

AARHUS SCHOOL OF ENGINEERING

TELEMEDICINSK ROBOTSTYRET ULTRALYDSSKANNING

PROJEKT NR: 15138

BACHELORPROJEKT EFTERÅR 2015

Udviklingsdokumentation

Forfattere:

Rasmus Holm Laursen [201270843]

Andreas Lauridsen [10571]

Karsten Michaelsen [11007]

Vejleder:

Michael Alrøe

15. december 2015

Indholdsfortegnelse

Kapitel 1 Indledning	1
1.1 Defination af begreber	1
Kapitel 2 Systemarkitektur	2
2.1 Systemoversigt	2
2.2 Udviklingsmiljø	3
2.3 Systemets interfaces	4
2.3.1 Geomagic Touch	4
2.3.2 UR10	6
2.3.3 Ultralydsscanner	12
2.3.4 Webcam	14
2.4 Softwarearkitektur	16
Kapitel 3 Systemdesign	17
3.1 Pakkediagram	17
3.2 Klassediagram	18
3.3 Sekvensdiagrammer	21
3.3.1 Etabler forbindelse	22
3.3.2 Opdater hovedvindue	31
3.3.3 Konfigurer UR	36
3.3.4 Konfigurer ultralydsscanner	36
3.3.5 Positioner UR	37
3.3.6 Force Feedback	39
3.3.7 URprogram	40
3.4 Detaljeret specifikation af klassediagrammer	41
Kapitel 4 Unittests	50
4.1 Kommunikation med UR10	50
4.2 Kommunikation med Geomagic Touch	53
4.3 Fjernskrivebord	54
4.4 Videofeed	54

Indledning 1

1.1 Defination af begreber

Begreber/Forkortelse	Betydning
GMT	GeoMagic Touch
UR10	Universal Robots UR10
RE	Robotenhed er den samlede enhed ved robotten. Robot, Ultralydsscanner og Videokamera
SE	Styringsenhed er den samlede enhed ved kontrolsiden. GeoMagic Touch og PC applikation.
[]	Repræsentere en knap
HDAPI	Haptic Device API
TCP	Tool Center Point på UR10
PDU	Protocol Data Unit
RDP	Remote Desktop Protocol
RTSP	Real Time Streaming Protocol
RTP	Real Time Protocol

Tabel 1.1. Defination af begreber

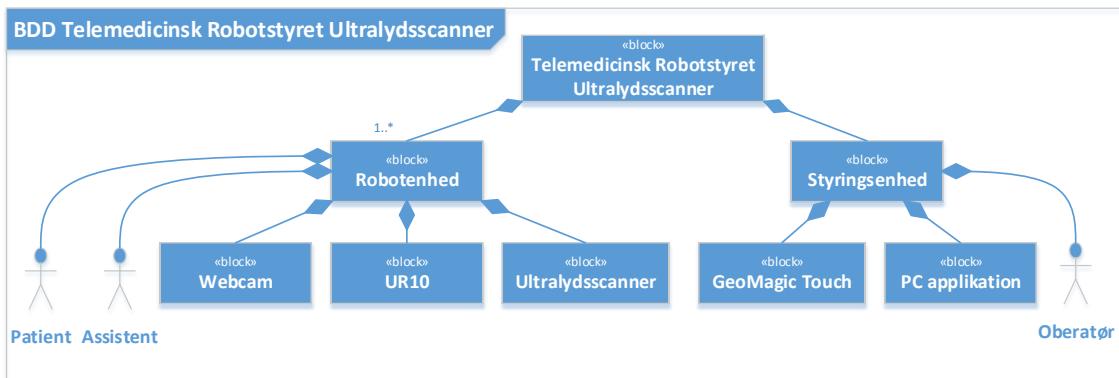
Systemarkitektur 2

2.1 Systemoversigt

Systemet Telemedicinsk Robotstyret Ultralydsscanner består af to separate enheder, der kommunikerer via et netværk, og tre aktører der interagerer med hver deres enhed. Robotenheden er den enhed der udfører selve ultralydsscanningen af patienten vha. en ultralydsscanner, der er monteret på Universal Robot UR10 (UR10). Under scanningen bliver der filmet med et webcam og assistenten er til stede for at assistere patienten og operatøren såfremt det bliver nødvendigt. Assistentens opgave er også at opstarte robotenheden før brug. Systemet kan bestå af flere robotenheder på en gang, men der kan kun kommunikeres med én ad gangen.

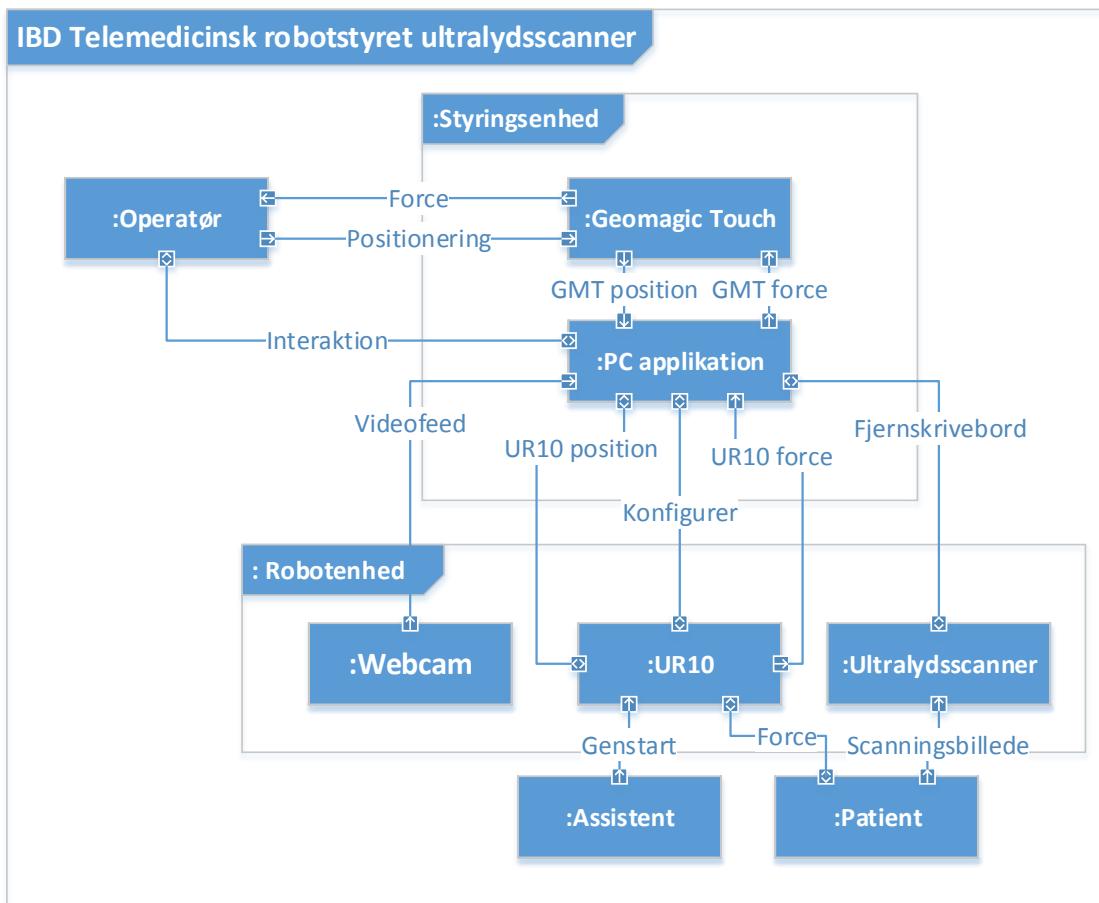
Robotenheden styres fra styringsenheden af en operatør, der interagerer med en PC applikation og et GeoMagic Touch (GMT) joystick. Applikationen køres på en pc, hvor GMT joysticket er tilsluttet og herfra kan operatøren betjene UR10'en og orientere sig om dens bevægelser og selve scanningen vha. skærbilleder på PC applikation.

Oversigten over systemets blokke og aktører kan ses på blokdiagrammet i figur 2.1.



Figur 2.1. BDD for Telemedicinsk robotstyret ultralydsscanner.

Figur 2.2 viser den interne sammenhængen mellem blokkene i systemet. PC applikation er ansvarlig for kommunikation og håndtering af forbindelserne mellem blokkene i Styringsenhed og Robotenhed. Diagrammet viser dermed systemets interfaces mellem PC applikation og de andre blokke.



Figur 2.2. IBD for Telemedicinsk Robotstyret Ultralydsscanner.

2.2 Udviklingsmiljø

Der er i nedenstående tabel 2.1 opstillet de rammer der er for udviklingen af Telemedicinsk Robotstyret Ultralydsscanner. Foruden nedenstående er der anvendt en række eksterne biblioteker i forbindelse med Fjernskrivebord og Videofeed, disse er beskrevet nærmere i afsnit for pågældende enhed.

