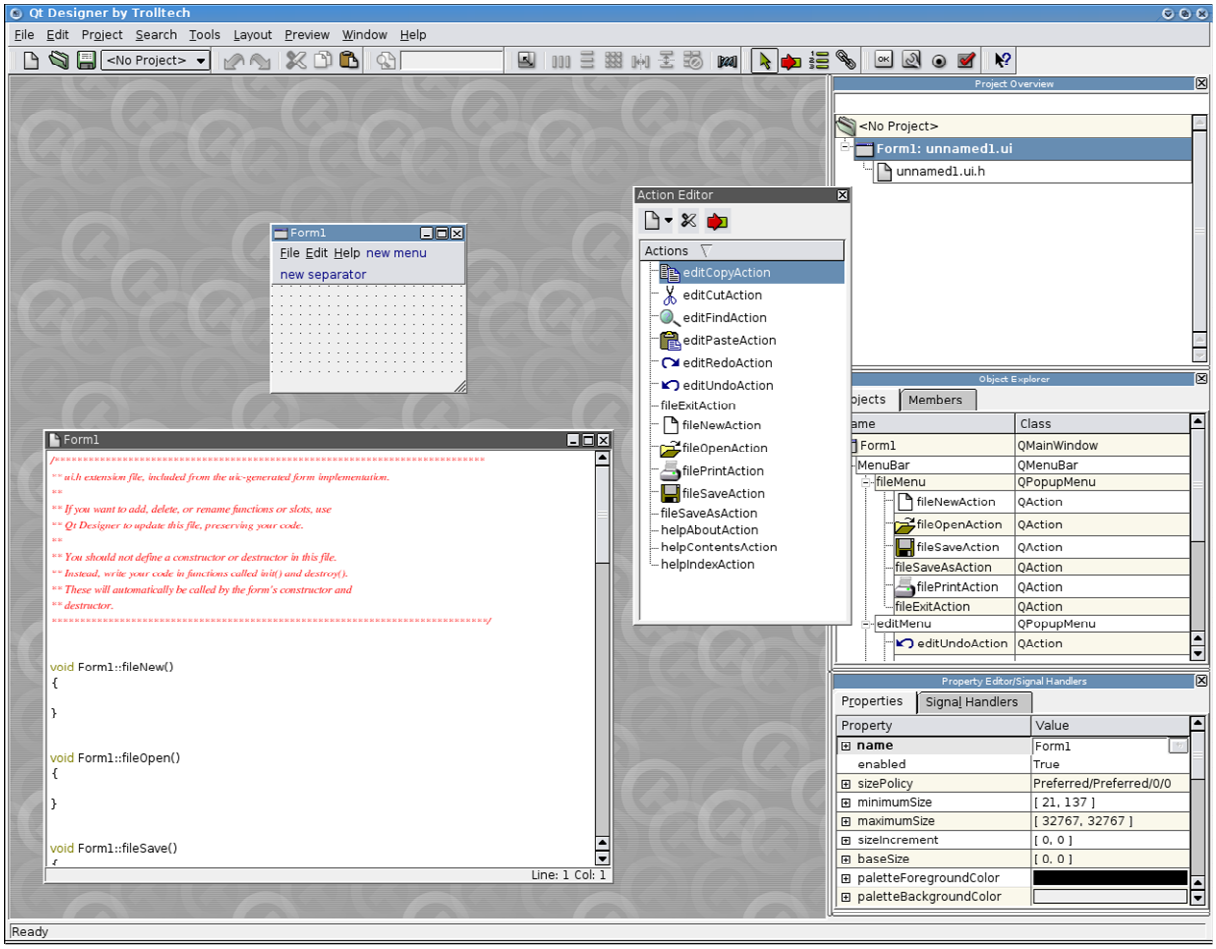
- **P val** je posledica depolarizacije preddvorov.
- **PR segment** povezuje konec P vala in začetek QRS kompeksa in je ponavadi izoelektričen, kar se na grafu kaže kot ravna črta.
- **QRS kompleks** predstavlja celoten čas depolarizacije prekatov.
- **J koleno** je prehod med QRS kompleksom in začetkom **ST segmenta**, ki je ponavadi izoelektričen.
- **T val** prikazuje spremembe ob repolarizaciji prekatov, sledi mu **U val**, ki napoveduje naslednji P val.
- **QT interval** predstavlja čas električne sistole.

5 Orodja in knjižnice

5.1 Okolje Qt in tipične podobe

5.1.1 Qt Designer

Qt je okolje za hitro izdelavo grafičnih vmesnikov v jeziku C++, ki so podprti na več različnih operacijskih sistemih. Je objektno usmerjen. V okolju Qt je na primer razvito grafično okolje KDE operacijskega sistema Linux. Za razvoj grafičnega vmesnika smo uporabljali Qt 3.x. Orodje za izdelavo grafičnih vmesnikov se imenuje Qt Designer (Slika 4).



Slika 4: QT Designer 3

Podprte platforme:

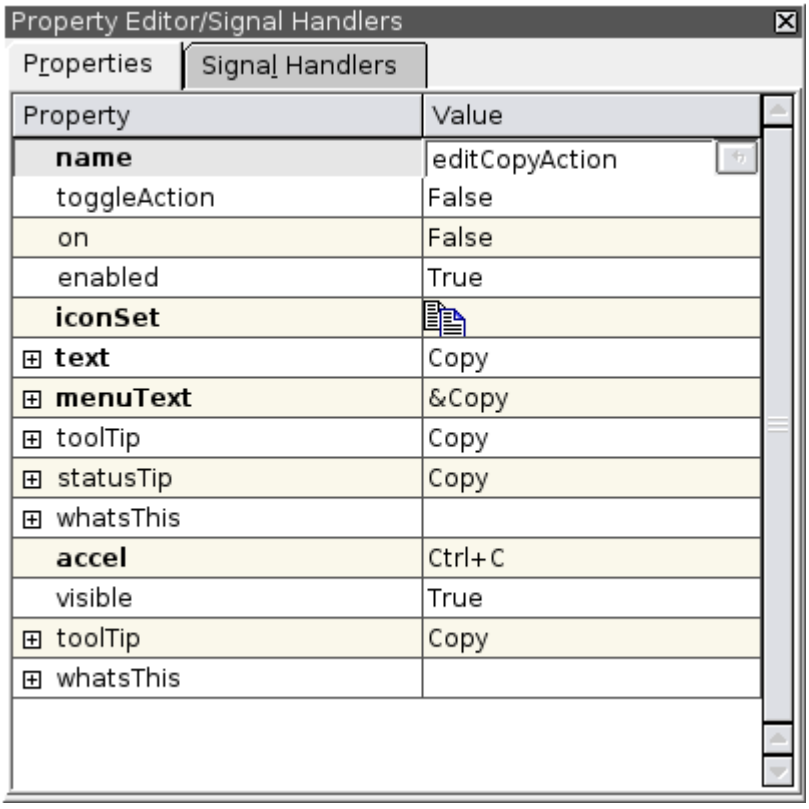
- Microsoft Windows (95, 98, NT 4.0, ME, 2000, XP, ...)

- Unix/X11 (npr. Linux, Sun Solaris, HP-UX, Compaq Tru64 UNIX, IBM AIX, SGI IRIX)
- Macintosh (Mac OS X)
- vgrajeni sistemi (ang. embedded systems)

Orodje Qt Designer smo uporabljali predvsem za vizaulno urejanje posameznih elementov grafičnega vmesnika, izvorno kodo pa smo pisali v preprostem urejevalniku besedila. Urejanje izvorne kode, ki pripada oknom, je sicer možno tudi znotraj orodja QT Designer, vendar dopušča samo kreiranje oken s privzetimi vhodnimi parametri. Če želimo kreirati okno s poljubnimi vhodnimi parametri, v zunanjem urejevalniku besedila napišemo razred, ki deduje našo formo in jo razširi po naših željah.

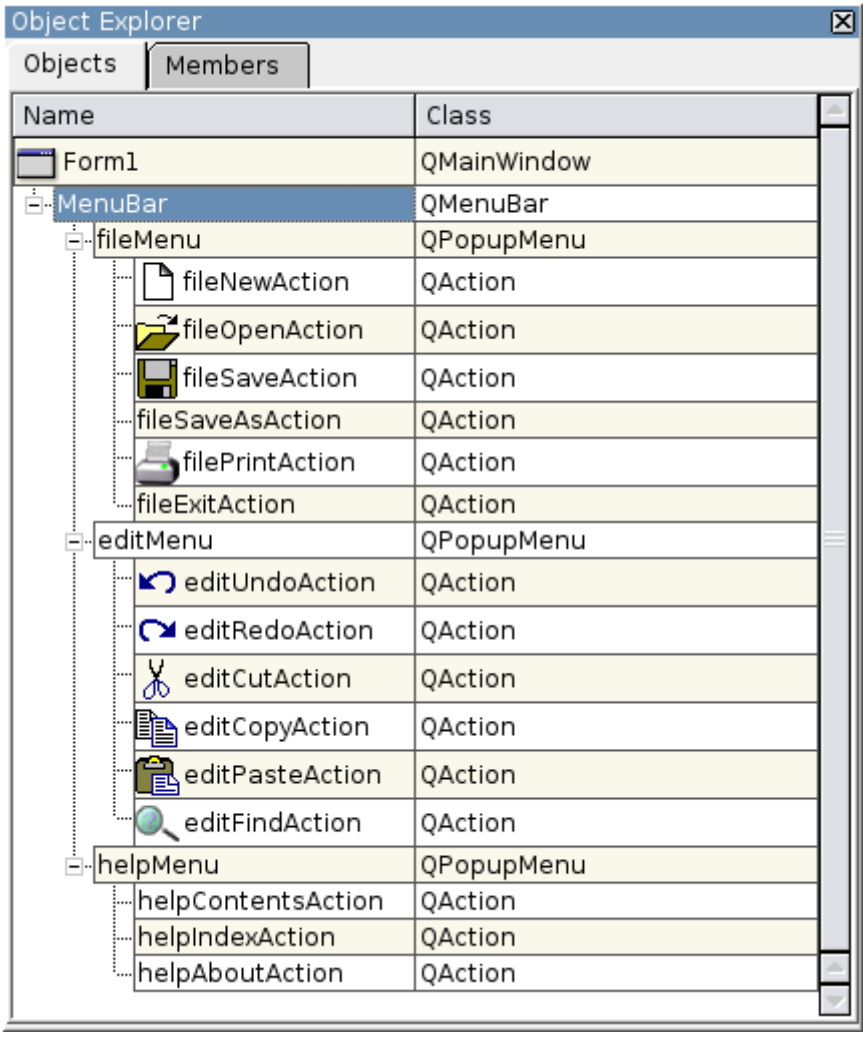
Oglejmo si nekatera izmed oken za urejanje projekta, ki jih ponuja Qt Designer.

Okno za urejanje lastnosti podobe (property editor, Slika 5) nam omogoča editiranje lastnosti vseh objektov našega grafičnega vmesnika.



Slika 5: Okno za urejanje lastnosti podob

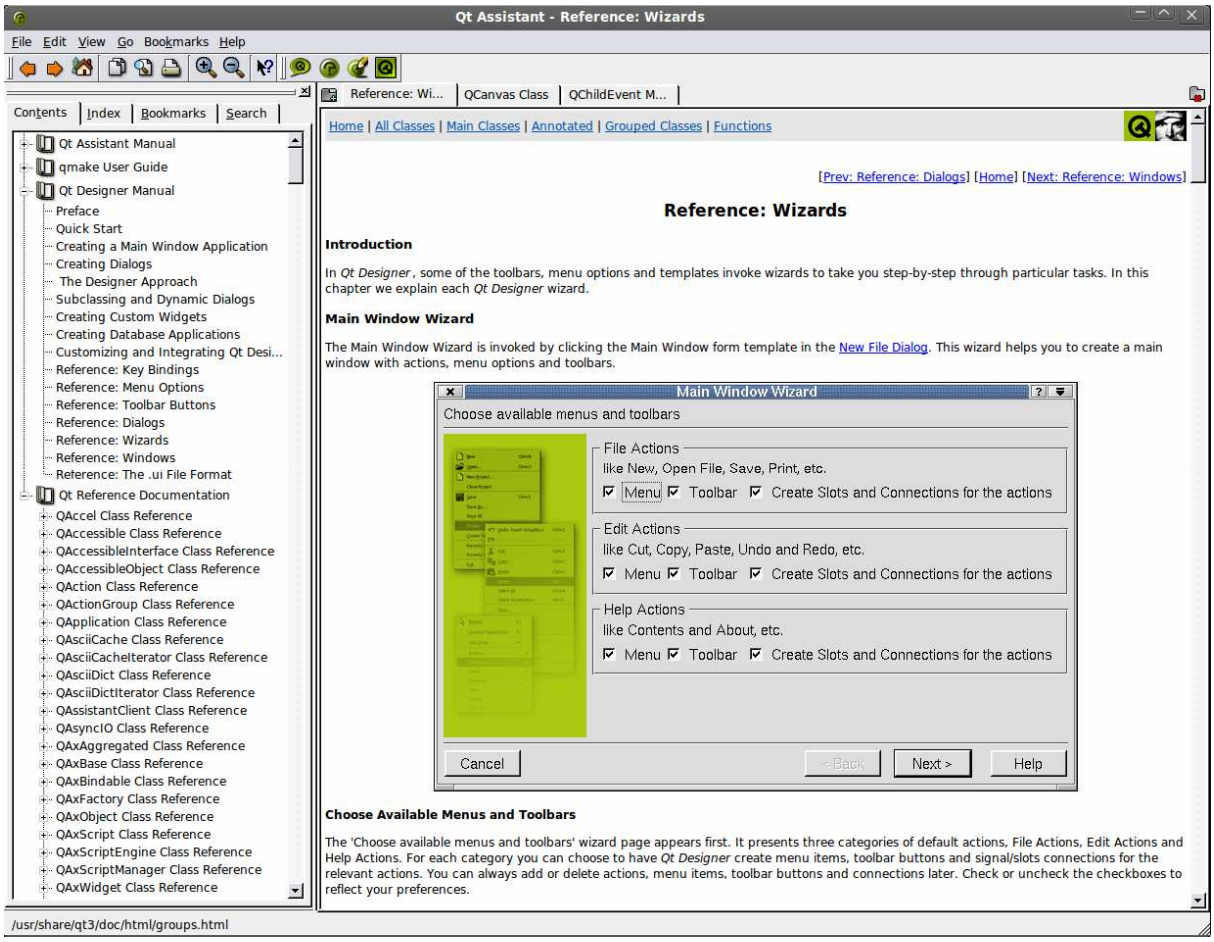
Raziskovalec podob (object explorer) vsebuje vse elemente grafičnega vmesnika. Podobe oz. elementi grafičnega vmesnika v okolju QT so razporejeni hierarhično. Tako so tudi v raziskovalcu podob prikazani hierarhično v obliki drevesne strukture (Slika 6).



Slika 6: Raziskovalec podob

5.1.2 QT Assistant

Okolje QT vsebuje odlično dokumentacijo, ki je enostavno dostopna preko orodja QT Assistant. QT Assistant na pogled in tudi po uporabi spominja na sodobne spletne brskalnike. Dokumenti vsebujejo povezave na druge dokumente. Omogočeno je odpiranje dokumentov v več zavihkih in dodajanje zaznamkov. Orodna vrstica vsebuje gumb za pomikanje naprej in nazaj. Na voljo je iskanje po naslovih in vsebini dokumentacije. Slika 7 prikazuje QT Assistant z odprtimi tremi dokumenti.