Emne	Version
Windows Visual Studio Ultimate 2013	12.0.30501.00 Update 2
Windows styresystem	8.1
PC-Applikation: WPF Program (.Net Framework)	4.5.51650
VideoFeedServer: C#-Consol (.Net Framework)	4.5.51650
RemoteDeskTopServer: C#-Consol (.Net Framework)	4.5.51650
GeoMagic Touch: Driver	v3.4 GTDD v2015.7.31
UR10	Se tabel 2.10

Tabel 2.1: Oversigt over udviklingsmiljø for Telemedicinsk Robotstyret Ultralydsscanner

2.3 Systemets interfaces

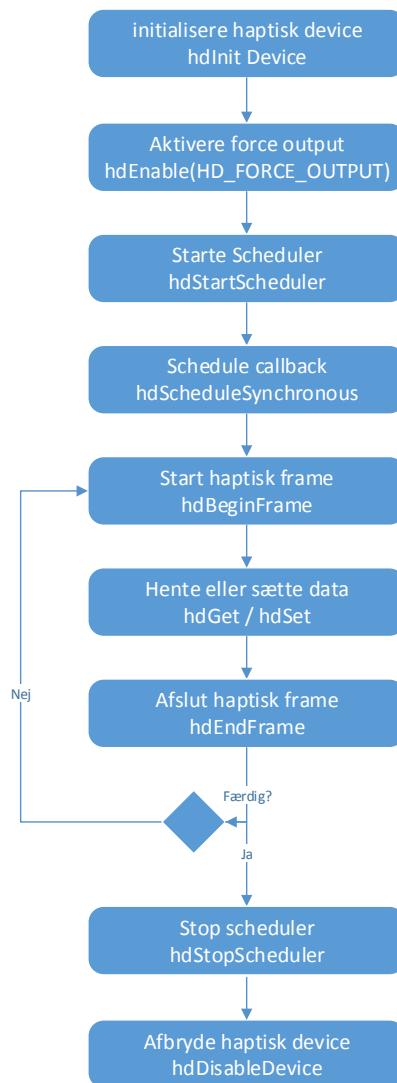
Systemet består af fire interfaces mellem PC applikation, der udvikles i C# og GeoMagic Touch, UR10, Ultralydsscanner samt Webcam. PC applikation fungerer som en client, der forbinder til fire servere, en for hver ekstern enhed, hvor den kan sende og/eller modtage data.

De fire interfaces ser ud som følgende:

2.3.1 Geomagic Touch

Kommunikation med Geomagic Touch

Kommunikationen med GMT foregår gennem en del af Open Haptic Toolkit, som er et redskab udviklet af GeoMagic til udvikling af simple haptiske applikationer. Der anvendes Haptic Device API (HDAPI) til at kommunikere med GMT. HDAPI har følgende flowdiagram for kommunikation med GMT, hvor der i hver led ses en beskrivende tekst og syntaksen, der er krævet for at udføre handlingen:



Figur 2.3. Flowdiagram for kommunikation med Geomagic Touch HDAPI

Flowdiagrammet i figur 2.3 viser hvordan håndteringen af kommunikation med GMT HD API forløber. Først initialiseres det haptiske device, hvilket leverer en handle til det tilsluttede device. Herefter aktiveres force output, som tænder alle motorer på GMT. Der startes en Scheduler som håndterer en højprioritets tråd, der kører med 1000 Hz. For at kommunikere med denne tråd arrangeres der et callback som starter et nyt frame, hvor det er muligt at hente eller sætte data for GMT, hvorefter framet afsluttes. Hvis der ønskes at kommunikere yderligere startes et ny frame ellers stoppes Scheduleren og der afbrydes forbindelse til devicet.

Værdier

Som beskrevet i flowdiagrammet på figur 2.3 er det muligt at hente data fra GMT. I nedenstående tabel er der beskrevet værdier der mulige at hente, deres datatype og betydning.

Navn	Antal værdier	Enhed	Betydning
HD_CURRENT_FORCE	3	N	Force i kartesisk koordinat vektor.
HD_CURRENT_POSITION	3	mm	X, Y og Z-Koordinat
HD_CURRENT_GIMBAL_ANGLES	3	radian	Rotation i gimbal
HD_CURRENT_JOINT_ANGLES	3	radian	Rotation i joints
HD_CURRENT_BUTTONS	1	Unitless	Knaptryk der beskriver hvor mange knapper der er trykket ned. Værdien er 1 hvis det den forreste knap på GMT, 2 hvis det den bagerste knap på GMT og 3 hvis det er begge knapper der er trykket ned.

Tabel 2.2: Specifikationer af GMT værdier som kan hentes.

I Bilag 4 side 56-59 kan der findes en fuld liste over værdier der kan hentes.

For at hente disse informationer fra GMT anvendes følgende syntaks: `hdGetDoublev(HDenum pname, HDdouble *params)`. For yderlige syntaks se Bilag 4 side 7.

Det er igennem kommunikationen med GMT muligt at indstille forceen med en specifik værdi som er beskrevet i tabellen nedenfor.

Navn	Antal værdier	Enhed	Betydning

HD_CURRENT_FORCE	3	N	Forcen består x(+ er til højre), y(+ er op) og z(+ er udad) kraft som beskriver kraften i hver retning i et kartesisk koordinatsystem.
------------------	---	---	--

Tabel 2.3: Specifikationer af GeoMagic Touch værdier som kan sættes.

I OpenHaptics_RefGuide side 60 kan der findes en fuld liste hvorledes det er muligt at indstille forceen.

For at indstille forceen kan følgende syntaks anvendes: `hdSetFloatv(HDenum pname, const HDfloat *params)`. For yderlig information se Bilag 4 side 10.

Fejlhåndtering

HDAPI indeholder et fejlhåndteringssystem hvor hver error er opbygget med følgende indhold:

Indhold	Betydning
errorCode	En error code fra definitionsfilen.
internalErrorCode	En error der er genereret af kald til devicet.
hHD	Et id på det device der var skyld i fejlen.

For yderlig information se Bilag 3 side 96-97.

Kommunikation mellem PC applikation og Geomagic Touch

HDAPI er udviklet i C++ og derfor skal koden der anvender denne API også skrives i C++. PC applikation bliver udviklet i C# og det vil derved være nødvendigt at have kald fra managed code til unmanaged code, som er illustreret på figuren nedenfor.



Figur 2.4. Kald fra PC applikation til Geomagic Touch

2.3.2 UR10

Styring af og kommunikation med UR10 håndteres på flere måder. Selve UR10 programmeres i sproget URScript, som bruges til at styre UR10 på script-niveau. I URScriptet er der flere indbyggede metoder, som bruges til positionering af UR10. Dette gøres ved at UR10 ses fra dens Tool Center Point(TCP), hvor den kan positioneres i sit Tool-space ift. basen, og den kraft der påvirker dens TCP kan registreres. Til positionering af TCP benyttes der af følgende funktion:

moveL(pose, a=1.2, v=0.3)	
Parameter	Beskrivelse
pose	Ønskede position
a	Tool acceleration (m/s ²)
v	Tool hastighed (m/s)

Tabel 2.5. Funktion til at positionere UR10'en (Bilag 11 s. 13)

Funktionen *moveL* bevæger TCP lineært i Tool-space hen til den ønskede *pose*. En *pose* er givet ved $p[x,y,z,Rx,Ry,Rz]$, hvor x,y,z er positionen af TCP og Rx,Ry,Rz er orienteringen af TCP i en axis-angle notation(Bilag 11 s. 5). Værdierne for acceleration og hastighed angivet i tabel 2.5 er standardværdier såfremt andet ikke er angivet.

Funktionen **get_tcp_force()** returnerer kraftdrejningen i TCP, som er beregnet på baggrund af den fejl, der måles i moment på hver joints for at holde den ønskede bane, og det forventede moment. Kraften returneres som en *pose* og er målt i Newtons og Newtons/Radian.

Til at konfigurere UR10 benyttes der yderligere to funktioner, som kan ses i følgende tabeller:

set_payload(m)	
Parameter	Beskrivelse
m	masse i kg

Tabel 2.6. Funktion til at definere vægt på ultralydsprobe på TCP (Bilag 11 s. 19)

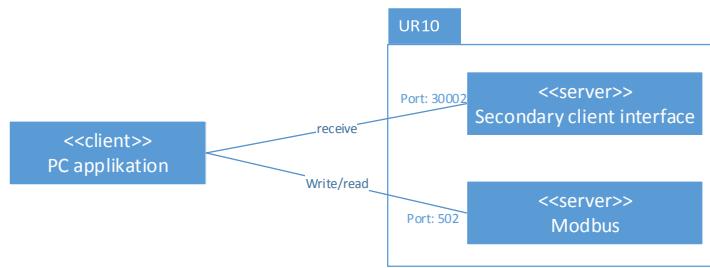
Funktion i tabel 2.6 anvendes til at kunne angive vægten på den ultralydsprobe der er monteret på UR10, som den kan bruge i beregninger af sine bevægelser.

set_tcp(pose)	
Parameter	Beskrivelse
pose	<i>pose</i> der beskriver transformationen

Tabel 2.7. Funktion til at definere transformationen af TCP (Bilag 11 s. 19)

Funktion i tabel 2.7 anvendes til at transformere TCP baseret på den ultralydsprobe der monteres på UR10, således dens omdrejningspunkt defineres til spidsen af ultralydsproben.

Ved positionering af UR10 og konfigurering heraf, benyttes der to porte på dens server, når der kommunikeres fra PC applikation. Kommunikationen på de to porte har hver deres struktur, som er beskrevet nedenfor.



Figur 2.5. Diagram over PC applikationens forbindelse til de to porte på UR10

UR10 har specifikationerne jvf. tabel 2.8 omkring den installerede software. Det vigtigste omkring disse specifikationer er, at det er version 3.1, hvilket har afgørende betydning for det senere beskrevet data stream.