Slika 7: QT Assistant

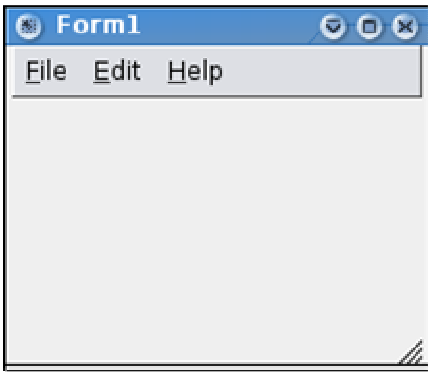
QT Assistant vsebuje uporabniško dokumentacijo za naslednje vsebinske sklope:

- Predstavitev in uporaba orodja QT Assistant
- Vodnik za uporabo orodja qmake
- Uporabniška dokumentacija za orodje QT Designer
- Referenčna dokumentacija za vse razrede okolja QT

5.1.3 Osnovne podobe in gradniki okolja Qt

Oglejmo si nekaj osnovnih podob, ki smo jih uporabili pri izdelavi grafičnega vmesnika.

QMainWindow je osnovno okno in nudi osnovo za aplikacijo. Vsebuje privzeti meni tipa **QMenuBar** z elementi File, Edit in Help, ki ga enostavno razširimo po svojih željah. Slika 8 ja primer osnovnega okna, kot ga kreira QT Designer. Na osnovno okno nanizamo elemente grafičnega vmesnika. Ob kreiranju objekta so avtomatično generirani tudi signali in njihove odzivne funkcije (ang. signals and slots).

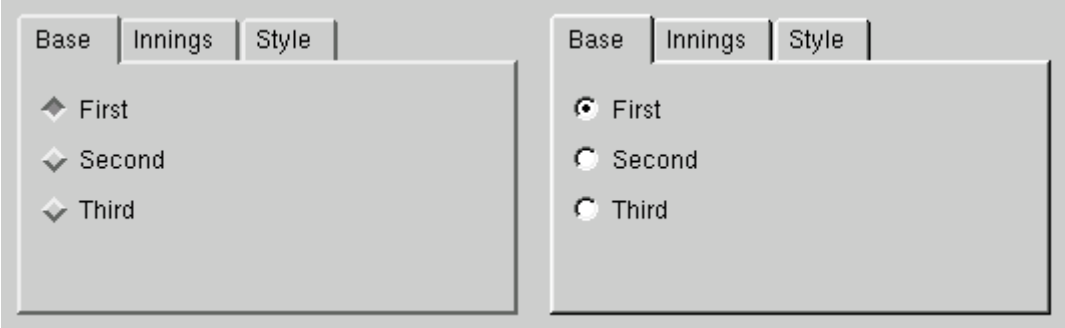


Slika 8: Osnovno okno

Primer odzivne funkcije, kakor jo generira QtDesigner za element osnovnega menuja File -> New:

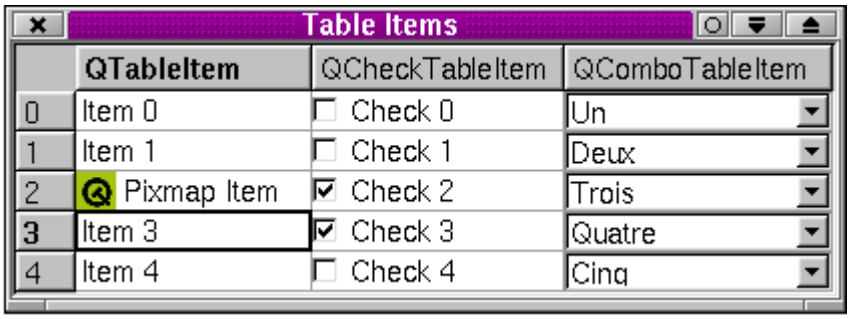
```
void Form1::fileNew()
{

}
```

QTabWidget oz. podoba z zavihki omogoča prikaz več zavihkov (Slika 9). Vsakemu zavihku pripada po ena podoba.



Slika 9: Podoba z zavihki

QTable je tabela, ki vsebuje vrstice in stolpce. Ponuja mnogo možnosti za prilagajanje in razširitev. Celice tabele lahko vsebujejo tekst, slike in druge podobe. Slika 10 prikazuje tabelo, ki vsebuje tekstovna polja, sliko, potrditvena polja in izvlečne sezname. Implementiramo lahko poljubno dinamično sortiranje po stoplcih.



Slika 10: Tabela

QPixmap je platno za rastersko risanje oz. tabela pik. Ta gradnik smo uporabili za izrisovanje krivulje EKG. Postopek risanja :

- Kreiramo objekt tipa QPixmap enake velikosti kot podoba, na katero želimo risati.
- Napolnimo platno z barvo ozadja.
- Narišemo sliko/grafiko na platno.
- Narišemo vsebino platna na podobo s klicom metode bitBlt.

5.1.4 Orodje qmake

Orodje qmake samodejno generira skripto za prevajanje (ang. makefile) za celotno vsebino projekta. Vsebina projekta je definirana v tekstovni datoteki s končnico .pro. Ta se samodejno kreira ob definiranju projekta v orodju QT Designer, vendar v tem primeru vključuje samo razrede, ki so bili kreirani znotraj orodja QT Designer. Nad poljubno izvorno kodo lahko datoteko generiramo sami iz ukazne vrstice s pomočjo uporabe orodja qmake. Lahko pa projektno definicijo napišemo ali dopolnimo sami.

Kot primer si predstavljajmo enostaven program za izpis sporočila »Hello, world«, ki vsebuje naslednje datoteke:

- hello.h
- main.cpp
- hello.cpp

Definicijo projekta generiramo s pomočjo orodja qmake z naslednjim ukazom:

```
$ qmake –project hello.pro
```

Orodje qmake bo v definicijo projekta privzeto vključilo vse datoteke trenutnega direktorija in njegovih poddirektorijev s končnicami *.c; *.ui; *.y; *.l; *.ts; *.h; *.hpp; *.hh; *.H; *.hxx; *.cpp; *.cc; *.cxx; *.C. Dobimo tekstovno datoteko hello.pro z naslednjo vsebino:

```
#############################################################
########
# Automatically generated by qmake (1.07a) Sat Sep 10 16:47:11
2011
#############################################################
########

TEMPLATE = app
INCLUDEPATH += .

# Input
HEADERS += hello.h
SOURCES += hello.cpp main.cpp
```

Ko imamo definicijo projekta, je nadaljni postopek prevajanja zelo enostaven:

```
$ qmake –o Makefile hello.pro
$ make
```

Orodje qmake zelo olajša razvoj programske opreme, saj je ročno pisanje skripte za prevajanje pogosto zamudno in podvrženo napakam. Definicija projekta, ki jo uporablja orodje qmake, je bistveno krajša in bolj razumljiva od skripte za prevajanje, ki jo zahteva orodje make.

5.2 Knjižnica Standard Template Library

Standard Template Library [8] ali krajše STL ponuja mnoge koristne gradnike, ki lajšajo programiranje v jeziku C++, npr. lista, vektor in več različnih tipov iteratorjev. Implementirane ima različne algoritme, npr. sort za urejanje seznamov in find za iskanje po seznamu.

Vektor (ang. vector) je dinamičen linearen seznam. Performanca ukazov nad vektorjem je primerljiva z uporabo običajnega polja, poleg tega pa se velikost vektorja samodejno prilagaja trenutnim zahtevam.

Primer uporabe vektorja:

```
// constructing vectors
#include <iostream>
```

```
#include <vector>
using namespace std;

int main ()
{
  unsigned int i;

  // constructors used in the same order as described above:
  vector<int> first;              // empty vector of ints
  vector<int> second (4,100);     // four ints with value 100
  vector<int> third (second.begin(),second.end());
// iterating through second
  vector<int> fourth (third);     // a copy of third

  // the iterator constructor can also be used to construct
from arrays:
  int myints[] = {16,2,77,29};
  vector<int> fifth (myints, myints + sizeof(myints) /
sizeof(int) );

  cout << "The contents of fifth are:";
  for (i=0; i < fifth.size(); i++)
    cout << " " << fifth[i];

  cout << endl;

  return 0;
}
```

5.3 Pametni kazalniki

Pametni kazalnik (ang. smart pointer) je abstraktni podatkovni tip, ki za uporabo posnema običajne kazalnike, hkrati pa ima lahko implementirane še dodatne funkcionalnosti. Podobnost s kazalniki dosežemo z implementacijo operatorjev * in ->. Inicializacija pametnega kazalnika ni potrebna, ker za to že poskrbimo v konstruktorju. S tem se izognemo uporabi neinicializiranega kazalnika.

Z uporabo pametnih kazalnikov lahko olajšamo upravljanje s pomnilnikom in se izognemo uhajanju pomnilnika (ang. memory leaks). Z implementacijo destruktorja dosežemo, da se ob

uničenju pametnega kazalnika izbriše tudi objekt, na katerega kaže, in tako poskrbimo za sproščanje dodeljenega pomnilnika.

Poleg tega lahko implementiramo tudi različne metode za direkten dostop do operacij nad objektom, na katerega pametni kazalnik kaže.

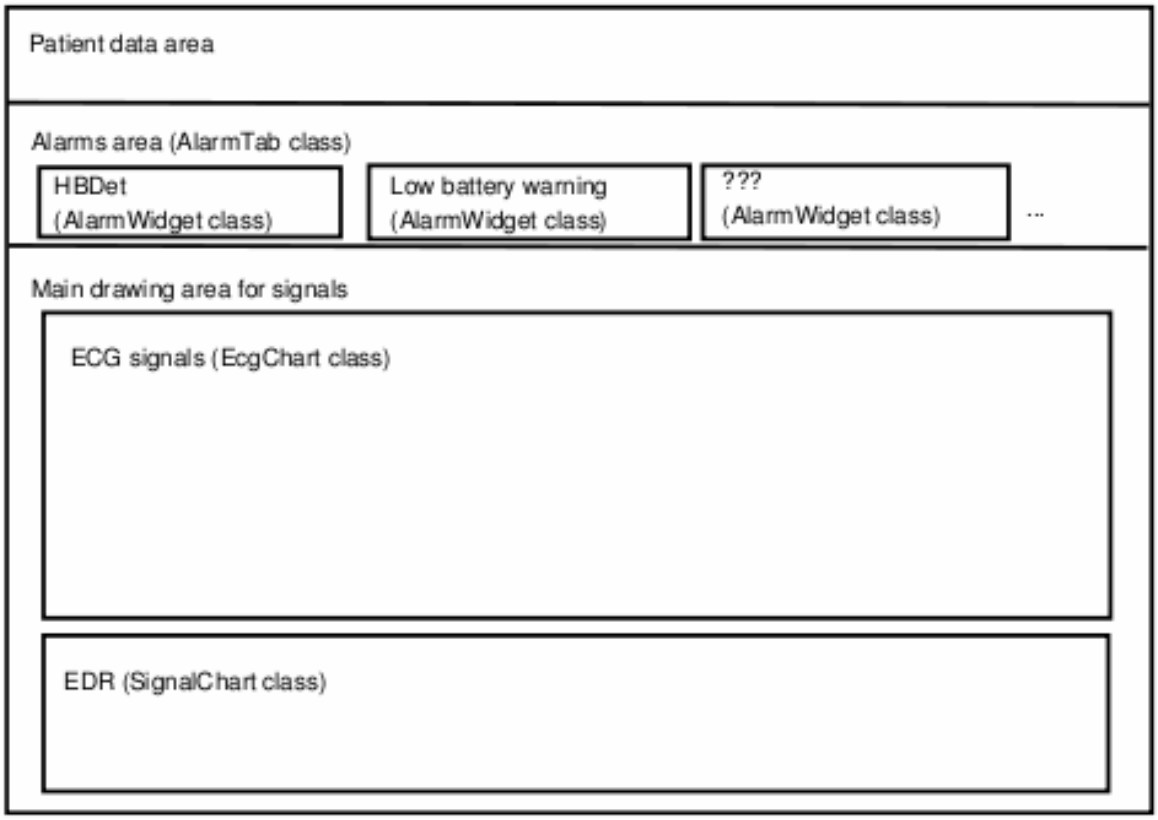
Enostaven primer pametnega kazalnika je razred auto_ptr [14], ki je del standardne knjižnice jezika C++. Oglejmo si del implementacije in enostaven primer uporabe:

```
template <class T> class auto_ptr
{
    T* ptr;
public:
    explicit auto_ptr(T* p = 0) : ptr(p) {}
    ~auto_ptr()                 {delete ptr;}
    T& operator*()              {return *ptr;}
    T* operator->()             {return ptr;}
    // ...
};
// without the use of auto_ptr
void foo()
{
    MyClass* p(new MyClass);
    p->DoSomething();
    delete p;
}

// using auto_ptr
void foo()
{
    auto_ptr<MyClass> p(new MyClass);
    p->DoSomething();
}
```

6 Načrtovanje grafičnih vmesnikov

Pri svojem delu smo uporabljali spiralni del načrtovanja, kjer se izmenjujejo faze Načrtuj, Implementiraj, Vrednoti [3]. Gre za uporabniško usmerjeno načrtovanje, ki se izvaja v več iteracijah.

Najprej smo analizirali naloge, ki jih vmesnik ali posamezno notranje okno mora opravljati. Naloge smo razdelili v jasne in enostavne cilje, določili pa smo jih s pomočjo testnih uporabnikov. Začeli smo z osnovno postavitvijo okna, ki smo jo grobo skicirali na list papirja ali na hitro narisali v enostavnem programu za risanje. Slika 11 je primer takšne skice, ki smo jo narisali ob preoblikovanju že obstoječega okna za prikaz meritev posameznega bolnika.



Slika 11: Računalniška skica okna

Ko smo se razvijalci strinjali glede papirnatega prototipa, smo sestavili komponente okna s pomočjo orodja QT Designer. QT Designer omogoča hitro in enostavno gradnjo uporabniških vmesnikov, zato je zelo primeren za pripravo prototipov. Takšen računalniški prototip že da

vtis končnega izdelka, saj ponuja realno uporabniško izkušnjo oz. »look&feel«. Ta prototip smo pokazali testnim uporabnikom in jih vprašali za mnenje.

Ko je bil prototip prilagojen zahtevam uporabnikom, smo se lotili implementacije. Ker je bil prototip zgrajen v okolju QT, smo ga pogosto lahko vsaj delno uporabili tudi pri dejanski implementaciji našega uporabniškega vmesnika.

Hevristično vrednotenje uporabnosti je metoda preiskovanja in ocenjevanja uporabniških vmesnikov. Ocenjevalci smo prototipe in tudi že implementirani uporabniški vmesnik primerjali s principi načrtovanja. Na takšen način smo iskali potencialne probleme. Vzporedno je potekalo vrednotenje s strani testnih uporabnikov, ki so bili po večini neprogramerji. Ko so bile napake in zahteve definirane, smo se spet vrnili v fazo načrtovanja.

6.1 Principi načrtovanja uporabniških vmesnikov

Obstaja več principov oz. hevristik, ki so visokonivojski koncepti za načrtovanje uporabniških vmesnikov na podlagi mentalnega modela uporabnika. Oglejmo si nekatere izmed njih.

Mandelovi principi [3]:

1. Zagotovi nadzor uporabnika
    - Omogoči uporabo tipkovnice in miške (fleksibilnost)
    - Omogoči prekinitev danih opravil (prekinljivost)
    - Prikazuj obvestila in tekste (pomoč)
    - Zagotovi takojšnje in ponovljive akcije ter povratno informacijo (odzivnost)
    - Zagotovi značilne poti in izhod (navigacija)
    - Prilagodi se uporabnikom z različnimi nivoji znanja (dostopnost)
    - Zagotovi jasnost in čistost vmesnika (preglednost)
    - Omogoči spreminjanje lastnosti vmesnika (želje)
    - Omogoči direktno manipulacijo z grafičnimi gradniki (interaktivnost)
    - Uporabljaj »načine« pametno (nedvoumnost)
2. Reduciraj obremenitev uporabnikovega spomina
    - Razbremenjuj kratkotrajni spomin (pomnjenje)
    - Zanašaj se na »razpoznavanje« in ne na spomin (prepoznavanje)
    - Zagotovi vizuelne namige (informiranost)
    - Zagotovi vgrajene akcije in razveljavitev ter ponovitev akcij (preprostost, reševanje)
    - Zagotovi bližnjice (hitrost)
    - Podpiraj način gradnik-akcija (intuitivnost)

   - Uporabljaj metafore realnega sveta (prenos)
   - Uporabljaj progresivni dostop (navigabilnost)
   - Podpiraj vizuelno čistost (organiziranost)
3. Zagotovi konsistentnost
   - Uporabljaj kontekst uporabnikovih opravil (zveznost)
   - Ohranjaj enovitost v predstavitvi informacij, obnašanju gradnikov in tehnikah interakcije (izkušnje)
   - Ohranjaj enovitost rezultatov interakcij (pričakovanje)
   - Zagotovi estetsko privlačnost in polnost (izgled)
   - Vzpodbujaj preiskovanje (napovedljivost)

Nielsenovih 10 principov [3]:

1. Prilagodi se realnemu svetu
   - Jezik naj bo prilagojen dani domeni
   - Ne omejuj imen, definiranih s strani uporabnikov
   - Dovoli okrajšave in sinonime v ukaznih jezikih
   - Uporaba smiselnih metafor
2. Konsistentnost in standardi
   - Princip najmanjšega presenečenja
   - Vrstni red ukazov in argumentov
   - Standardi platform
   - Zunanja, notranja konsistentnost
3. Pomoč in dokumentacija
   - Jasen vmesnik mora biti sam po sebi razumljiv
   - Priročnik in takojšnja pomoč sta nujna
   - Uporabniška dokumentacija mora biti kratka, konkretna, lahko iskanje
4. Uporabnikov nadzor in svoboda
   - Dobro označeni izhodi
   - Možnost preklica (Cancel) in razveljavitve akcije (Undo)
   - Zagotovi nadzor nad podatki: podpora za vnos, branje, ažuriranje in brisanje
   - Dolge operacije naj bodo prekinljive
5. Vidljivost statusa sistema
   - Iluzija napredka – uporabnika ves čas jasno obveščaj o trenutnem stanju sistema
   - Vidljivost akcij, ukazov, načinov in stanja navigacije (gumbi, menuji, namigi)
   - Odzivni časi (sprememba kurzorja, indikator napredka)

6. Fleksibilnost in učinkovitost
    - Bližnjice za pogoste operacije
    - Okrajšave ukazov
    - Ukazne datoteke
    - Stili in predloge
    - Zaznamki
    - Privzete nastavitve
    - Zgodovina
    - Pričakovanja
7. Izogibanje napakam
    - Omogoči izbiro namesto tipkanja
    - Onemogoči napačne akcije
    - Menuji in forme namesto ukaznega jezika
    - Kombinirani izvlečni seznami namesto besedilnega območja
    - Zaščita uporabnikovega dela (samodejni Save, Undo)
8. Raje prepoznaj kot si zapomni
    - Menuji namesto ukaznega jezika
    - Uporaba kombiniranih izvlečnih seznamov namesto tekstovnih polj
    - Uporaba generičnih ukazov, kjer je to možno (Open, Save, Copy...)
    - Vsa potrebna informacija naj bo vidna
9. Javljanje napak, diagnoza, reševanje
    - Natančno, dobro obvestilo o napaki
    - Ne grajaj uporabnika
    - Ponudi konstruktivno pomoč – vzrok napake in kako jo odpraviti
    - Ne obremenjuj uporabnika s tehničnimi detajli, prikaži na željo
10. Estetika in minimalistično načrtovanje
    - Manj je več – preprostost, izogibanje odvečnim informacijam, grafikam, lastnostim, gradniki naj imajo dvojno vlogo
    - Dobro grafično načrtovanje – poravnave, grupiranje, uporaba barv in pisav
    - Kratek in jedrnat jezik

6.2 Navodila

Iz principov izhajajo navodila, ki so dejanski napotki za grajenje uporabniškega vmesnika. V grobem bi lahko rekli, da nas navodila učijo, kako naj sestavimo uporabniški vmesnik, katere gradnike naj pri tem uporabimo ter kakšna naj bo interakcija uporabnika z grafičnim vmesnikom.

Navodila načrtovanja grafičnih vmesnikov pokrivajo naslednja področja [3]:

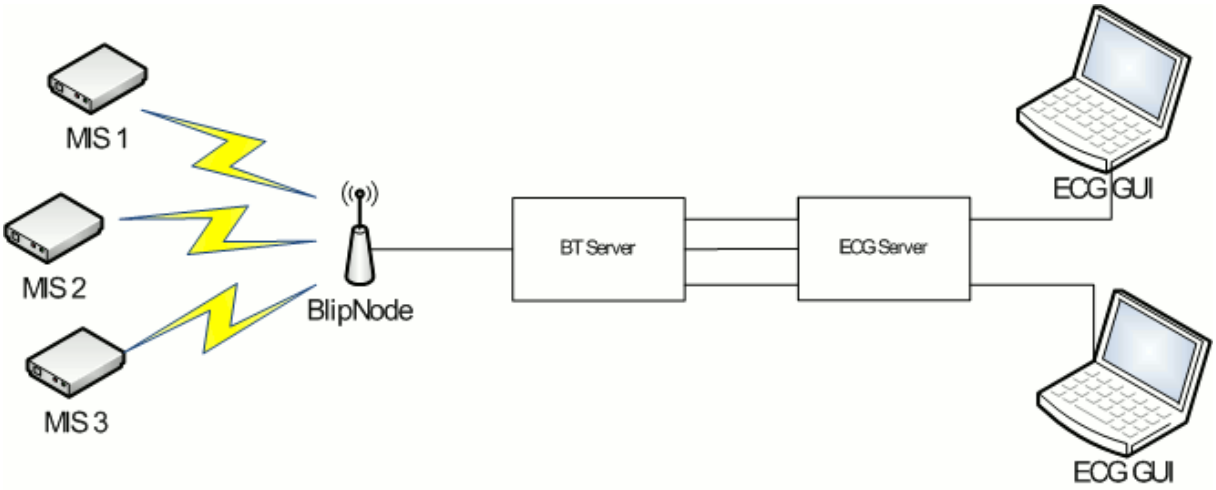
1. Načrtovanje oken
    - Komponente oken
    - Primarno okno
    - Sekundarna okna
    - Dialogi
    - Razvrstitev oken
2. Načrtovanje menujev: izbor najprimernejšega menuja glede na število opcij, pogostost uporabe in prostor, ki ga imamo na voljo
3. Izbor grafičnih gradnikov za interakcijo
    - Kateri gradnik je primeren za določen tip interakcije?
    - Gumbi ali menuji
    - Polje za vnos ali polje za izbiro
    - Stikala ali menuji
    - Seznami ali kombinirani seznami
    - Paleta, krožno polje, drsnik za paramete
4. Aranžiranje grafičnih gradnikov za interakcijo
    - Preprostost, manj je več
    - Balansiranje – enakomerna razporeditev, simetričnost, estetika
    - Grupiranje – boljša preglednost, uporaba belih presledkov, separatorjev in okvirjev
    - Orientacija in poravnava gradnikov
5. Izbor teksta, barv, slik in animacije
    - Kako izberemo obliko in velikost pisave?
    - Kako uporabljamo barve za doseganje različnih učinkov?
    - Kdaj in kako izberemo sliko?
6. Povratna informacija in interakcije
    - Kako prikazujemo obvestila in kakšna naj bodo?
    - Kdaj uporabimo zvok?
    - Kakšni so dialogi?

7 Opis okolja

Celotni sistem za spremljanje krivulje EKG v realnem času je sestavljen iz naslednjih kompoment:

- Brezžični Bluetooth merilniki EKG (Mobile Instrumentation System oz. MIS), ki so bili razviti za potrebe takratnega podjetja Konel;
- Bluetooth brezžična dostopna točka BlipNode podjetja Blip Systems;
- BT Server kot vmesnik, ki povezuje BlipNode z višjenivojsko programsko opremo;
- strežnik, ki omogoča komunikacijo med merilniki in grafičnim vmesnikom (na diagramu spodaj ECG Server);
- grafični vmesnik, preko katerega spremljamo dogajanje celotnega sistema (na diagramu spodaj ECG GUI).

Slika 12 prikazuje diagram komponent sistema in povezave med njimi.



Slika 12: Diagram sistema za merjenje EKG

Merilniki merijo odvode V4, V5 in V6, zato je tudi grafični vmesnik prilagojen prikazu teh treh kanalov.

Nekatere knjižnice in podatkovne strukture, ki so bile razvite za strežnik, so bile uporabljene tudi pri izdelavi grafičnega uporabniškega vmesnika.

7.1 Simulator merilnika

Za potrebe testiranja smo uporabljali simulator merilnika, ki se obnaša enako kot merilnik. V tekstovni datoteki smo shranili numerične vrednosti realne meritve. Simulator bere vrednosti iz datoteke in jih predvaja kot aktivno meritev. Grafični vmesnik se ne zaveda, ali se priključi dejanski merilnik ali simulator.

Ob prvem zagonu simulatorja kreiramo merilnik, nato ga dodelilmo bolniku. Za predvajanje meritve proces merilnika zaganjamo z naslednjimi argumenti:

- identifikacijska oznaka merilnika
- pot do datoteke, ki shranjuje posnetek meritve.

Primer izseka podatkov iz takšne datoteke:

```
-17  -25  1
-17  -25  1
-16  -23  3
-19  -25  0
-18  -25  1
-23  -27  -1
-22  -28  -4
-23  -30  -4
-24  -32  -6
-23  -32  -4
-25  -32  -6
-17  -32  -2
-6   -24  7
7    15   25
11   74   60
1    61   55
-19  -2   23
-36  -24  3
-32  -29  -3
-26  -30  -4
-24  -32  -6
-22  -32  -4
-23  -32  -5
-22  -31  -4
```

```
-23  -31  -6
-22  -31  -6
-21  -30  -4
-20  -30  -4
-18  -28  -2
-18  -28  -3
-16  -28  -2
-16  -26  0
-14  -26  0
-11  -21  4
-11  -20  3
-8   -17  5
-7   -14  6
-6   -14  7
-2   -11  10
-4   -10  10
-2   -7   11
-4   -9   11
```

Stolpci prikazujejo vrednosti odvodov po naslednjem vrstnem redu: V5, V6, V4. Pri tem predstavlja 1 enota 20μV.

7.2 Detekcija kritičnih stanj in proženje alarmov

Ločimo tri prioritete oz. stopnje nujnosti alarmov:

- opozorila
- resno oz. nevarno stanje
- življenjska nevarnost

Strežnik ves čas sproti analizira izmerjene vrednosti. Vrednosti se primerjajo s pogoji v filtrih. V primeru, da izmerjene vrednosti ustrezajo kateremu od definiranih kriterijev, strežnik sproži alarm. Alarmu nastavi pričakovano življenjsko dobo. Če po izteku življenjske dobe pogoji še vedno ustrezajo filtru, se trajanje alarma podaljša. Pri nekaterih alarmih se ob podaljšanju poviša stopnja nujnosti. V nasprotnem primeru alarm poteče. Grafični vmesnik alarm prestreže in ga prikaže.

Sistem je sposoben zaznati naslednja stanja:

- apnea
- asistolija
- bradikardija
- tahikardija

Poleg alarmov, ki se prožijo glede na izmerjene vrednosti, je sistem sposoben javljati stanje baterije merilnika in opozoriti, ko je baterijo potrebno zamenjati/napolniti. Ta alarm ni konfigurabilen, saj ga javlja merilnik sam.

Privzeta konfiguracija za alarme:

```
#ID  E/D  PARAMETER CONDITION THRESHOLD DELAY_SEC ALARM
DEFAULT_LIFETIME_SEC
#-------------------------------------------------------------
-----------
D1   1    RESP_FREQ      <    3    30   ApneaW    10
D2   1    RESP_FREQ      <    3    60   ApneaLT   10

D3   1    RR_INTERVAL    >    2000 0    MissedBeatW    0
D4   1    RR_INTERVAL    >    5000 0    MissedBeatS    0
D5   1    HACTION        <    10   10   Asystolia 10

D6   1    HB_RATE   <    60   10   BradycardiaW   10
D7   1    HB_RATE   <    40   10   BradycardiaS        10
D8   1    HB_RATE   <    20   10   BradycardiaLT  10

D9   1    HB_RATE   >    150  10   TachycardiaW   10
D11  1    HB_RATE   >    200  10   TachycardiaS   10
D12  1    HB_RATE   >    250  10   TachycardiaLT  10


D100 1    ecg1/SNR_ecg1  >    100  5    LeadOffEcg1 10
D101 1    ecg2/SNR_ecg2  >    100  5    LeadOffEcg2 10
D102 1    ecg3/SNR_ecg3  >    100  5    LeadOffEcg3 10
```

Seznam veljavnih nizov za kriterije:

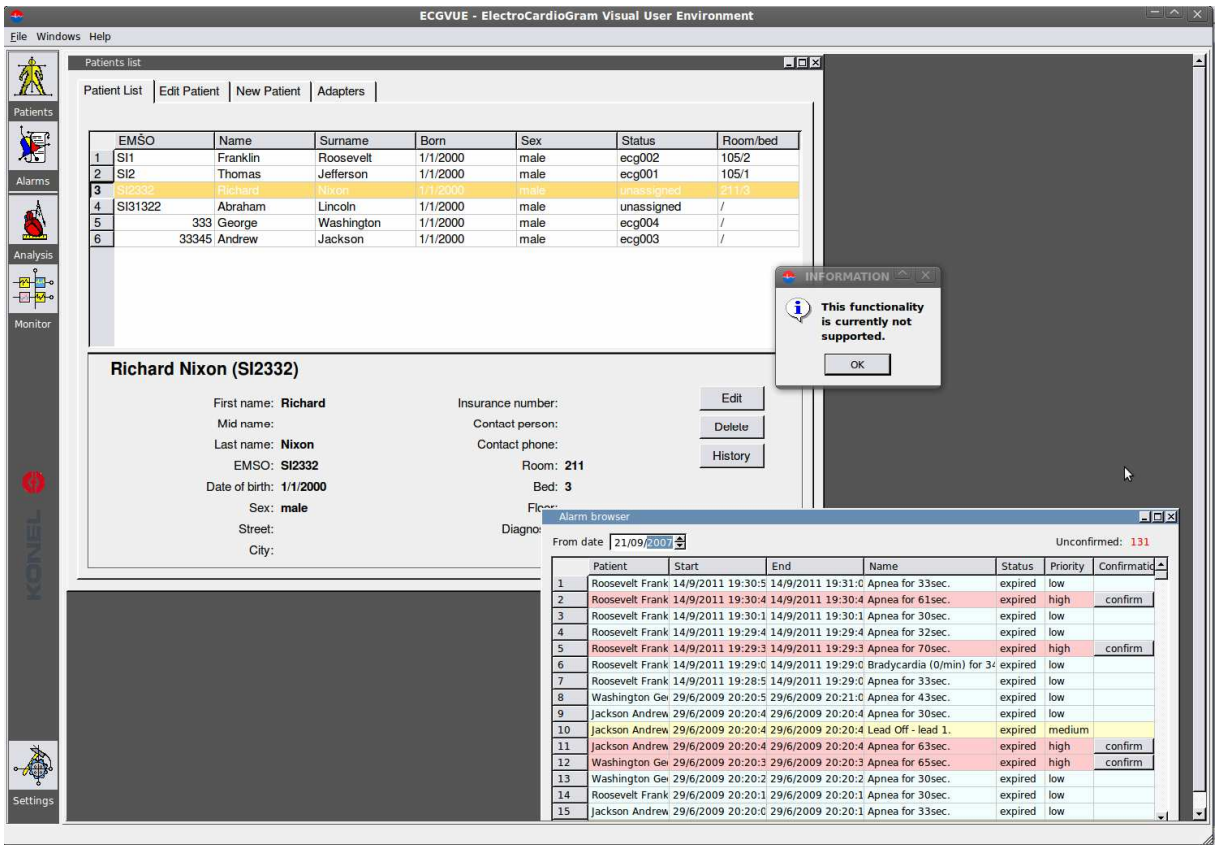
- RESP_FREQ: frekvenca dihanja oz. število vdihov na minuto
- RR_INTERVAL: čas med dvema utripoma