User interface	Polyscope 3.1.18024
Robot controller	URControl 3.1.18024
Last updated	Jul 07 2015
Hostname	ur-2014300294
s/n	2014300294

Tabel 2.8. Specifikationer af anvendt UR10 version

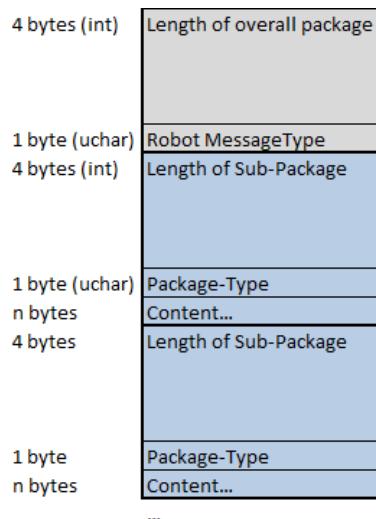
Secondary client interface

UR10 controlleren leverer hele tiden data omkring robottens version og tilstand med bl.a. position og meget andet på port 30002. På dette data stream sendes der to typer beskeder/pakker.

Pakke nr.	Pakketype
16	Robottilstand
20	Versionsbesked

Tabel 2.9. Oversigt over de to beskedtyper modtaget på port 30002

De pakker der modtages følger en bestemt struktur, hvilket kan ses på figur 2.6. Der modtages en overordnet pakke, der kan indeholde ingen eller flere underpakker. Som figuren viser afgør de første fem bytes, hvor stor den modtaget pakke er og hvilken af pakketyperne i tabel 2.9, der er modtaget. Længden af pakkerne der er angivet i de respektive fire bytes er inklusiv sig selv.



Figur 2.6. Datastruktur over data stream modtaget fra port 30002 (Bilag 6, worksheet DataStream-FromURController)

Når der oprettes forbindelse til UR10, vil den første pakke der modtages på streamet altid være versionsbeskeden, pakke nr. 20, hvilket informerer om den software der er installeret og hvornår den sidst er blevet opdateret. Disse oplysninger bruges sammen med ip-adressen til at kontrollere at det er den rigtige robot der forbindes til, når der skal testes en ny forbindelse. Pakke nr. 20 ser ud som følgende:

Størrelse i bytes	Datatype	Betydning
4	integer	Pakkens længde
1	unsigned char	Pakkens type
8	uint64	Timestamp
1	char	Kilde
1	char	Beskedens type
1	char	Projektnavn størrelse
variabel	char array	Projektnavn
1	unsigned char	Major version
1	unsigned char	Minor version
1	int	SVN Revision
Resterende bytes	char array	Byggedato

Tabel 2.10: Indholdet af versionsbeskeden, som er den første pakke der modtages på streamet (pakke nr. 20)

De efterfølgende pakker på streamet er af pakketypen Robottilstand (pakke nr. 16), som består af 10 underpakker. Ud af disse 10 pakker anvendes der 3 specifikke pakker ifm. positionering af UR10. De tre pakker er:

- Pakke nr. 0 - Robot Mode Data
- Pakke nr. 3 - Masterboard data
- Pakke nr. 4 - Cartesian info

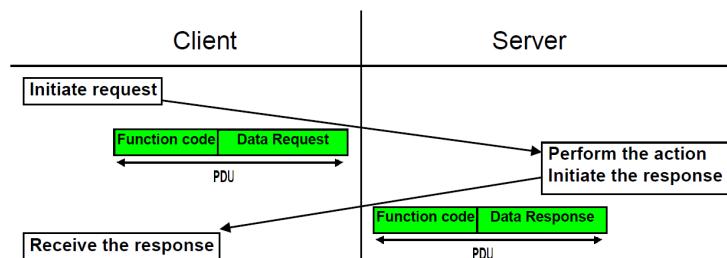
Robot mode data leverer informationer omkring UR10's tilstand, om den er tændt, kørende osv. Sammen med oplysninger om UR10's sikkerhedstilstand fra pakken masterboard data, bruges disse til at overvåge om UR10 er i den rette tilstand til at kunne positionere. Positioneringen sker på baggrund af UR10's aktuelle position, som kan læses ud fra pakken cartesian info. Oplysningerne heri viser UR10's position i dens eget koordinatsystem og ser ud som i tabel 2.11. For analyse af alle pakkerne der modtages på streamen, henvises der til Bilag 6, hvor der findes yderligere information om de resterende underpakker der modtages.

Størrelse i bytes	Datatype	Betydning
4	integer	Pakkens længde
1	unsigned char	Pakkens type
8	double	X-koordinat i meter
8	double	Y-koordinat i meter
8	double	Z-koordinat i meter
8	double	Rx-koordinat i radianer
8	double	Ry-koordinat i radianer
8	double	Rz-koordinat i radianer
8	double	TCPOffset-X i meter
8	double	TCPOffset-Y i meter
8	double	TCPOffset-Z i meter
8	double	TCPOffset-Rx i radianer
8	double	TCPOffset-Ry i radianer
8	double	TCPOffset-Rz i radianer

Tabel 2.11: Indholdet af cartesian info (underpakke nr. 4), som indeholder information om UR10's position

Modbus

Modbusserveren er på port 502 og der kommunikeres via en TCP-forbindelse. Modbusen består af en række registre, der læses og skrives til vha. requests. Modbus protokollen definerer en Protocol Data Unit (PDU), der gør at ved en request skal der oprettes en header, der indeholder en funktionskode og oplysninger om de registre der ønskes at skrive til eller læse fra. Udvekslingen af en request kan ses på figur 2.7.



Figur 2.7. Modbus transaktion(Bilag 10 s. 4).

Der anvendes to funktionskoder til kommunikation med modbusserveren, som er beskrevet

i tabel 2.12.

Funktionskode	Definition	Beskrivelse
04 (0x04)	Read input register	Læser 1-125 sammenhængende registre, hvor PDU'en definerer start registret og antallet af registre der læses fra. Responsen er pakket i to bytes pr. register, hvor første byte er det mest betydende, og andet byte er det mindst betydende (Bilag 10 s. 16)
16 (0x10)	Write multiple register	Skriver i 1-123 sammenhængende registre, hvor PDU'en definerer start registret og antallet af registre der skrives til. Data er pakket i to bytes pr. register (Bilag 10 s. 30).

Tabel 2.12: Funktionskoder og deres definition.

Kommunikationen mellem PC applikation og modbusserveren fungerer ved at der anvendes et general purpose 16 bits register på modbusen, som ligger fra adresse 128-255. Her skriver PC applikation UR's ønskede position ind, som et program på UR10 læser koordinaterne fra. Ligeledes skriver programmet på UR10 den aktuelle force ind i registeret, som PC applikation kan læse fra. Derudover er der afsat registre til diverse konfigurationer på UR10'en. En oversigt kan ses i tabel 2.13.

Adresse	Formål	Enhed
128	x-koordinat	meter
129	y-koordinat	meter
130	z-koordinat	meter
131	Rx-koordinat	radianer
132	Ry-koordinat	radianer
133	Rz-koordinat	radianer
137	x-force	Newton
138	y-force	Newton
139	z-force	Newton
143	Ny konfiguration	0/1 (false/true)
146	z-offset	meter
150	Payload	kg
151	Acceleration	m/s ²
152	Hastighed	m/s

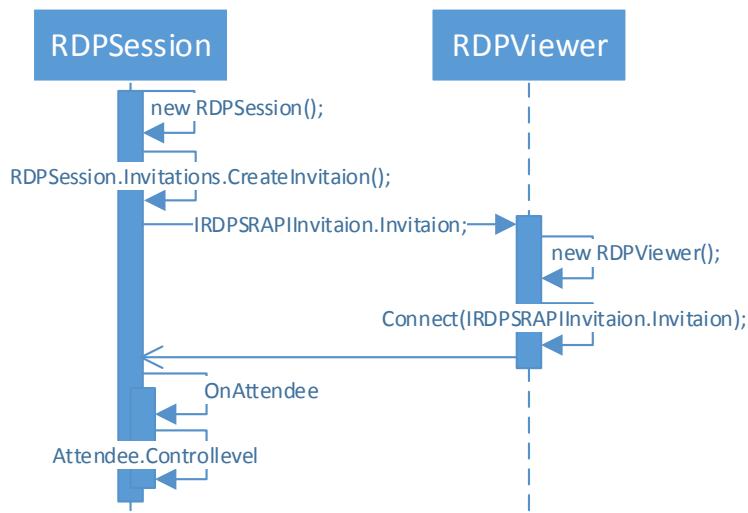
Tabel 2.13: Benyttede registre i general purpose register på modbusserveren og deres formål. For fuld oversigt se Bilag 8.

2.3.3 Ultralydsscanner

Kommunikationen mellem PC applikation og Ultralydsscanner forgår via et fjernskrivebord. Det betyder at der er en part der skal dele et skrivebord og en anden part der skal tilgå dette. Denne opdeling opnås igennem Windows Desktop Sharing, som anvender Remote Desktop Protocol(RDP). For at dele et skrivebord anvendes en RDPSession og for at fremvise denne anvendes RDPViewer.

Sammenhæng mellem RDPSession og RDPViewer

På nedenstående sekvensdiagram ses hvorledes RDPSession opretter en ConnectionString som bliver anvendt af RDPViewer til at forbinde til RDPSession.