- HACTION: vrednost, ki je pri normalnem utripu enaka srčni frekvenci, ob odsotnosti pulza pa pade za 30% vsake 3 sekunde
- HB_RATE: število srčnih utripov na minuto
- ecg1, ecg2, ecg3: izmerjene vrednosti merilnika
- SNR_ecg1, SNR_ecg2, SNR_ecg3: ocena razmerja med signalom meritve in šumom oz. motnjami (ang. signal to noise ratio)

8 Rezultati dela

8.1 Predstavitev grafičnega vmesnika

Grafični vmesnik sestavlja primarno okno, v katerem se odpirajo sekundarna notranja okna. Slika 13 prikazuje osnovno oz. primarno okno grafičnega vmesnika z odprtimi nekaterimi notranjimi okni. Osnovno okno vsebuje vrstični menu zgoraj, orodno vrstico s povezavami na osnovne funkcionalnosti levo in osrednji del, v katerem se opirajo notranja okna. Želeli smo oblikovati enostavno in estetsko okno, ki upošteva 10. Nielsenov princip minimalizma. Spodnja vrstica osnovnega okna je rezervirana kot območje za obvestila, saj smo želeli upoštevati 5. Nielsenov princip, ki svetuje stalno vidljivost statusa sistema. Statusna vrstica trenutno ni v uporabi, nameravali smo jo vključiti kasneje.



Slika 13: Osnovno okno grafičnega vmesnika

Grafični vmesnik vsebuje naslednje vsebinske sklope:

- bolniki in merilniki

- aktivne meritve
- zgodovina alarmov
- zgodovina meritev

Levi rob osnovnega okna vsebuje orodno vrstico. Ta vsebuje naslednje povezave:

- bolniki: odpre seznam bolnikov
- alarmi: odpre seznam alarmov
- analiza: ni v rabi
- aktivne meritve: odpre aktivne meritve za več bolnikov hkrati
- nastavitve: osnovne nastavitve vmesnika, kot so barve in kontrasti

Z uvedbo bližnjic preko tipkovnice bi lahko upoštevali 6. Nielsenov princip, ki nas uči, da bližnjice povečajo učinkovitost. Ker osnovnih možnosti ni veliko, se za to nismo odločili.

Vsa okna se odpirajo znotraj osnovnega okna. Osnovno okno ponuja možnost avtomatičnega sortiranja notranjih oken: kaskadno oz. prekrivajoče in razpostavljeno eno zraven drugega. Odprta notranja okna in njihovi položaji znotraj osnovnega okna se ob naslednjem zagonu vmesnika ohranijo. Ohrani se tudi izbrano sortiranje v tabelah. Notranja okna niso modalna, ker želimo dopuščati možnost enostavnega preklapljanja med različnimi področji.

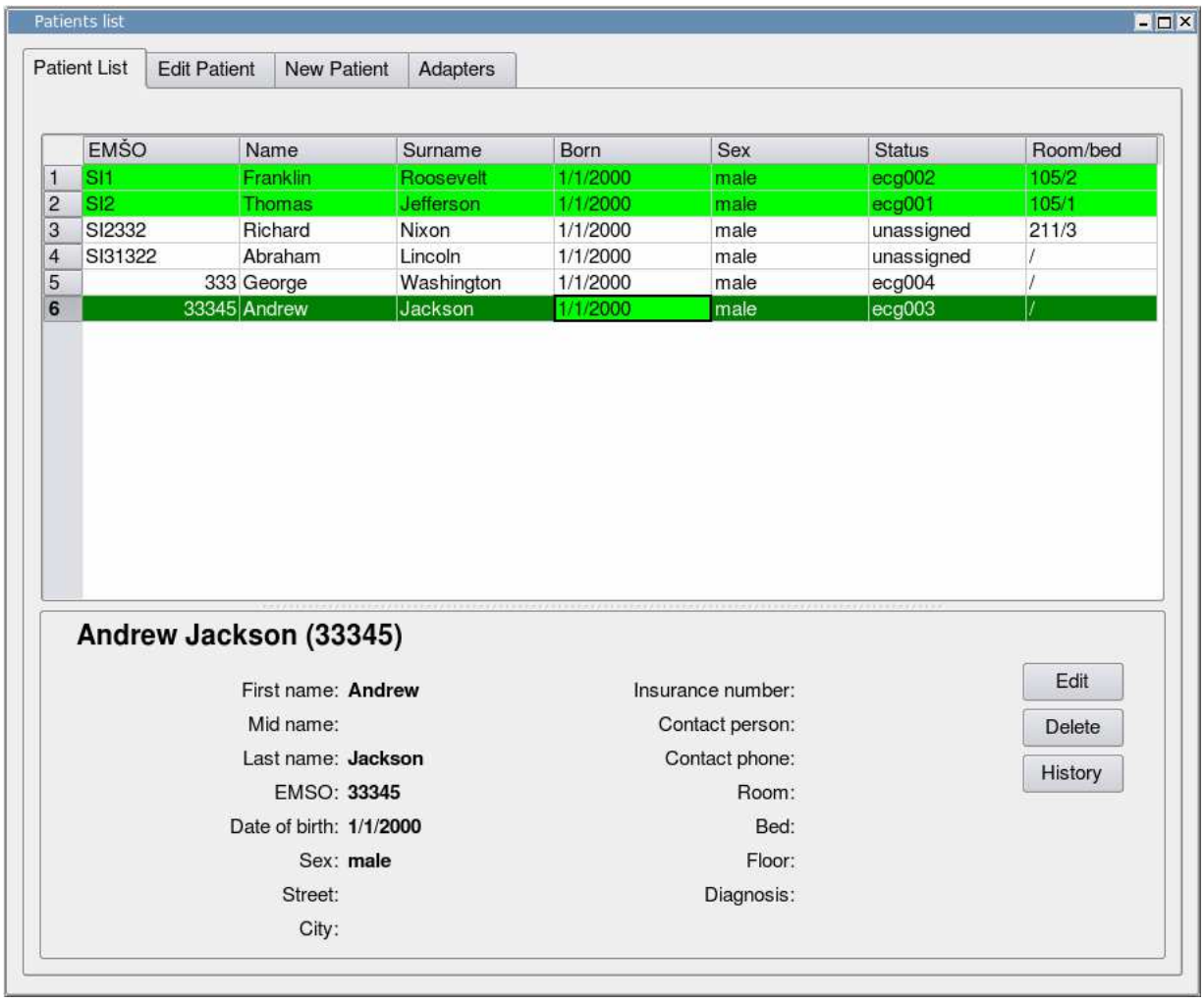
Glavni menu osnovnega okna je trenutno precej okrnjen in vsebuje samo osnovne funkcionalnosti nad okni. Vsebuje naslednje povezave:

- File → Exit
- Window → Cascade
- Window → Tile
- Help → Documentation
- Help → About

Menu bi bilo potrebno razširiti še z vsebinskim menujem (npr. Application), ki bi vseboval povezave do vseh funkcionalnosti, ki so dostopne preko orodne vrstice. Ker trenutno menu vsebuje zelo malo možnosti, grupiranje ukazov in uporaba separatorjev ni smiselna.

8.2 Bolniki

Slika 14 prikazuje okno bolnikov, ki izgleda takole:



Slika 14: Seznam bolnikov

Vsebuje osnovne podatke o bolnikih in merilnikih. Sestavljeno je iz naslednjih zavihkov:

- seznam bolnikov (ang. Patients list)
- urejanje bolnika (ang. Edit patient)
- dodajanje bolnika (ang. New patient)
- merilniki (ang. Adapters)

V skladu s 6. Nielsenovim principom bi lahko povečali fleksibilnost in učinkovitost, če bi vpeljali bližnjice za pomikanje med zavihki. Tako bi lahko npr. kombinacija CTRL+L odprla seznam bolnikov, ob pritisku CTRL+N pa bi postal aktiven zavihek za dodajanje novega bolnika. Uporabnikov nadzor (4. Nielsenov princip) je zagotovljen z možnostmi vnosa, pregledovanja, urejanja in brisanja podatkov.

Za vsakega bolnika hranimo naslednje podatke:

- ime, srednje ime, priimek
- spol
- datum rojstva
- enolična identifikacija (EMŠO)
- naslov
- kontaktna oseba (ime, telefonska številka)
- lokacija v bolnišnici (nadstropje, soba, postelja)
- diagnoza
- mejne vrednosti za proženje alarmov (ob vnosu novega bolnika v sistem se dodelijo privzete vrednosti)
- zgodovina merjenj krivulje EKG
- dodeljeni merilnik

V seznamu bolnikov prikazujemo osnovne podatke o bolnikih ter dodeljenih merilnikih. Podatki so prikazani v obliki tabele. Seznam omogoča dvosmerno razporejanje po kateremkoli stolpcu. Vmesnik samostojno zazna, kdaj je merilnik bolnika priklopljen in oddaja podatke. Takrat se vrstica s podatki bolnika v seznamu obarva zeleno. Ko meritev ne poteka, je ozadje vrstice belo. Tako je že na prvi pogled jasno, kakšno je stanje meritve za posameznega bolnika, s čimer upoštevamo 5. Nielsenov princip.

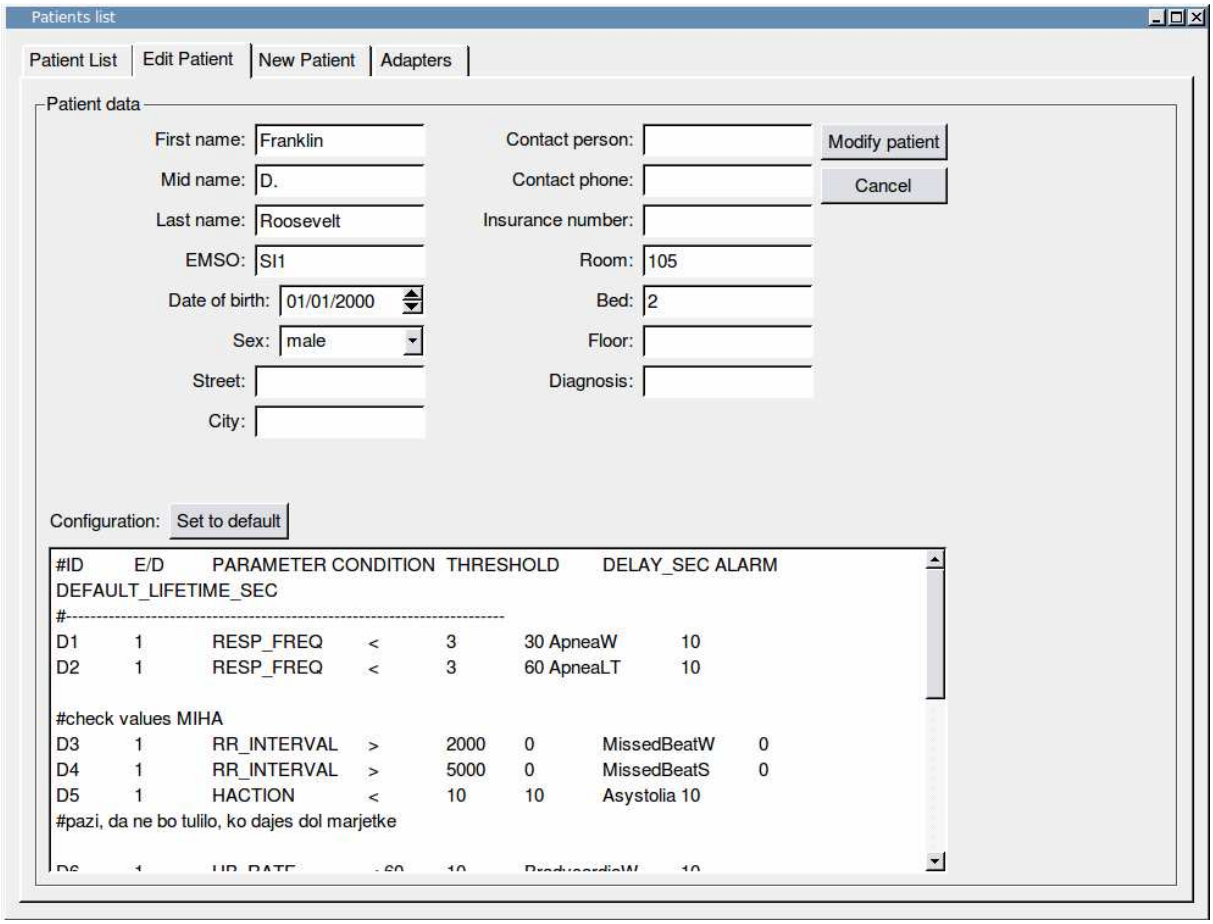
Spodnji del okna vsebuje detajlno okno po načinu osnovno-detajlno (ang. master-detail), v katerem vidimo dodatne informacije o izbranem bolniku. Vsebina se dinamično osveži, če v tabeli izberemo drugo vrstico.

Detajli so prikazani v okvirju, vsebina je izpisana v dveh vertikalno orientiranih stolpcih. Pri tem so pojasnila desno poravnana, polja oz. vsebina pa levo poravnana. Za dva stolpca smo se odločili zato, da je prostor čim boljše izkoriščen, s smiselno poravnavo pa ne izgubimo na preglednosti. Desno se nahajajo še gumbi s povezavami do nekaterih akcij, povezanih z izbranim bolnikom:

- spreminjanje podatkov: preusmeri na zavihek za urejanje podatkov
- brisanje: prikaže dialog za potrditev brisanja in v primeru potrditve izbriše bolnika iz seznama
- zgodovina meritev: odpre okno z zgodovino meritev za izbranega bolnika

Gumbi so razporejeni en pod drugim, enako veliki in vertikalno poravnani. S takšno poravnavo komponent smo hoteli doseči uravnotežen izgled okna in preglednost.

Zavihka za spreminjanje podatkov o bolniku in dodajanje novega bolnika sta vizuelno enaka, s čimer upoštevamo 2. Nielsenov princip konsistentnosti. Zavihek za dodajanje bolnika se odpre prazen, vpisana je samo privzeta konfiguracija za alarme. 6. Nielsenov princip nas uči, da povečamo učinkovitost, če uporabniku ponudimo privzete nastavitve. Zavihek za urejanje podatkov pa je napolnjen z že obstoječimi podatki izbranega bolnika. Slika 15 prikazuje zavihek za spreminjanje podatkov o bolniku.