Figur 2.8. Sekvensdiagram for kommunikationen mellem RDPSession og RDPViewer

RDPSession

For at anvende en RDPSession initialiseres en ny RDPSession med følgende syntaks:

`new RDPSession();`

som implementerer et IRDPSRAPISharingSession hvor det er muligt at oprette en *Invitations* som implementerer et IRDPSRAPILInvitationManager interface. Det er via dette interface at det er muligt at oprette en IRDPSRAPILInvitation, som ses i nedenstående syntaks:

`RDPSession.Invitations.GetConnectionString(String AuthString, String GroupName, String Password, int AttendeeLimit).`

På baggrund af ovenstående IRDPSRAPILInvitation er det muligt at hente ConnectionString ved følgende syntaks:

`IRDPSRAPILInvitation.ConnectionString`

Herefter registreres en ny bruger af RDPSessionen, hvor der er mulighed for at indstille, hvilket niveau brugeren skal have mulighed for at tilgå RDPSessionen med. Dette gøres ved at oprette et event som registrerer en ny *Attendee*, hvor der kan indstilles interaktion niveau via syntaksen:

`Attendee.Controllevel`

For yderlig information vedrørende RDPSession se [https://msdn.microsoft.com/en-us/library/windows/desktop/aa373307\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa373307(v=vs.85).aspx)

RDPViewer

For at fremvise et fjernskrivebord skal der oprettes en ny RDPViewer som skal tilslutte sig en RDPSession. Dette gøres ved at oprette en ActiveX komponent som implementerer en RDPViewer med interfacet IRDPSRAPIViewer.

Via dette interface er det muligt at forbinde til RDPSessionen via syntaksen:

RDPViewer.Connect(string ConnectionString, string Name, string Password) *ConnectionString* som anvendes til at forbinde kommer fra RDPSession.

For yderlig information vedrørende *RDPViewer* se [https://msdn.microsoft.com/en-us/library/windows/desktop/aa373307\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa373307(v=vs.85).aspx)

API

Til fjernskrivebord benyttes Windows Sharing API. Denne API består af to biblioteker hvor AxRDPCOMAPILib anvendes til en RDPViewer og rdpcomplib til at danne en RDPSession. Disse to biblioteker kan tilføjes via Visual Studio ved at følge dette link: <http://sandaruwmp.blogspot.dk/2014/05/remote-desktop-application-with-rdp.html>.

2.3.4 Webcam

Webcam integrationen forgår ved et Client-Server forhold mellem webcam(server) og videoafspiller(client). Serveren skal etablere et videofeed ved brug af Real Time Streaming Protocol (RTSP), som videoafspilleren kan åbne og afspille.



Figur 2.9. Overordnede elementer i Webcam integration

RTSP er udelukkende til kontrol af den pågældende streaming session, selve data overførelsen foregår via Real-Time Transport Protocol (RTP). Serveren er ansvarlig for at "fange"video fra et webcam, derefter encode videoen og sende den via RTP-pakker. Opbygning af en RTP-pakkes header kan ses nedenfor.

RTP packet header							
Bit offset ^[b]	0–1	2	3	4–7	8	9–15	16–31
0	Version	P	X	CC	M	PT	Sequence number
32	Timestamp						
64	SSRC identifier						
96	CSRC identifiers ...						
96+32×CC	Profile-specific extension header ID				Extension header length		
128+32×CC	Extension header ...						

Figur 2.10. RTP-pakkes headers opbygning. For yderlig information se https://en.wikipedia.org/wiki/Real-time_Transport_Protocol.

På baggrund af RTP-Headeren vil afspilleren decode videoen i henhold til payloaden typen (PT) defineret i RTP-header og afspille videoen i afspilningsvinduet. For yderligere detaljer omkring RTP-pakken se <https://tools.ietf.org/html/rfc3550>. RTP-pakken styres automatisk af det bibliotek, der bliver benyttet til implementering af serveren og ligeledes vil den valgte videoafspiller automatisk kunne decode den modtagede RTP-pakke.

VideoFeedServer

Videofeedserveren benytter et bibliotek fra Ozeki. Serveren er opbygget på en sådan måde at det er muligt med tilkobling af flere webcams, samt den til streaming benytter protokollerne RTSP og RTP. Serveren har indbygget encoders til at "fange"video fra webcam.

Se link for yderligere dokumentation af bibliotek anvendt: http://www.camera-sdk.com/p_13-download-onvif-standard-ozeki-camera-sdk-for-webcam-and-ip-camera-developments-onvif.html
Version: v1.2.0

Videoafspiller

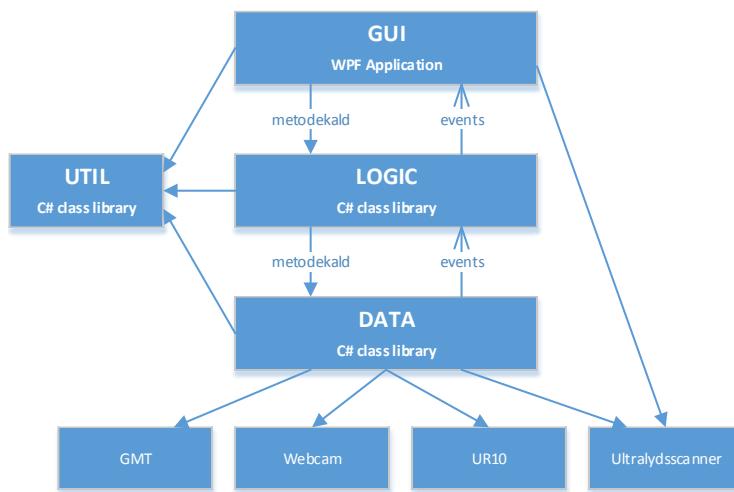
Videoafspilleren bygger på opensource fra Videolan's VLCplayer, hvilket giver afspilleren mulighed for at arbejde med en lang række forskellige videotyper. VLCPlayer har indbygget en decoder til at omsætte det modtaget stream til faktiske billeder og kan tolke pakker fra protokollerne RTSP og RTP. Derfor er der benyttet et bibliotek (nVLC) som er beregnet til implementering af VLCplayer komponenter i .NET frameworkt.

Se link for yderligere dokumentation af det anvendte bibliotek:

<http://www.codeproject.com/Articles/109639/nVLC> Version: 3 Oct 2013

2.4 Softwarearkitektur

PC applikation arkitektur følger 3-lags-modellen, som det ses i figur 2.11. Da systemet består både af en præsentation for brugeren, logisk håndtering til bl.a. beregning af forcefeedback og positionering og kommunikation med eksterne enheder (se tabel 2.14 for yderligere beskrivelse.), anvendes arkitekturen for at fordele ansvaret af de forskellige opgaver. Som figur 2.11 viser, er referencen mellem de tre lag således at der udføres kald nedad og skal en process i et underliggende lag informere opad, sker det vha. events. En enkel afvigelse herfra er når RDVieweren skal forbinde til RDPSessionen, sker dette direkte fra RDVieweren i GUI-laget.



Figur 2.11. Oversigt over PC applikationens arkitektur.

GUI	Håndterer operatørens interaktion med diverse vinduer i systemet, når der skal oprettes forbindelse, oprettes ny scanningsenhed og hovedvinduet. I hovedvinduet interagerer operatøren med de to viewere til fjernskrivebord og webcam og yderligere konfigurationspanelet, når UR10 skal konfigureres.
LOGIC	Håndterer den logiske del af PC applikationen. Når der skal etableres forbindelse, skal der køres en sekvens der sørger for at datalaget opretter forbindelse til diverse objekter. Står for at udføre testen og godkendelsen af ny RE. Når forbindelsen er oprettet skal LOGIC håndtere beregningen af forcefeedback og positionering af UR10'eren.
DATA	Håndterer forbindelserne til diverse enheder udenfor PC applikationen. Når der er oprettet forbindelse er det herigennem der kommunikeres med enhederne, når der skal hentes position fra GMT, data stream fra UR10 skal analyseres og der skal sendes nye positioner og konfigurationer til UR10. Det er også her fejlhåndtering af forbindelserne initialiseres, hvis der opstår fejl.
UTIL	Indeholder diverse DTO'er som gør det muligt for de tre lag at sende objekter imellem lagene.

Tabel 2.14: De tre lag og deres ansvar.