Slika 15: Urejanje podatkov o bolniku

Pod polji za vnos osnovnih podatkov o bolniku se v zavihku za dodajanje bolnikov nahaja tudi vnosno polje za konfiguracijo alarmov, ki vsebuje privzeto konfiguracijo v tekstovni obliki. Mejne vrednosti za proženje alarma in zamik oz. dobo trajanja dogodka pred proženjem alarma je možno spreminjati za vsakega bolnika posebej. Vedno pa je možno povrniti privzeto konfiguracijo, kar zlasti pride prav, če zaradi napake pri vnosu pride do neveljavne konfiguracije. Ker vnašamo večjo količino besedila, smo izbrali vnosno polje, ki omogoča vnos več vrstic hkrati z drsnikom na desnem robu.

Spremembo podatkov bolnika lahko potrdimo z gumbom Modify patient in prekličemo z gumbom Cancel. To in možnost povrnitve privzete konfiguracije za alarme poveča uporabnikov nadzor in svobodo, o čimer govori 4. Nielsenov princip. 7. Neilsenov princip pa predlaga vpeljavo gumba Undo oz. Razveljavi, s čimer bi še dodatno olajšali izogibanje napakam in zaščitili uporabnikovo delo.

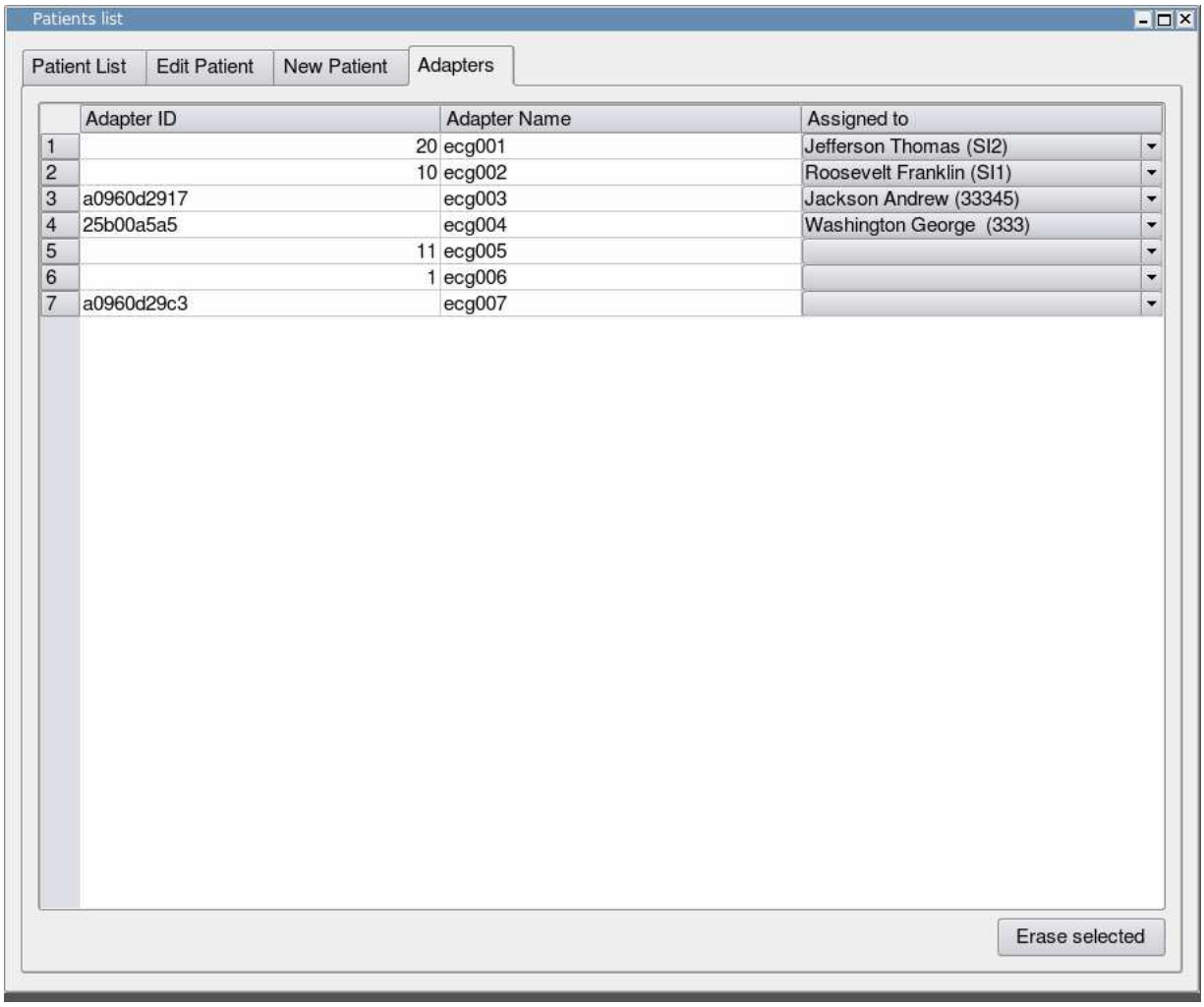
Podobno kot pri seznamu bolnikov so tudi tu vnosna polja razporejena v dva stolpca. Pri tem so pojasnila in vnosna polja desno poravnana. Polja, v katera vpisujemo poljubno besedilo, so običajna vnosna polja. Polje za vnos datuma rojstva je komponenta za izbiro datuma, ki je posebne vrste krožno polje. Z izbiro te komponente uporabniku olajšamo vnos datuma, hkrati pa ne dopuščamo vnosa neustrezne vsebine, zato naknadna validacija ni potrebna. Spol bolnika izberemo preko izvlečnega seznama, čeprav bi glede na samo dve možni izbiri lahko izbrali tudi stikalo z eno možnostjo. Na takšen način se v skladu s 7. Nielsenovim principom izogibamo napakam, saj možnost izbire vedno lažja od ročnega vnosa in ne dopušča napake.

Podatki bolnika in konfiguracija alarmov so za boljšo preglednost ločeni z belim presledkom. S takšno postavitvijo upoštevamo 10. Nielsenov princip, ki nas uči, kako doseči estetski izgled.

8.3 Merilniki

Merilnik je avtomatsko dodan v seznam merilnikov, ko ga strežnik prvič zazna. Ročno dodajanje preko grafičnega vmesnika ni mogoče. Enolična oznaka merilnika ni spremenljiva in je ponavadi kar strojni naslov MAC merilnika, ki ga merilnik sporoči ob povezavi na sistem. V primeru uporabe simulatorja pa je oznaka niz alfanumeričnih znakov, ki ga podamo simulatorju kot vhodni parameter ob zagonu. Opisno ime pri novem merilniku ni nastavljeno. Nov merilnik ni dodeljen nobenemu bolniku. Uporabnik ima možnost določiti opisno ime merilnika in merilnik dodelilti bolniku iz seznama bolnikov.

Merilniki so predstavljeni v obliki tabele (Slika 16). Merilnik bolniku dodelimo tako, da izberemo bolnika iz izvlečnega seznama. Za izvlečni seznam smo se odločili zato, ker želimo omogočati samo izbiro med obstoječimi bolniki, zato običajno vnosno polje ne bi prišlo v poštev. Na takšen način upoštevamo 7. Nielsenov princip, ki uči, da je izbira vedno boljša od vnosa z vidika izogibanja napakam. Zaradi boljše preglednosti in omejenega prostora želimo, da se za vsak merilnik vidi samo izbrani bolnik, zato tudi npr. običajna lista ne bi bila primerna izbira.



Slika 16: Seznam merilnikov

Brisanje merilnikov je možno, vendar se v tem primeru ob ponovnem zagonu merilnika kreira nova entiteta, ki ni dodeljena nobenemu bolniku in nima določenega opisnega imena. Predhodne nastavitve se ob brisanju izgubijo. Merilnik izbrišemo tako, da ga izberemo s klikom na vrstico v tabeli in kliknemo gumb za izbris.

8.4 Aktivne meritve

Aktivne meritve lahko spremljamo za enega ali več bolnikov hkrati. Da bi se izognili nepotrebnemu podvajanju kode, smo za potrebe obeh oken razvili nekatere skupne komponente, ki jih večkrat uporabimo:

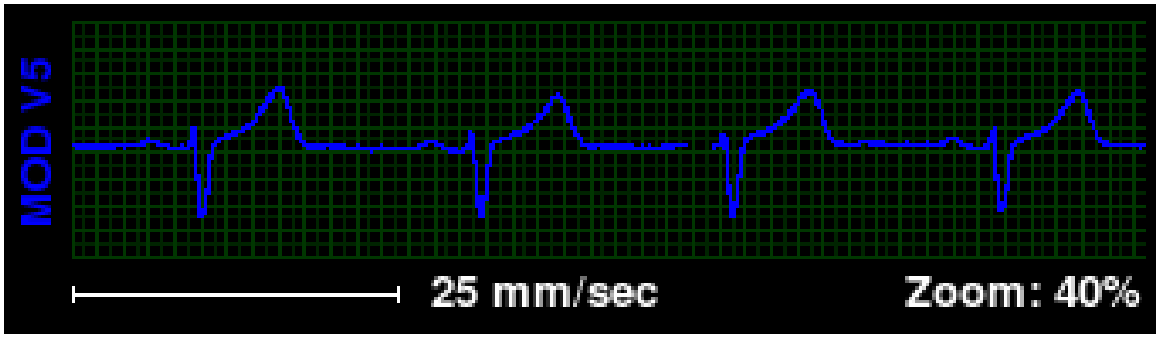
- podoba z osnovnimi podatki o bolniku, ki se barva glede na trenutni alarm in omogoča prilagajanje velikosti izrisanih krivulj

- podoba za izpis trenutnega srčnega utripa, ki se dinamično osvežuje
- podoba za izris krivulje meritve
- podoba za prikaz alarmov

Z večkratno uporabo skupnih komponent upoštevamo 2. Nielsenov princip. Konsistentnost zagotavljamo tako, da podobne stvari tudi izgledajo podobno ne glede na to, kje se uporabljajo.

8.4.1 Podoba za izris krivulje

Podoba za izris krivulje (Slika 17) je popolnoma prilagodljiva. Ob kreiranju ji določimo oznako krivulje, barvo, velikost in merilo. Podajamo ji numerične vrednosti, ki se sproti v realnem času izrisujejo v graf. Velikost lahko naknadno poljubno spreminjamo. Ob spremembi velikosti podobe se dinamično prilagodi merilo izrisanega grafa, medtem ko hitrost izrisovanja ostane vedno enaka, to je 25mm na sekundo za krivulje EKG in 5mm na sekundo za respiratorno krivuljo.

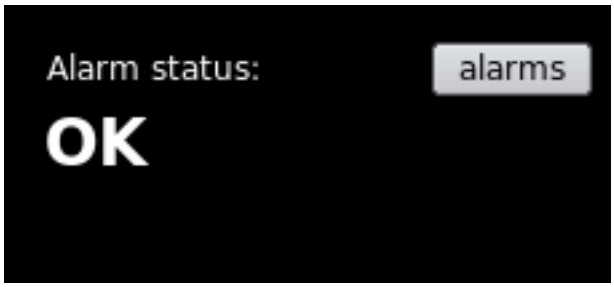


Slika 17: Podoba za izris krivulje

Z izbiro temno zelene mreže na črnem ozadju smo želeli doseči harmonijo. Z izrisom krivulj v svetlih intenzivnih barvah na črnem ozadju pa smo želeli z uporabo kontrasta doseči dobro vidljivost.

8.4.2 Podoba za prikaz alarmov

Podoba za prikaz alarmov prikazuje trenutne alarme. V času, ko ni aktivnih alarmov, je izpisano sporočilo „OK“, ozadje pa je črne barve (Slika 18).



Slika 18: Podoba za prikaz alarmov v času, ko ni nobenega alarma

Slika 19: Podoba za prikaz alarmov v času aktivnega alarma srednje prioritete

Izgled sporočila alarma je odvisen od prioritete alarma:

- Alarmi najnižje prioritete so opozorila. Ozadje podobe obarvajo s svetlo cianovo barvo. Ne zahtevajo nobene posebne akcije. Ko alarm poteče, ga ne prikazujemo več.
- Alarmi srednje prioritete označujejo resne oz. možno nevarne dogodke. Ozadje podobe obarvajo z rumeno barvo (Slika 19). Ozadje počasi utripa s hitrostjo en utrip na vsaki dve sekundi. Alarm zahteva potrditev. Če poteče brez potrditve, gumb za prikaz potridtve utripa rdeče dokler alarm ni potrjen.
- Alarmi najvišje prioritete označujejo dogodke, ki lahko ogrožajo življenje bolnika. Ozadje podobe obarvajo z rdečo barvo. Ozadje utripa hitreje s hitrostjo dveh utripov na sekundo. Oglasi se zvočni signal. Alarm zahteva potrditev. Če poteče brez potrditve, gumb za prikaz potrditve utripa rdeče dokler alarm ni potrjen.

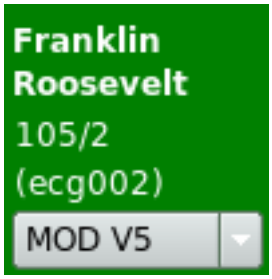
Pri izboru barv smo upoštevali naslednja navodila glede izbora barv:

- vzbujanje pozornosti: alarmi so živih, svetlih barv, da so dobro opazni
- poudarjanje podobnosti: alarm visoke prioritete rdeč, srednje prioritete pa rumen
- izražanje skupnega pomena
- tople barve predstavljajo nujne akcije – alarmi, ki zahtevajo potrditev, so rdeči in rumeni
- hladne barve (svetlo cianova) za alarme najnižje prioritete

8.4.3 Podoba z osnovnimi podatki o bolniku

Podoba z osnovnimi podatki o bolniku ima izpisano ime in priimek bolnika, njegovo lokacijo v bolnišnici in ime dodeljenega merilnika. Vsebuje tudi dve komponenti, ki ju prikažemo in skrijemo po potrebi: polje zoom za nastavitev velikosti grafa in izvlečni menu za izbiro krivulje za prikaz.

Polje zoom je krožno polje, ki omogoča tudi vnos preko tipkovnice. Polju je dodeljena maska, ki dopušča samo vnos številk, ne pa tudi ostalih alfanumeričnih znakov. Za izbiro izrisovane krivulje smo izbrali izvlečni menu predvsem zato, ker smo želeli ponuditi uporabniku omejen predefiniran nabor opcij za izbiro. Hkrati smo želeli, da je izbrana krivulja jasno vidna in ne zasede preveč prostora. Z uporabo komponent za izbiro namesto vnosa preprečujemo napake v skladu s 7. Nielsenovim principom.