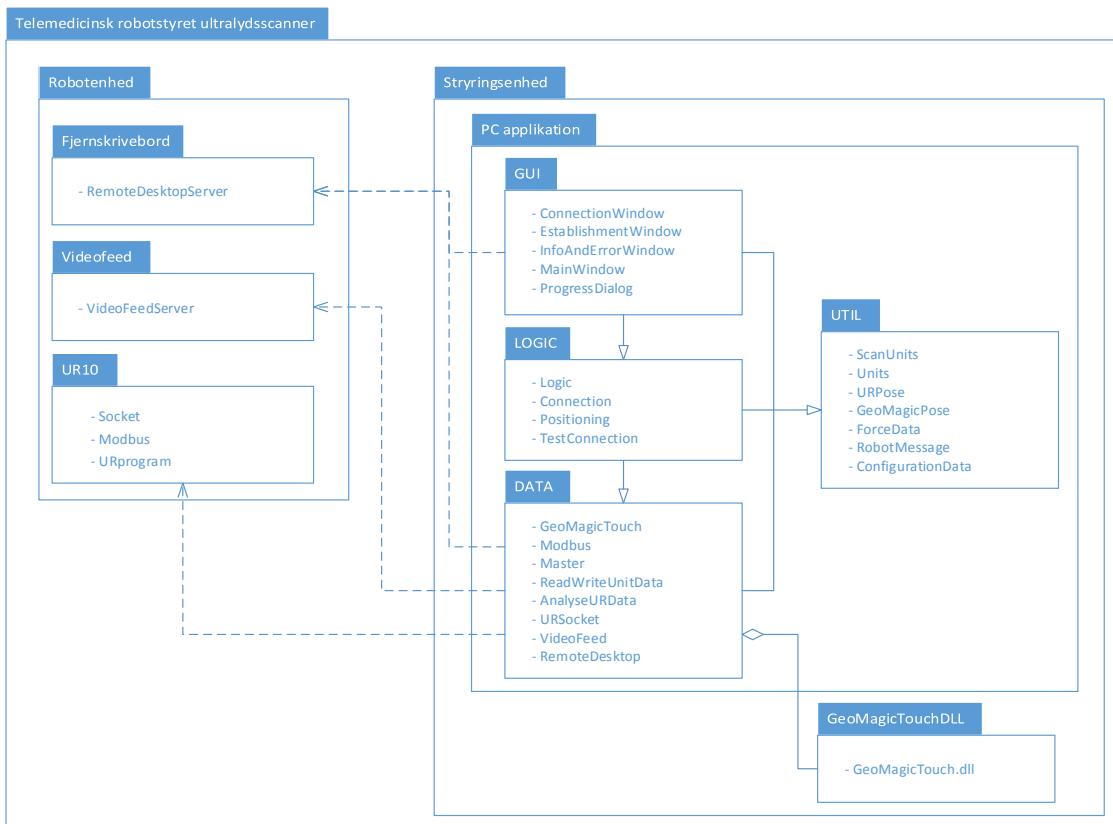
Systemdesign 3

3.1 Pakkediagram

På baggrund af softwarearkitekturen (se figur 2.11 på side 16) er der lavet et pakkediagram over systemet, som der ses på figur 3.1.

Styringsenhed og Robotenhed er delt op i hver deres pakke, hvor robotenheden har en pakke for hver enhed den består af. I styringsenheden er PC applikationen delt op i pakker der relaterer sig til de forskellige lag i trelagsopdelingen og en GeoMagicTouchDLL. Pakkerne indeholder yderligere klasser, som vil blive beskrevet senere.

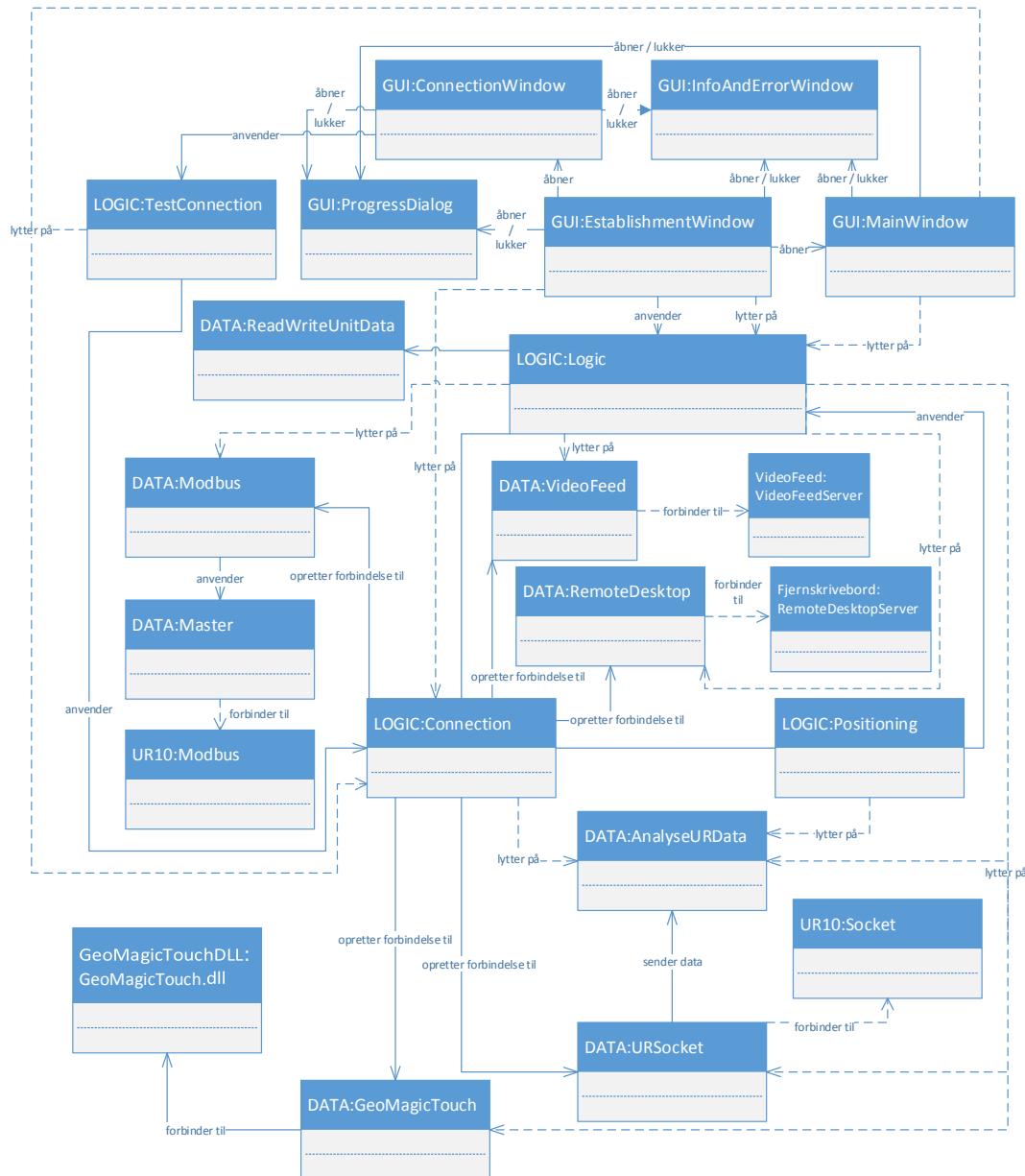
BEMÆRK at pakken UR10 angiver elementer som Socket og Modbus, som ikke er selvstændige klasser der udvikles i dette system, men blot anvendes og er derfor medtaget i pakke- og klassediagrammet for overskuelighedens skyld. URprogram er derimod et selvstændigt program der kører på UR10 og vil blive beskrevet i afsnittet Sekvensdiagrammer.



Figur 3.1. Pakkediagram for Telemedicinsk robotstyret ultralydsscanner

3.2 Klassediagram

Klassediagrammet nedenfor viser klasserne fra pakkediagrammet og deres relationer. For at identificere metoder og attributter er der på baggrund af dette klassediagram lavet sekvensdiagrammer for de enkelte use cases fra kravspecifikationen's funktionaliteter.



Figur 3.2. Klassediagram for Telemedicinsk robotstyret ultralydsscanner

Klasserne i figur 3.2 har følgende ansvar:

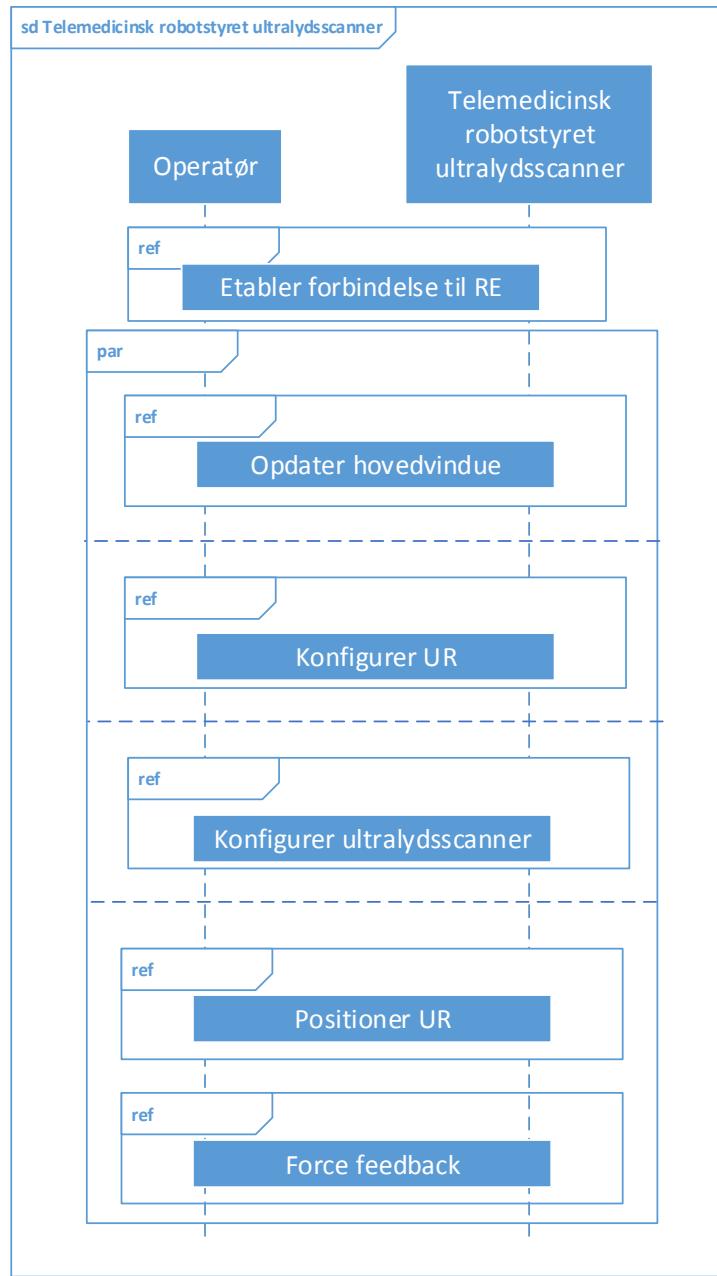
- **GUI:ConnectionWindow** har ansvaret for at der kan indskrives oplysninger om nye RE'er og initialisere test heraf. Bliver testen godkendt er det her muligt at gemme oplysningerne, så de kan tilføjes til listen over RE'er. Derudover står klassen for at informere brugeren om fejl, der kan forekomme under dette forløb.

- **GUI:EstablishmentWindow** håndtere knaptryk der giver brugeren mulighed for at etablere forbindelse til en specifik RE og et knaptryk der gør at brugeren har mulighed for at tilføje en ny RE. Derudover skal klassen visualisere eventuelle fejl der kan ske i denne fase.
- **GUI:InfoAndErrorHandler** har ansvaret for at fremvise eventuelle fejl der kan forekomme mens programmet er tændt. Klassen håndterer derudover knaptryk der giver brugeren mulighed for at handle på fejlen eller afvise fejlen.
- **GUI:MainWindow** har ansvaret for at fremvise videofeed, fjernskrivebord og konfigurationer for UR10. Derudover håndterer klassen tekstfelter hvor brugeren kan indtaste konfigurationer til UR10 og en knap hvor brugeren kan gemme disse konfigurationer. Klassen håndterer derudover også visualisering og håndtering af fejl i denne fase.
- **GUI:ProgressDialog** har ansvaret for at vise en progressbar med tilhørende tekst.
- **LOGIC:Logic** er en singleton-klasse der sørger for at der kun er en instance af denne klasse under programmets levetid. Har ansvar for at klasserne i GUI kan tilgå denne og de oprettede objekter heri. Har ansvaret for at hente/gemme scan units i en lokal xml-fil, håndtere alt fejlhåndtering videre til GUI, og indeholder objektet af den oprettede forbindelse.
- **LOGIC:Connection** har ansvaret for at oprette forbindelse til Geomagic Touch, Fjernskrivebord, VideoFeed, Modbus og UR10. Her kan der enten oprettes forbindelse enkeltvis eller til alle på en gang.
- **LOGIC:Positioning** har ansvaret for at reagere på nye positioner fra Geomagic Touch og positionere UR10'en herefter. Derudover har klassen ansvaret for at håndtere forcefeedback fra UR10'en, som omregnes og sendes til Geomagic Touch.
- **LOGIC:TestConnection** har ansvaret for at oprette forbindelse til en testenhed og når forbindelsen er oprettet, skal der testes om det er den korrekte enhed der er forbundet.
- **DATA:GeoMagicTouch** har ansvaret for kommunikationen med GeoMagicTouch.dll. Her sikrer klassen at der er tilsluttet en GMT til computeren og at den er opstartet korrekt. Derudover håndterer klassen nye positioner ved at kaste et event hver gang der registreres en ny position og påvirke GMT med en tilsvarende force. Hvis der skulle opstå fejl undervejs sørger klassen for at kaste en exception således der kan handles på fejlen.
- **DATA:Modbus** har ansvaret for at anvende DATA:Master til at oprette forbindelse til modbussen på UR10 og hente og skrive informationer fra modbussen.

- **DATA:Master** håndterer oprettelse af forbindelse til modbussen og overvåger denne forbindelse. Den gør det muligt at udføre requests til modbussen, så der kan hentes eller skrives data i registrene. Den sørger for at oprette den header der skal indgå i en request.
- **DATA:ReadWriteUnitData** har ansvaret for at gemme og hente ScanUnits fra en lokal xml-fil.
- **DATA:AnalyseURData** har ansvaret for at analysere den data stream der bliver modtaget af URSocket fra UR10.
- **DATA:URSocket** har ansvaret for at håndtere forbindelse til UR10 på port 30002, hvorefter der kontinuert modtages pakker fra data streamen, som videresendes til AnalyseURData for at blive analysere og håndteret videre.
- **DATA:VideoFeed** har ansvaret for at godkende forbindelse til VideoFeed:VideoFeedServer og oprette en RTSP forbindelse som åbner og afspiller et videofeed. Derudover håndterer klassen hvad der skal ske hvis der mistes forbindelse til VideoFeed:VideoFeedServer.
- **DATA:RemoteDesktop** har ansvaret for at indhente en invitation fra RemoteDesktopServer som skal anvendes for at forbinde til fjernskrivebordet. Hvis klassen ikke kan indhente en invitation vil klassen kaste en exception således der kan handles på fejlen.
- **GeoMagicTouchDLL:GeoMagicTouch.dll** har ansvaret for at håndtere alt kommunikation med GMT. Klassen skal læse positionen, vinklen, knaptryk og forceen for GMT. Dertil skal klassen kunne påvirke GMT med en force. Hvis der skulle ske nogle fejl skal klassen derudover kunne informere om dette.
- **Fjernskrivebord:RemoteDesktopServer** har ansvaret for at oprette en session og en tilhørende invitation, som en udefrakommende gæst kan anvende til at fjernstyre skrivebordet fra en anden computer. Derudover sørger RemoteDesktopServer for at afslutte sessionen, når den udefrakommende gæst afbryder forbindelsen.
- **VideoFeed:VideoFeedServer** har ansvaret for at starte en server hvorpå DATA:VideoFeed har mulighed for at tjekke om der er tilsluttet et webcam og om der kan streames. Klassen har også ansvaret for at starte en server der streamer et videofeed.
- **UR10:Socket** er en server på UR10 der står for at sende data ud på et stream, som indeholder informationer omkring UR10.
- **UR10:Modbus** er en server på UR10 der består af en række registre, som der kan læses og skrives til.

3.3 Sekvensdiagrammer

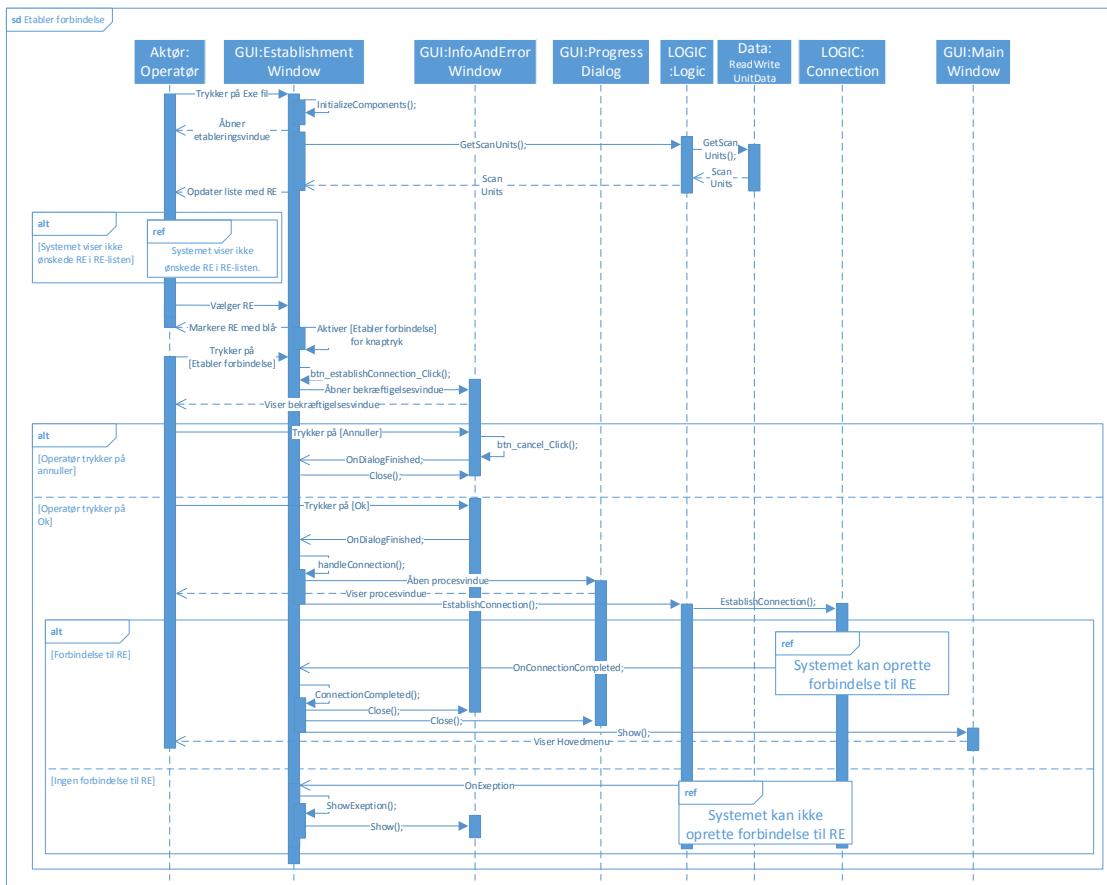
Dette afsnit vil beskrive systemets tidsafhængighed og funktionalitet via sekvensdiagrammer. Disse diagrammer er lavet på baggrund af use cases fra kravspecifikationen. I nedenstående diagram ses Operatørens interaktion med Telemedicinsk robotstyret ultralydsscanner. De enkelte referencer i diagrammet vil blive beskrevet i de kommende afsnit.