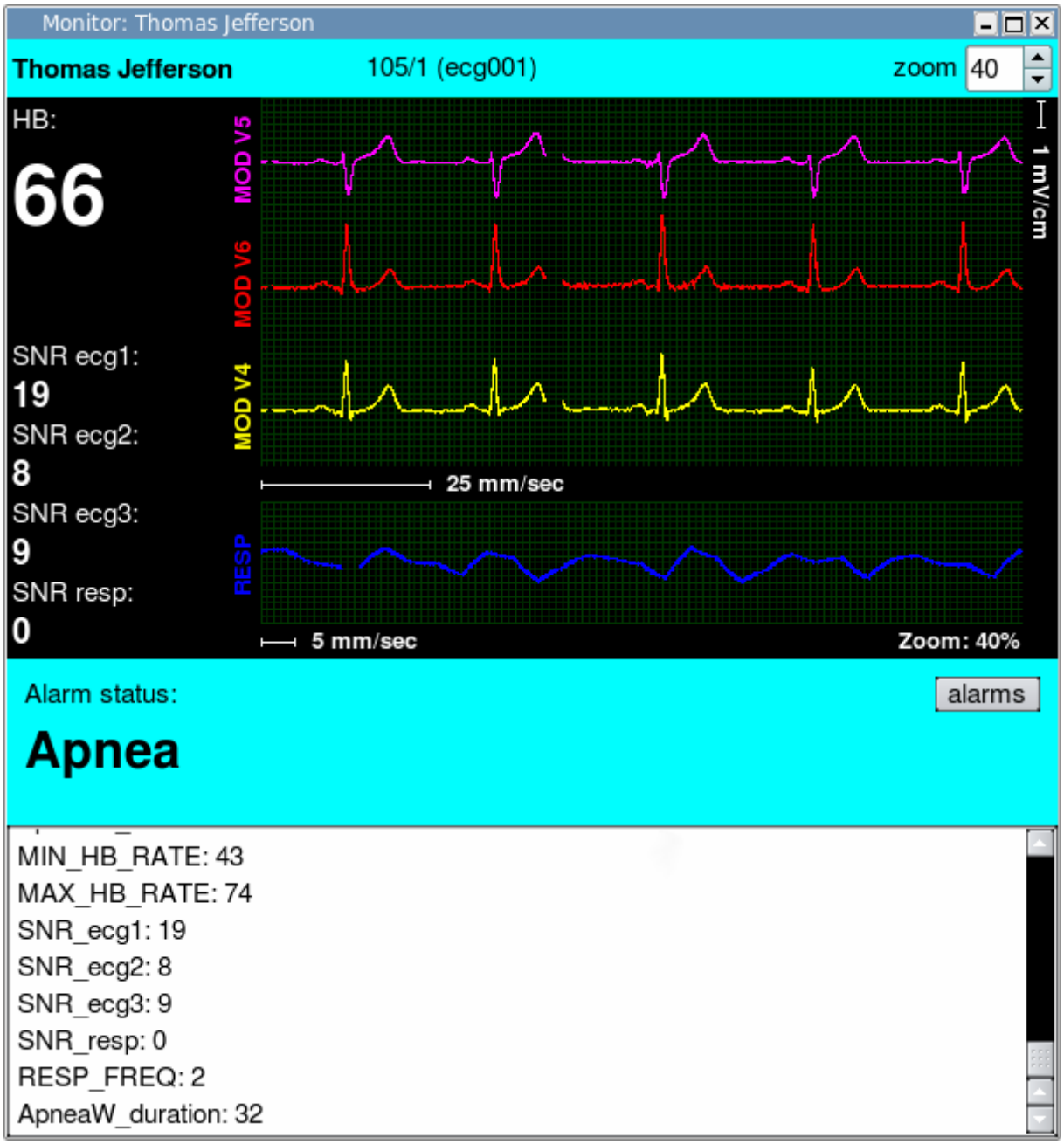


Slika 20: Podoba z osnovnimi podatki o bolniku v času aktivne meritve

Ta podoba se obnaša usklajeno s podobo za prikaz alarmov. V primeru aktivnega alarma se tudi ozadje te podobe enako obarva. S tem upoštevamo 2. Nielsenov princip konsistentnosti. Slika 20 prikazuje podobo v času, ko ni aktivnih alarmov in meritev poteka. Takrat je ozadje obarvano zeleno. V času, ko meritev ne poteka, pa je ozadje črno. Z izborom kontrastnih barv smo želeli čimbolj ločljivo prikazati razliko med posameznimi stanji: alarm, ni alarma, merilnik ugasnjen. Iz barve se že na prvi pogled jasno vidi status meritve v skladu s 5. Nielsenovim principom.

8.4.4 Aktivna meritev za posameznega bolnika

Slika 21 prikazuje okno s prikazom aktivne meritve enega bolnika in izgleda tako:



Slika 21: Aktivna meritev za enega bolnika

Izrisujejo se krivulje odvodov V3, V4 in V5 ter respiratorno krivuljo, ki jo strežnik sproti izračunava iz izmerjenih odvodov. Graf je možno povečati ali pomanjšati s spreminjanjem vrednosti polja zoom. Velikost izrisanih krivulj se dinamično spremeni tudi ob ročnem spreminjanju velikosti okna. Velikost je možno spreminjati samo vsem grafom v trenutnem oknu hkrati.

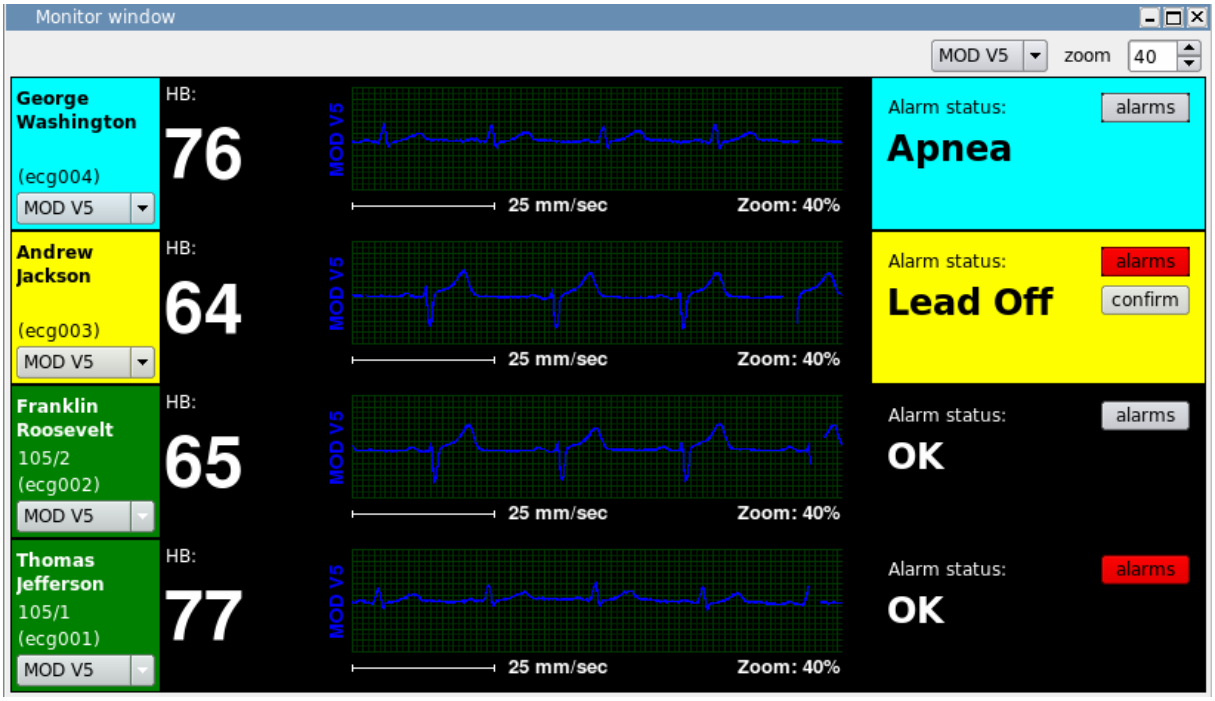
Na oknu se izpisuje trenutni srčni utrip, trenutni alarmi in trenutna ocena razmerja med kakovostjo signala in šumom posamezne krivulje. Za potrebe testiranja okno vsebuje tudi izvlečno polje, v katerem se izpisujejo vsi dogodki okna. To polje je privzeto skrito, iz končne izdaje vmesnika pa naj bi ga popolnoma odstranili.

Za enega bolnika je lahko odprto samo eno okno tega tipa, medtem ko je lahko hkrati odprtih še več oken istega tipa za druge bolnike.

Okno je vertikalno orientirano in levo poravnano. Za boljšo ločljivost je vsaka krivulja druge barve. Za poudarjanje kontrasta je ozadje črno, izrisane krivulje živih svetlih barv, numerični podatki (utrip, SNR) pa beli. Ozadje grafa je zaradi harmonije temno zeleno.

8.4.5 Aktivne meritve za več bolnikov hkrati

Slika 22 prikazuje okno za prikaz aktivnih meritev za več bolnikov hkrati in je podobno oknu za meritev posameznega bolnika. Izgleda takole:



Slika 22: Prikaz aktivnih meritev za več bolnikov hkrati

V oknu za prikaz aktivnih meritev za več bolnikov lahko prikazujemo vse bolnike ali pa prikaz filtriramo glede na lokacijo v bolnišnici: nadstropje ali soba. Lahko je odprtih več oken tega tipa, vsako od njih lahko prikaz filtrira po drugem kriteriju. Okno prikazuje samo tiste bolnike, ki imajo dodeljen merilnik in pri katerih poteka aktivna meritev.

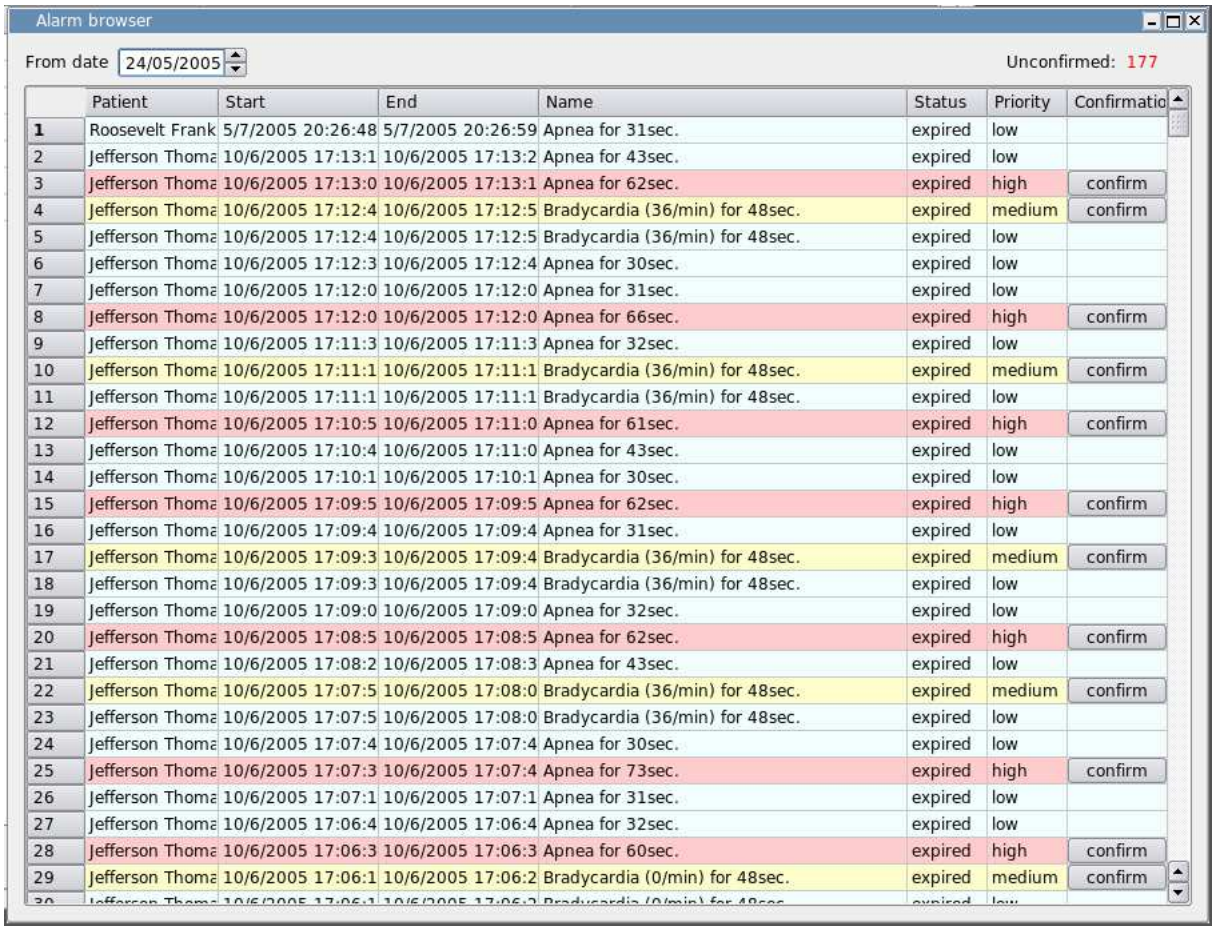
Za vsakega bolnika se prikazuje ena krivulja, trenutni srčni utrip in trenutni alarmi. Uporabnik lahko za vsakega bolnika posebej izbere krivuljo, ki se izrisuje. Lahko pa krivuljo izbere tudi za vse bolnike trenutnega okna. Izbira lahko med odvodi V4, V5 in V6 ter respiratorno krivuljo.

Izrisani grafi se dinamično povečajo ali pomanjšajo, če spremenimo velikost okna. Lahko pa velikost grafa spreminjamo tudi tako, da nastavljamo vrednost polja zoom. Tudi v tem oknu je možno spreminjati velikost samo vsem grafom trenutnega okna hkrati.

Komponente za vsakega bolnika so horizontalno orientirane. Komponente enakega tipa pa so razporejene ena pod drugo, enake velikosti in poravnane. Potrebna bi bila bolj očitna navpična razločitev, da bi se bolj jasno razločilo, za katerega bolnika se izrisuje posamezen graf.