Figur 3.3. Sekvensdiagram for Telemedicinsk robotstyret ultralydsscanner

3.3.1 Etabler forbindelse

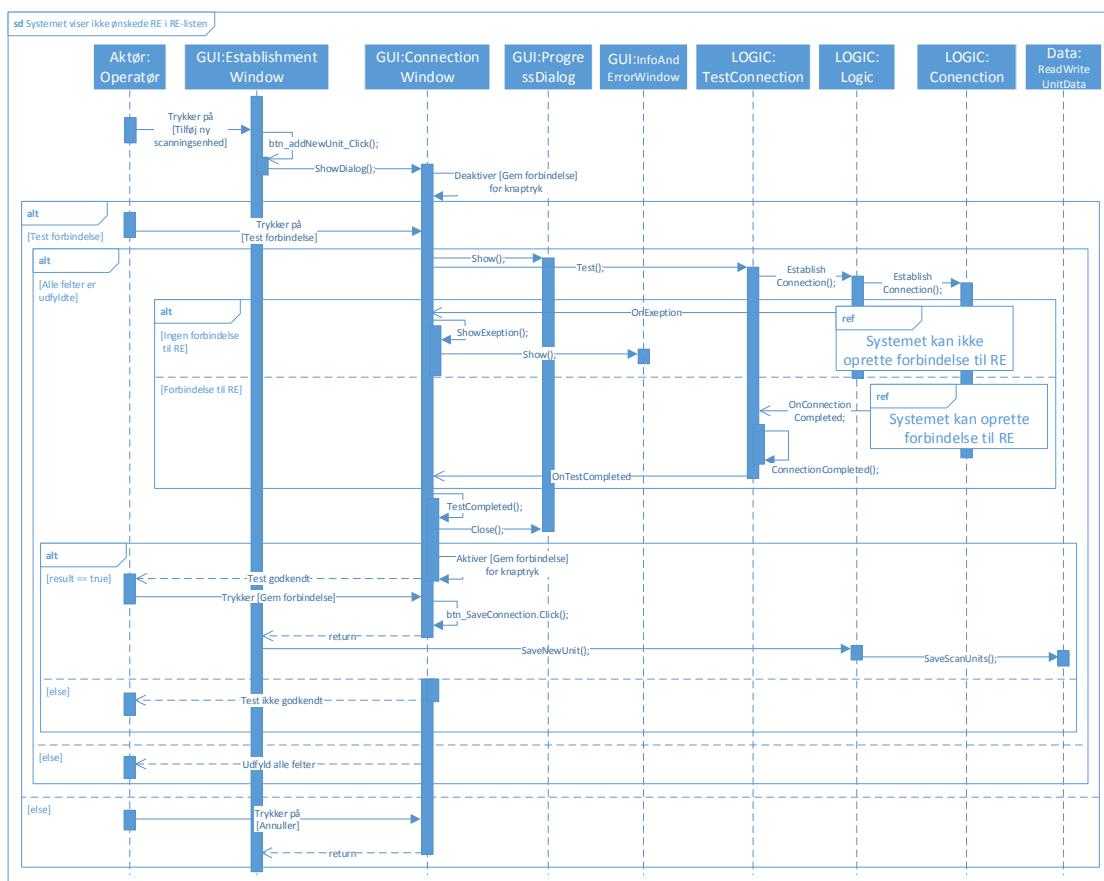
Nedenstående sekvensdiagram viser sekvensen for Etabler forbindelse. Diagrammet viser at Operatøren starter Telemedicinsk Robotstyret Ultralydsscanning, som åbner EstablishmentWindow. Her har Operatøren mulighed for at tilføje en ny RE hvis den ønskede ikke findes i listen (Se figur 3.5) eller trykke på [Etabler forbindelse] og forbinde til en eksisterende RE. Der vil enten kunne oprettes forbindelse til UR10, Remote desktop, Modbus, GMT og Videofeed eller ikke. Hvis der kan oprette forbindelse (Se figur 3.7) vil EstablishmentWindow lukke og åbne MainWindow. Hvis der ikke kan oprette forbindelse (Se figur 3.6) vil der returneres til EstablishmentWindow og Operatøren vil bliver informeret med en fejl.



Figur 3.4. Sekvensdiagram for Etabler forbindelse

Systemet viser ikke ønskede RE

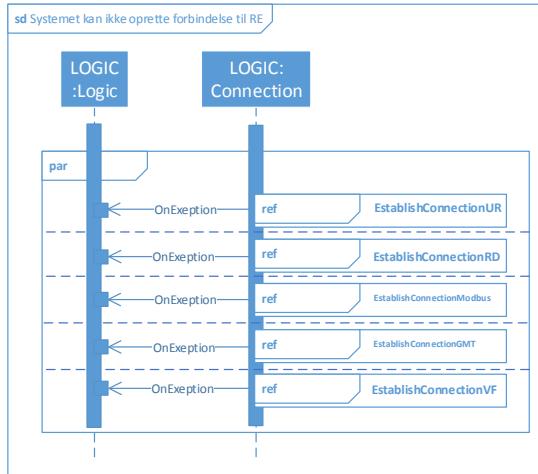
Sekvensdiagrammet nedenfor viser et scenarie hvor den RE, som der ønskes forbindelse til ikke findes i RE-listen. I sekvensen trykker Operatøren på [Tilføj ny scanningsenhed], som vil åbne ConnectionWindow. Her har Operatøren mulighed for at trykke på [Annuler] og vil blive returneret til EstablishmentWindow eller trykke på [Test forbindelse] som enten vil oprette forbindelse (Se figur 3.7) eller ikke kunne oprette forbindelse (Se figur 3.6). Hvis der kan oprettes forbindelse til RE vil information om de forbundne enheder, (UR10 og Fjernskrivebord) bliver holdt op imod indtastet data i ConnectionWindow. Er der overenstemmelse mellem data vil Operatøren blive informeret om at testen er godkendt og [Gem forbindelse] vil blive aktiveret for knaptryk. Ved tryk på [Gem forbindelse] returneres der til EstablishmentWindow hvor den nye RE findes i listen. Hvis der ikke er overenstemmelse i data vil Operatøren blive informeret om at testen ikke kunne godkendes. Ved tryk på [Annuler] vil ConnectionWindow lukke og Operatøren returneres til EstablishmentWindow.



Figur 3.5. Sekvensdiagram for Systemet viser ikke ønskede RE i RE-listen

Systemet kan ikke oprette forbindelse til RE

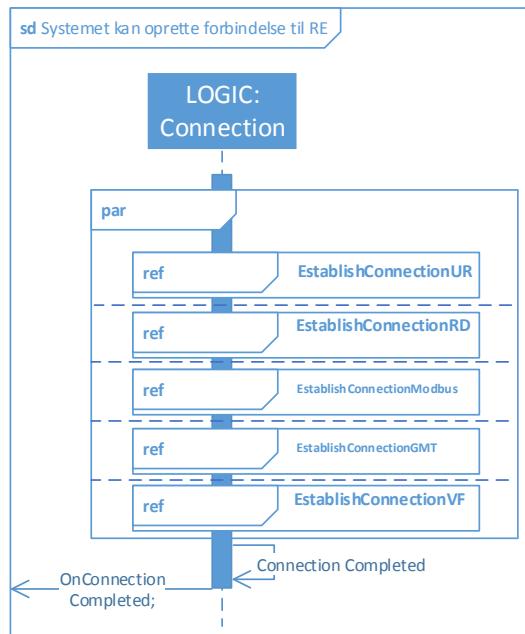
Sekvensdiagrammet nedenfor viser et diagram over scenariet, hvor der ikke kan oprettes forbindelse til enten UR10(Se figur 3.8), Remote desktop(Se figur 3.9), Modbus(Se figur 3.10), GMT(Se figur 3.12) eller Videofeed(Se figur 3.11). Hvert af disse tilfælde vil smide en OnException, hvor Logic informeres om at der er opstået en fejl.



Figur 3.6. Sekvensdiagram for at der ikke kan oprettes forbindelse

Systemet kan oprette forbindelse til RE

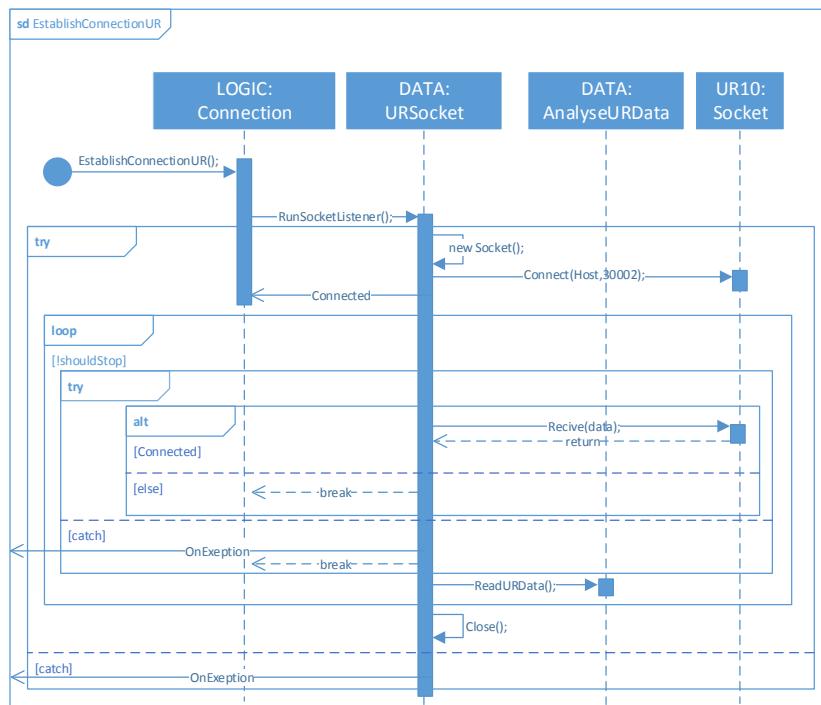
Sekvensdiagrammet nedenfor viser forløbet hvor der succesfuldt oprettes forbindelse til UR10(Se figur 3.8), Remote desktop(Se figur 3.9), Modbus(Se figur 3.10), GMT(Se figur 3.12) og Videofeed(Se figur 3.11). Her vil der blive sendt en OnConnectionCompleted som informerer om at forbindelsen er oprettet succesfuldt.