8.5 Zgodovina alarmov

Okno za prikaz zgodovine alarmov izgleda takole:



	Patient	Start	End	Name	Status	Priority	Confirmatio
1	Roosevelt Frank	5/7/2005 20:26:48	5/7/2005 20:26:59	Apnea for 31sec.	expired	low	
2	Jefferson Thoma	10/6/2005 17:13:1	10/6/2005 17:13:2	Apnea for 43sec.	expired	low	
3	Jefferson Thoma	10/6/2005 17:13:0	10/6/2005 17:13:1	Apnea for 62sec.	expired	high	confirm
4	Jefferson Thoma	10/6/2005 17:12:4	10/6/2005 17:12:5	Bradycardia (36/min) for 48sec.	expired	medium	confirm
5	Jefferson Thoma	10/6/2005 17:12:4	10/6/2005 17:12:5	Bradycardia (36/min) for 48sec.	expired	low	
6	Jefferson Thoma	10/6/2005 17:12:3	10/6/2005 17:12:4	Apnea for 30sec.	expired	low	
7	Jefferson Thoma	10/6/2005 17:12:0	10/6/2005 17:12:0	Apnea for 31sec.	expired	low	
8	Jefferson Thoma	10/6/2005 17:12:0	10/6/2005 17:12:0	Apnea for 66sec.	expired	high	confirm
9	Jefferson Thoma	10/6/2005 17:11:3	10/6/2005 17:11:3	Apnea for 32sec.	expired	low	
10	Jefferson Thoma	10/6/2005 17:11:1	10/6/2005 17:11:1	Bradycardia (36/min) for 48sec.	expired	medium	confirm
11	Jefferson Thoma	10/6/2005 17:11:1	10/6/2005 17:11:1	Bradycardia (36/min) for 48sec.	expired	low	
12	Jefferson Thoma	10/6/2005 17:10:5	10/6/2005 17:11:0	Apnea for 61sec.	expired	high	confirm
13	Jefferson Thoma	10/6/2005 17:10:4	10/6/2005 17:11:0	Apnea for 43sec.	expired	low	
14	Jefferson Thoma	10/6/2005 17:10:1	10/6/2005 17:10:1	Apnea for 30sec.	expired	low	
15	Jefferson Thoma	10/6/2005 17:09:5	10/6/2005 17:09:5	Apnea for 62sec.	expired	high	confirm
16	Jefferson Thoma	10/6/2005 17:09:4	10/6/2005 17:09:4	Apnea for 31sec.	expired	low	
17	Jefferson Thoma	10/6/2005 17:09:3	10/6/2005 17:09:4	Bradycardia (36/min) for 48sec.	expired	medium	confirm
18	Jefferson Thoma	10/6/2005 17:09:3	10/6/2005 17:09:4	Bradycardia (36/min) for 48sec.	expired	low	
19	Jefferson Thoma	10/6/2005 17:09:0	10/6/2005 17:09:0	Apnea for 32sec.	expired	low	
20	Jefferson Thoma	10/6/2005 17:08:5	10/6/2005 17:08:5	Apnea for 62sec.	expired	high	confirm
21	Jefferson Thoma	10/6/2005 17:08:2	10/6/2005 17:08:3	Apnea for 43sec.	expired	low	
22	Jefferson Thoma	10/6/2005 17:07:5	10/6/2005 17:08:0	Bradycardia (36/min) for 48sec.	expired	medium	confirm
23	Jefferson Thoma	10/6/2005 17:07:5	10/6/2005 17:08:0	Bradycardia (36/min) for 48sec.	expired	low	
24	Jefferson Thoma	10/6/2005 17:07:4	10/6/2005 17:07:4	Apnea for 30sec.	expired	low	
25	Jefferson Thoma	10/6/2005 17:07:3	10/6/2005 17:07:4	Apnea for 73sec.	expired	high	confirm
26	Jefferson Thoma	10/6/2005 17:07:1	10/6/2005 17:07:1	Apnea for 31sec.	expired	low	
27	Jefferson Thoma	10/6/2005 17:06:4	10/6/2005 17:06:4	Apnea for 32sec.	expired	low	
28	Jefferson Thoma	10/6/2005 17:06:3	10/6/2005 17:06:3	Apnea for 60sec.	expired	high	confirm
29	Jefferson Thoma	10/6/2005 17:06:1	10/6/2005 17:06:2	Bradycardia (0/min) for 48sec.	expired	medium	confirm

Slika 23: Zgodovina alarmov

Pregledujemo lahko zgodovino alarmov za posameznega bolnika ali za vse bolnike hkrati. Slika 23 prikazuje primer okna za vse bolnike hkrati. V obeh primerih uporabimo isti gradnik, kateremu spremenimo način prikazovanja. Zgodovina alarmov posameznega bolnika je dostopna iz okvirja za prikaz trenutnih alarmov s klikom na gumb Alarms, medtem ko je zgodovina alarmov vseh bolnikov dostopna iz orodne vrstice osnovnega okna. Po 10. Nielsenovem principu smo zgradili minimalistično in uravnoteženo okno.

Alarme prikazujemo v obliki tabele z naslednjimi stolpci:

- ime bolnika
- čas začetka alarma
- čas konca alarma
- opis alarma
- status (aktiven, potečen)
- prioriteta (nizka, srednja, visoka)
- potrditev

V polju potrditev se za nepotrjene alarme, ki zahtevajo potrditev, nahaja gumb za potrditev alarma. Ko alarm potrdimo, gumb za potrditev izgine, alarm pa dobi status potrjen. V desnem zgornjem robu okna prikazujemo skupno število nepotrjenih alarmov. Ob potrditvi se vrednost avtomatično osveži. Ker tabela vsebuje večje število vrstic, velikost okna pa želimo omejiti, se na desnem robu okna nahaja drsnik za pomikanje po tabeli.

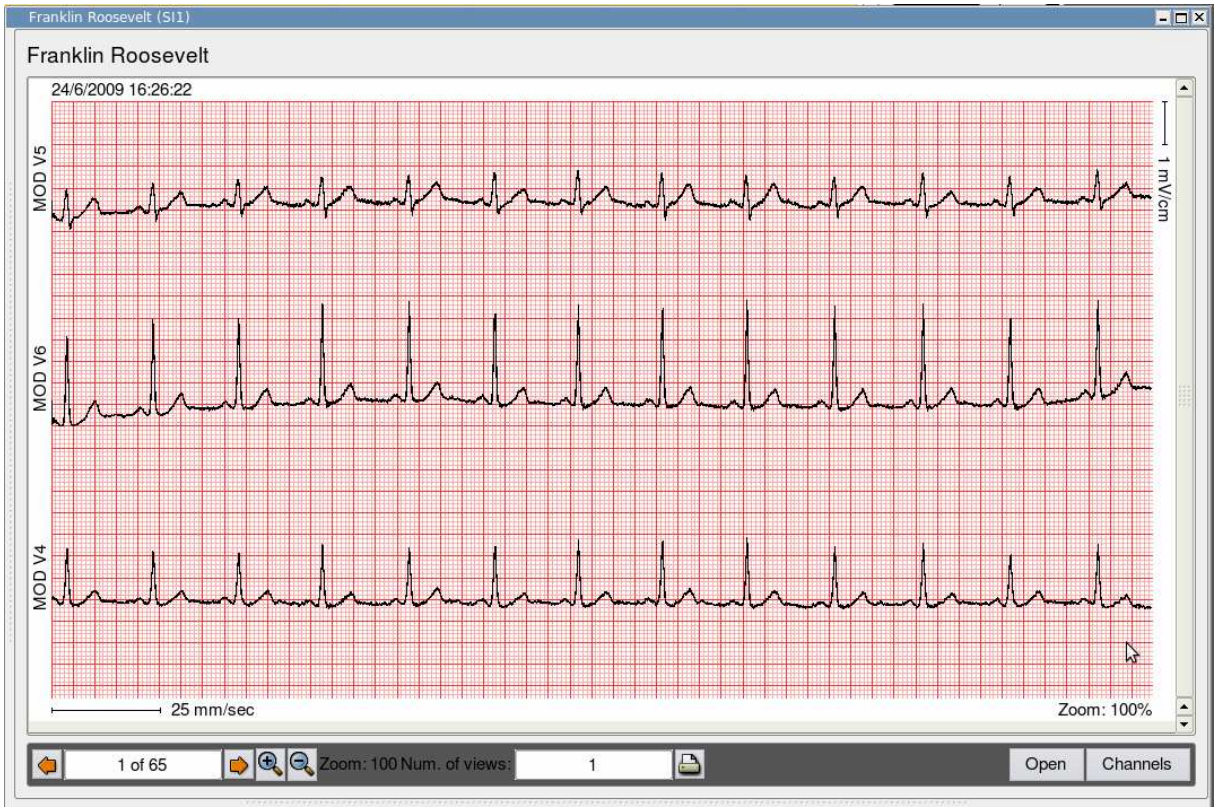
Ozadje vrstice alarma je pobarvano glede na prioriteto alarma. Barve so usklajene z barvami, ki so uporabljene v podobi za javljanje trenutnih alarmov. Zaradi boljše preglednosti so barve ozadja manj intenzivne kot pri aktivnih alarmih. Tako se vrstica alarma nizke prioritete obarva bledo cianovo, srednje prioritete bledo rumeno in barva alarma visoke prioritete bledo rdeče. S takšnim izborom barv smo želeli poudariti harmonijo in absolutno ločljivost.

Izpis zgodovine alarmov lahko časovno omejimo tako, da izberemo datum najstarejših alarmov, ki se še prikazujejo. Vsi alarmi, starejši od izbranega datuma, bodo skriti. Konsistentno s preostalimi datumski polji tudi tu uporabljamo krožni seznam za izbiro datuma in tako upoštevamo 2. Nielsenov princip o konsistentnosti. Tudi tukaj je komponenta uporabljena predvsem zato, da z izbiro namesto vnosa preprečimo možnost napake (7. Nielsenov princip).

Če je odprto okno za zgodovino alarmov posameznega bolnika, se v tabeli stolpec s podatki o bolniku skrije, ime bolnika pa se izpiše v naslovu okna. Na tak način dosegamo konsistentnost po 2. Nielenovem principu.

8.6 Zgodovina meritev

Zgodovina meritev je dostopna iz seznama bolnikov za vsakega bolnika posebej. V seznamu izberemo bolnika in kliknemo gumb History, da se nam odpre osnovno okno za pretekle meritve. Odpre se prazno okno za izris pretekle meritve. Slika 24 prikazuje okno z že izbrano preteklo meritvijo.



Slika 24: Izris pretekle meritve

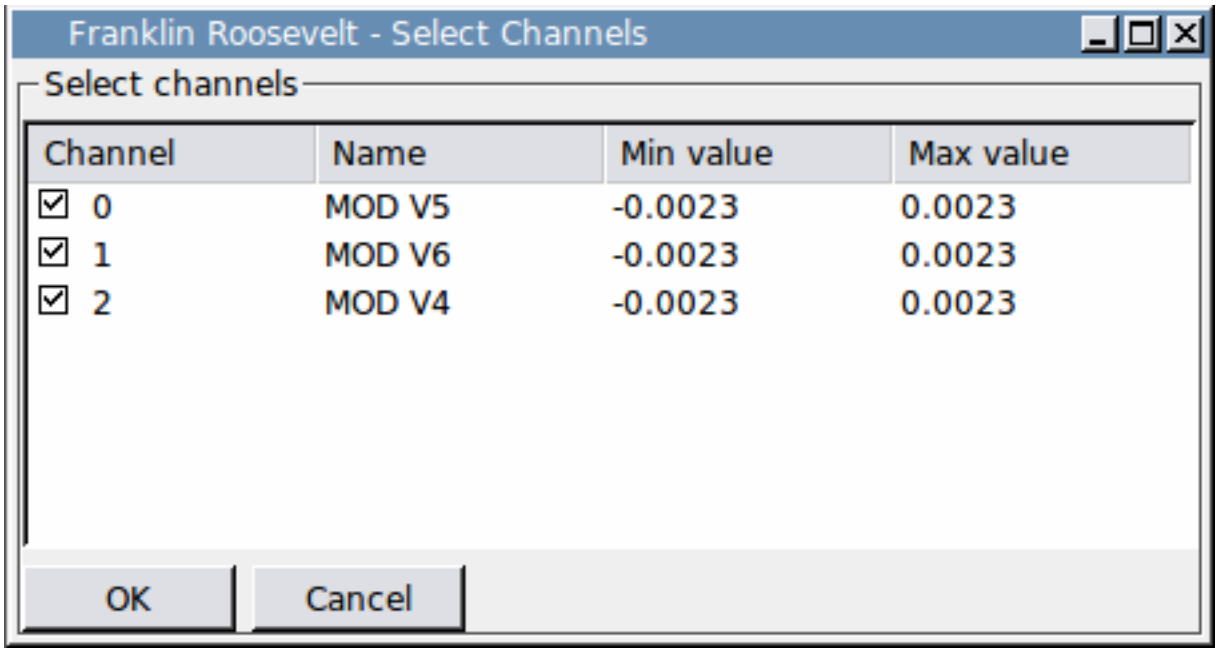
Okno je sestavljeno iz imena bolnika na vrhu, grafa meritve v sredini in navigacijske orodne vrstice na dnu. Orodna vrstica je horizontalno orientirana. Vsebuje komponento za pomikanje med stranmi meritve, gumba za prilagajanje velikosti izrisane krivulje, gumb za tiskanje, gumb za izbiro prikazanih krivulj in gumb za izbiro pretekle meritve. Za gumbe so uporabljene standardne ikone, s čimer zagotavljamo podobnost z drugimi produkti in na tak način upoštevamo konsistentnost po 2. Nielsonovem principu. Ikone predstavljajo metafore (npr. lupa – povečaj) in na tak način po 1. Nielsenovem principu prilagajajo okno realnemu svetu.

Navigacijska komponenta orodne vrstice je sestavljena iz gumbov za pomikanje naprej in nazaj ter okvirčka, ki izpisuje trenutno stran in število vseh strani po 5. Nielsenovem principu o vidljivosti stanja med navigacijo.

Izrisane krivulje lahko dinamično povečamo ali pomanjšamo za boljšo preglednost in prilagajanje vidnega časovnega intervala. To naredimo s pomočjo gumbov + in – orodne vrstice. Število strani se ob spremembi velikosti prikazanega grafa avtomatično osveži. Ozadje grafa je rdeča milimetrska mreža, krivulja pa je izrisana s črno barvo. Izbor barv je prilagojen temu, da elektrokardiograf v praksi najpogosteje tiskamo na papir z rdečo mrežo v ozadju. Lahko rečemo, da so izbrane standardne barve, kar ustreza 2. Nielsenovemu principu o konsistentnosti.

Okno ponuja možnost tiskanja izbrane pretekle meritve. S pritiskom na gumb za tiskanje se odpre sistemski menu za tiskanje z vsemi tiskalniki, ki so na voljo.

Gumb Channels odpre dialog za izbiro krivulje (Slika 25), ki jo želimo prikazovati v grafu. Privzeto so izbrani vsi trije odvodi.

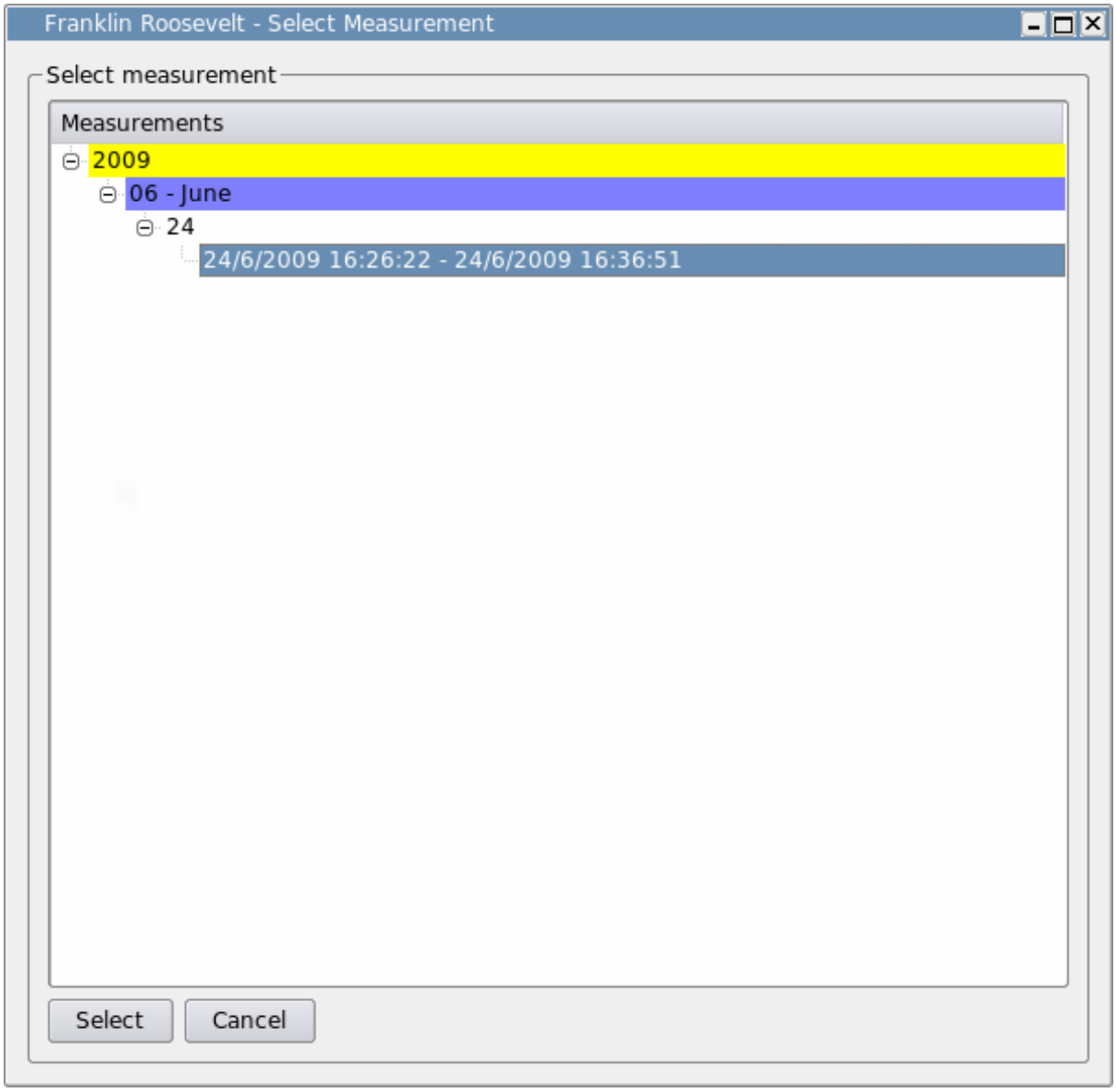


Slika 25: Dialog za izbiro krivulj za prikaz pretekle meritve

Z uporabo stikal za izbiro onemogočimo možnosti napak v skladu s 7. Nielsenovim principom. Gumba OK in Cancel sta standardna tudi v drugih pogosto uporabljanih aplikacijah, torej gre za konsistentnost po 2. Nielsenovem principu. S klikom na gumb OK uporabnik potrdi svojo izbiro, s Cancel pa jo prekliče. Potek akcij je definiran v skladu s 4.

Nielsenovim principom, ki govori o tem, kako zagotovimo nadzor uporabnika nad vmesnikom.

S klikom na gumb Open se odpre dialog s seznamom preteklih meritev, iz katerega izberemo meritev za prikaz. Slika 26 prikazuje primer takšnega dialoga.



Slika 26: Dialog za izbiro pretekle meritve

Pretekle meritve so razporejene po času začetka meritve. Za boljšo preglednost so meritve grupirane po letu, mesecu in dnevu. Z različnimi barvami ozadij smo želeli olajšati preglednost. Tako je ozadje leta obarvano rumeno, ozadje meseca pa modro. Najbrž uporaba barv v tej komponenti ni upravičena ali pa sama izbira barv ni najbolj posrečena, saj je

namesto ločljivosti bolj izraženo vzbujanje pozornosti, ki tukaj ni potrebna. Izbiro meritve potrdimo z gumbom Select ali prekličem z gumbom Cancel po 4. Nielsenovem principu, ki govori o uporabnikovemu nadzoru. Če smo dialog odprli po nesreči, ga lahko brez spremembe že prikazane meritve zapremo z gumbom Cancel in tako preprečimo napake po 7. Nielsenovem principu.

8.7 Vrednotenje rezultatov dela

Grafični vmesnik smo razvijalci hevristično ovrednotili. Poleg tega je bil posredovan tudi v testiranje uporabnikov v dejanskem kliničnem okolju.

8.7.1 Hevristično vrednotenje

Primeri hevrističnega vrednotenja našega uporabniškega vmesnika:

- ✓ Jezik je enostaven in prilagojen kliničnim uporabnikom; samo osnovna pojasnila brez nepotrebnih opisov, ki bi odvračali pozornost od bistvenih informacij (1. Nielsenov princip: prilagodi se realnemu svetu)
- ✓ Metafore pri ikonah (lupa, tiskalnik, puščice naprej/nazaj) so jasne in nedvoumne – dobro (1. Nielsenov princip: prilagodi se realnemu svetu, 2. Nielsenov princip: zunanja konsistenčnost)
- ✘ Gumb »Modify patient« za shranjevanje sprememb ni standarden (2. Nielsenov princip: standardi platform – najmanjše presenečenje). Rešitev: boljše bi bilo napisati »Save«
- ✓ Rdeča uporabljena za resne alarme, zelena za potekajočo meritev (2. Nielsenov princip: najmanjše presenečenje, 5. Nielsenov princip: vidljivost statusa sistema)
- ✘ Manjka uporabniška dokumentacija (3. Nielsenov princip: pomoč in dokumentacija). Rešitev: dodati priročnik za uporabnike.
- ✓ Gumb »Cancel« na voljo povsod, kjer obstaja možnost, da uporabnik želi preklicati akcijo (4. Nielsenov princip: uporabnikov nadzor in svoboda, 7. Nielsenov princip: preprečevanje napak)
- ✘ Če je okno, ki javlja alarm, prekrito z drugim notranjim oknom, uporabnik sliši zvočni signal, vendar mora sam ugotoviti, od kod prihaja (5. Nielsenov princip: vidljivost statusa sistema). Rešitev: ob alarmu višje prioritete okno, ki javlja alarm, postane aktivno in se pomakne v ospredje.

- ✓ Privzeta konfiguracija za alarme (6. Nielsenov princip: privzete nastavitve, 8. Nielsenov princip: raje prepoznaj kot si zapomni, 4. Nielsenov princip: uporabnikov nadzor)

- ✓ Izbira spola, dodeljevanje merilnikov, vnašanje datumov: komponente za izbiro namesto vnosnih polj (7. Nielsenov princip: izogibanje napakam)

- ✗ Konfiguracija za alarme v izključno tekstovni obliki (7. Nielsenov princip: izogibanje napakam, 8. Nielsenov princip: raje prepoznaj kot si zapomni). Rešitev: posamezne kriterije nadomestili s komponentami za izbiro.

- ✗ Slabo oz. nekonstruktivno javljanje napak v primeru neveljavnega vnosa v vnosno polje (9. Nielsenov princip: javljanje napak). Rešitev: vpeljati prijazna sporočila, ki uporabnika obvestijo, kako odpraviti napako.

- ✓ Enostavno in pregledno osnovno okno, poravnave gradnikov, grupiranje z belimi presledki (10. Nielsenov princip: estetika in minimalistično načrtovanje).

8.7.2 Testiranje uporabnikov

Uporabniški vmesnik je bil dlje časa na testiranju v Splošni bolnišnici Dr. Franca Derganca Nova Gorica v Šempetru pri Gorici. Njihov sistem je obsegal pet brezžičnih dostopnih točk BlipNode, več deset brezžičnih merilnikov in strežnik, na katerem je tekla naša programska oprema. Prinesli smo jim vso potrebno strojno in programsko opremo. Pokazali smo jim, kako se opremo uporablja in jih prosili, naj jo poskušajo vključiti v svoje vsakdanje delo. Opremo smo pustili tam in pustili testne uporabnike, da jo uporabljajo. Ves čas smo jih bili na voljo za morebitna vprašanja. Po nekaj tednih uporabe smo jih vprašali za mnenje.

Zdravniško osebje je nudilo precej pozitiven odziv in koristne napotke za nadgradnjo. Zbirali smo njihove pripombe in zahteve. Ko je bil sklop zahtev implementiran, smo jim v testiranje ponudili novejšo različico sistema. Takšen proces se je ponovil približno petkrat.

Tako so nas naprimer seznanili, da je standarda hitrost izrisovanja elektrokardiografa 25mm/sekundo. Pred to pripombo je bila hitrost izrisovanja v našem vmesniku odvisna od povečave, kar smo na njihovo pobudo popravili.

Večje kritike je bila deležna neintuitivna orodna vrstica. Ikone vsebujejo preveč detajlne slike, ki ne odražajo funkcionalnosti, na katero kažejo. Uporabnik se mora zato zanesti na napis pod ikono, torej ikona sama ne služi namenu. Poleg tega ni smiselno prikazovati gumbov, ki

trenutno še niso v uporabi. Povezave do funkcionalnosti orodne vrstice bi morale obstajati tudi znotraj glavnega menuja osnovnega okna.

Dodajanje bolnikov in dostop do meritev so komentirali kot precej intuitiven, prikaz alarmov pa kot jasen in dovolj vpadljiv. Utripanje ozadja in zvočni signal ob proženju alarmov je bilo implementirano v eni izmed iteracij na pobudo testnih uporabnikov, ki so želeli čimvečjo opaznost alarmov.

9 Sklepne ugotovitve

Razvili smo grafični uporabniški vmesnik, ki omogoča spremljanje elektrokardiografskih signalov v realnem času. Omogoča hranjenje baze bolnikov in merilnikov ter dodeljevanje merilnikov bolnikom. V realnem času izrisuje elektrokardiografske krivulje za izmerjene vrednosti. V primeru kritičnih stanj javlja alarme. Omogoča pregled preteklih alarmov in izrisovanje ter tiskanje krivulj zaključenih preteklih meritev.

Moj splošni vtis je, da smo v iskanju idealnih barvnih kombinacij zelo učinkovito uporabili žive barve in utripanje za pritegnitev pozornosti ob javljanju alarmov. Na nekaterih drugih področjih, kjer smo barve uporabili z namenom boljše ločljivosti, pa morda nismo bili najbolj uspešni. Preveliko število uporabljenih barv namreč hitro da vtis prenasičenosti, zaradi česar ob več odprtih pisanih notranjih oknih hitro izgubimo občutek, katera informacija je bolj pomembna in katera manj. Po ponovnem razmisleku bi vse razločilne barve, ki niso povezane s prikazom alarmov, bodisi odstranila bodisi spremenila v blede manj intenzivne odtenke.

Grafični vmesnik je pregleden in uravnotežen, torej smo zadostili osnovnim kriterijem estetike in uporabnosti. Uporabljen je jasen in razumljiv jezik, primeren okolju, kjer se naj bi vmesnik uporabljal. Prikazani podatki so opremljeni z ustreznimi pojasnili. Pri tem smo se omejili samo na prikaz bistvenih pojasnil tam, kjer so ta potrebna, saj bi prevelika količina nepotrebnih pojasnil preusmerila pozornost stran od dejanskih informacij.

Uporabniškemu vmesniku trenutno manjka uporabniška dokumentacija. Navodilo za uporabo bi moral v kasnejši fazi razvoja pripraviti oddelek za pisanje dokumentacije, zato v opisani različici uporabniškega vmesnika še ne obstaja. Ker je na voljo precej omejen nabor funkcionalnosti, pri sami implementaciji pa smo se trudili za jasen in razumljiv vmesnik, lahko nov uporabnik tudi brez predznanja in navodil hitro prične z uporabo.

Glavni cilj, ki smo si ga ob začetku razvoja zadali, je bilo pregledno in ažurno izrisovanje meritev ter čimbolj opazno javljanje alarmov. Mislim, da smo oboje uspeli precej uspešno implementirati.

10 Smernice za nadaljno delo

Ob delu se nam je porodilo še veliko idej, s katerimi bi izboljšali učinkovitost in varnost uporabe grafičnega vmesnika. Oglejmo si nekatere izmed njih.

10.1 Avtentikacija in avtorizacija

V bolnišničnem okolju bi bila smiselna uvedba uporabniških računov. Ob zagonu grafičnega vmesnika bi se moral uporabnik prijaviti s svojim uporabniškim imenom in geslom. Če bi se pojavila potreba, bi lahko razdelili uporabnike v skupine z dovoljenim dostopom do smiselnih vsebinskih sklopov. Tako bi na primer ločili administrativno osebje, ki bi vnašalo podatke o bolnikih, od zdravniškega osebja, ki potrebuje predvsem dostop do meritev in alarmov. V tem primeru bi bila potrebna izdelava komponente za upravljanje z uporabniškimi računi.

10.2 Vmesnik za pametne telefone

Okolje QT mogoča izdelavo grafičnih vmesnikov za mobilne naprave. Trenutno poteka tudi prenos podpore za QT na operacijski sistem Android, ki je pogost tako na telefonih kot tudi na tabličnih računalnikih. S to tematiko se trenutno ukvarjata kar dva projekta: Necessitas Qt Suite for Android in android-lighthouse. Glede na splošno razširjenost pametnih telefonov (ang. smartphone) bi bilo zelo uporabno prenesti poenostavljeno različico grafičnega vmesnika v mobilno okolje. S tem bi bili ažurni podatki na voljo ob vsakem trenutku.

10.3 Arhiv bolnikov

Trenutni vmesnik dopušča dodajanje in brisanje bolnikov. Funkcija arhiviranja bi bolnika, ki je bil odpuščen iz bolnišnice, skrila iz aktivnega seznama. Bolniki iz arhiva bi bili enostavno dostopni, vsi njihovi podatki pa bi bili shranjeni. V primeru ponovnega sprejema v bolnišnico bi bolnika enostavno prestavili iz arhiva nazaj v aktivni seznam.

10.4 Branje kartice zdravstvenega zavarovanja

V teh časih si težko predstavljamo zares uporabno aplikacijo, ki ne zna brati kartice zdravstvenega zavarovanja in dostopati do podatkov iz baze ZZZS. V primeru povezave s sistemom ZZZS bolnikov ne bi bilo potrebno vnašati ročno, temveč bi njihove podatke lahko poiskali v bazi ZZZS. Vmesnik bi po potrebi lahko razširili še s pregledovalnikom kartotek bolnikov.

10.5 Razširjeni filtri za izpis zgodovine alarmov

Zgodovino alarmov bi za boljšo preglednost lahko dodatno filtrirali z definiranjem časovnega intervala za prikaz, prioritete alarma, iskanjem po sporočilu alarma ali imenu bolnika. Izpis bi lahko razdelili na posamezne strani, ki bi vsebovale omejeno število alarmov.

11 Literatura

[1] Bručan A. ... et al., VSE o srcu in žilah, Društvo za zdravje srca in ožilja Slovenije, Ljubljana, 1996

[2] Dalheimer K. L. , Qt vs. Java: A Comparison of Qt and Java for LargeScale, Industrial-Strength GUI Development, dostopno na: http://turing.iimas.unam.mx/~elena/PDI-Lic/qt-vs-java-whitepaper.pdf

[3] Jager F., Komunikacija človek računalnik – zapiski predavanj za študijsko leto 2010/11

[4] Jerše M., Srčni infarkt, Založba Centralnega zavoda za napredek gospodinjstva, Ljubljana, 1987

[5] Kacin B., Razvoj polavtomatskih grafičnih orodij za razvrščanje morfologij segmenta ST elektrokardiograma, FRI, Ljubljana, 2002

[6] Strobl M., Optimizacija postopkov za iskanje referenčnih točk srčnega cikla, FRI, Ljubljana, 2002

[7] (2011) Basic Electrophysiology ECG, dostopno na: http://electro-medicals.com/2011/03/basic-electrophysiology-ecg/

[8] (2011) C++ Reference: STL containers, dostopno na: http://www.cplusplus.com/reference/stl/

[9] (2011) DIS računalniški slovarček, dostopno na: http://dis-slovarcek.ijs.si/

[10] (2011) islovar, Slovar informatike, dostopno na: http://www.islovar.org/iskanje_enostavno.asp?reset=true

[11] (2011) Texas heart institute, Heart anatomy, dostopno na http://texasheart.org/HIC/Anatomy/anatomy2.cfm

[12] (2006) The Alan E. Lindsay ECG Learning Center, The Standard 12 Lead ECG, dostopno na: http://library.med.utah.edu/kw/ecg/ecg_outline/Lesson1/index.html

[13] (2011) Wikipedia, the free encyclopedia, poglavja Heart, Electrical conduction system of the heart, Electrocardiography, dostopno na: http://en.wikipedia.org/

[14] (1999) Yonat S., Smart Pointers - What, Why, Which?, dostopno na: http://ootips.org/yonat/4dev/smart-pointers.html