Figur 3.7. Sekvensdiagram for oprettelses af forbindelse.

EstablishConnectionUR

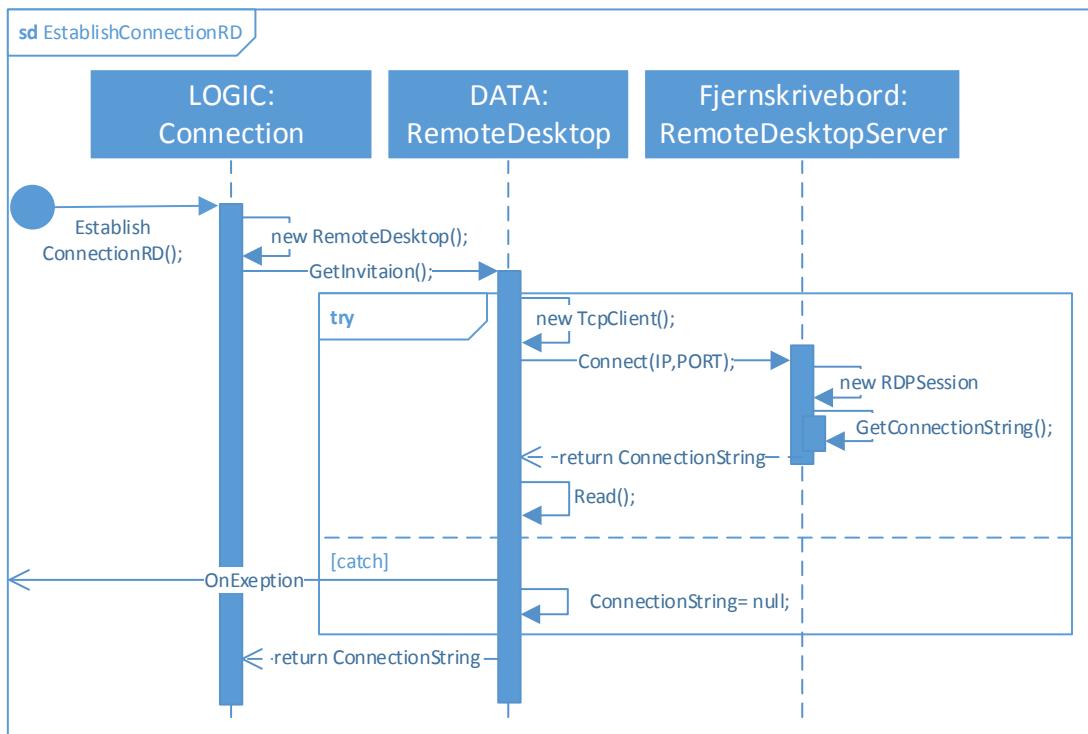
Nedenstående sekvensdiagram viser hvorledes der forsøges at oprette forbindelse til UR10. Hvis der oprettes forbindelse informeres Connection og der startes et loop der henter data fra UR10:Socket indtil loopet afsluttes. Hvis der ikke kan oprettes forbindelse kastes der en OnException som håndteres andensteds (se figur 3.6).



Figur 3.8. Sekvensdiagram for EstablishConnectionUR

EstablishConnectionRD

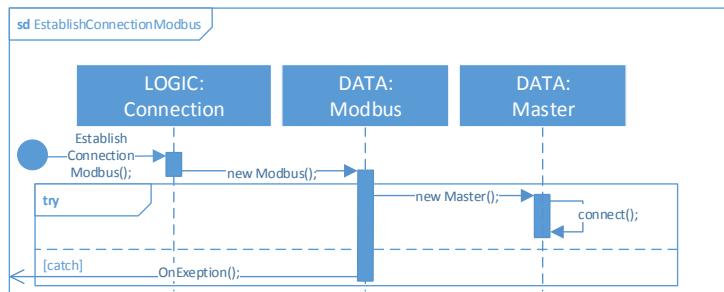
Sekvensdiagrammet nedenfor viser sekvensen, hvor der forsøges at oprette forbindelse til Remote Desktop. Her oprettes en forbindelse til RemoteDesktopServer som overfører en ConnectionString til RemoteDesktop og den returneres til Connection som herefter ved at der er oprettet forbindelse til RemoteDesktopServer. Hvis der ikke kan oprettes forbindelse er ConnectionString null og der sendes en OnException som håndteres andensteds (se figur 3.6).



Figur 3.9. Sekvensdiagram for EstablishConnectionRD

EstablishConnectionModbus

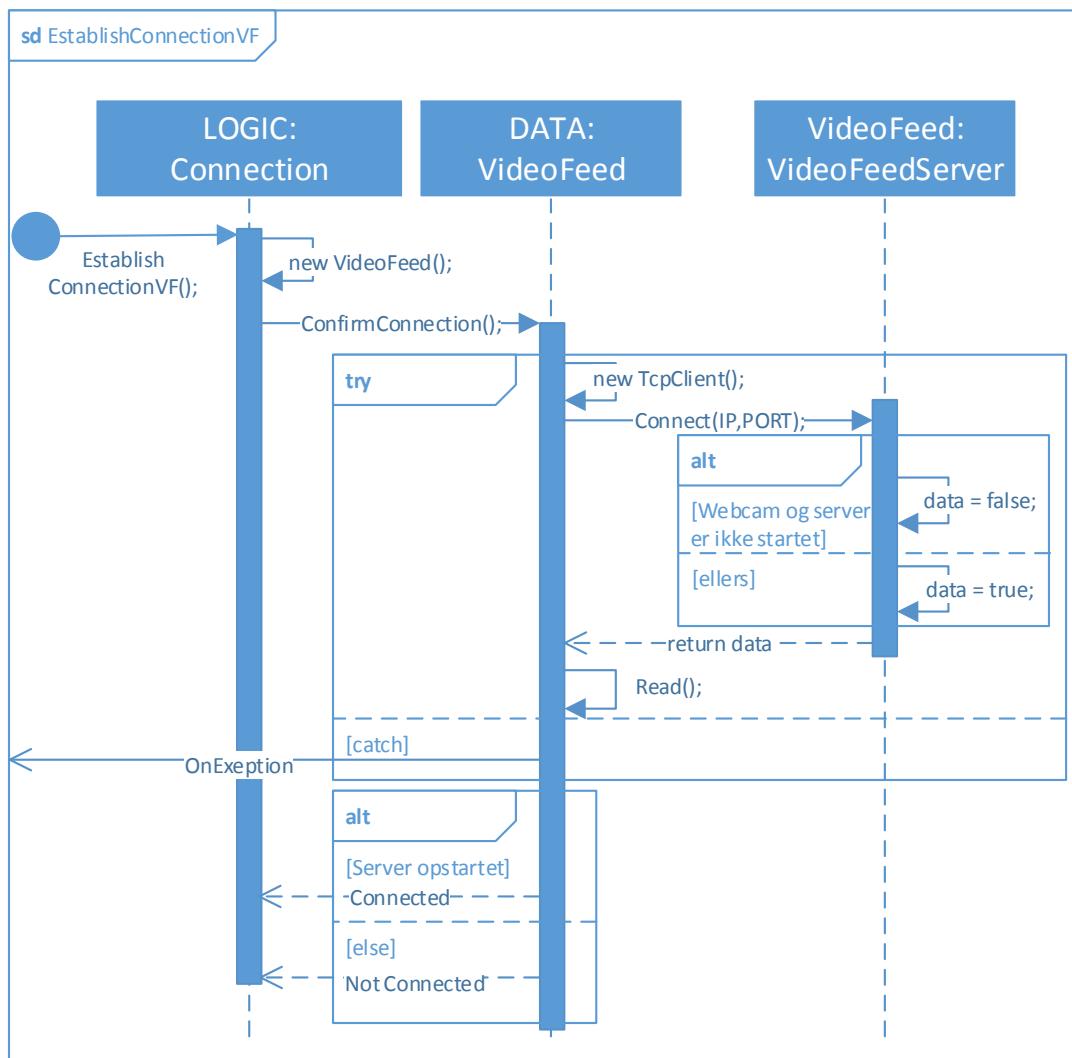
Sekvensdiagrammet nedenfor viser sekvensen, hvor der forsøges at oprette forbindelse til modbussen på UR10. Hvis der ikke kan oprettes forbindelse sendes en `OnException` som håndteres andensteds (se figur 3.6).



Figur 3.10. Sekvensdiagram for EstablishConnectionModbus

EstablishConnectionVF

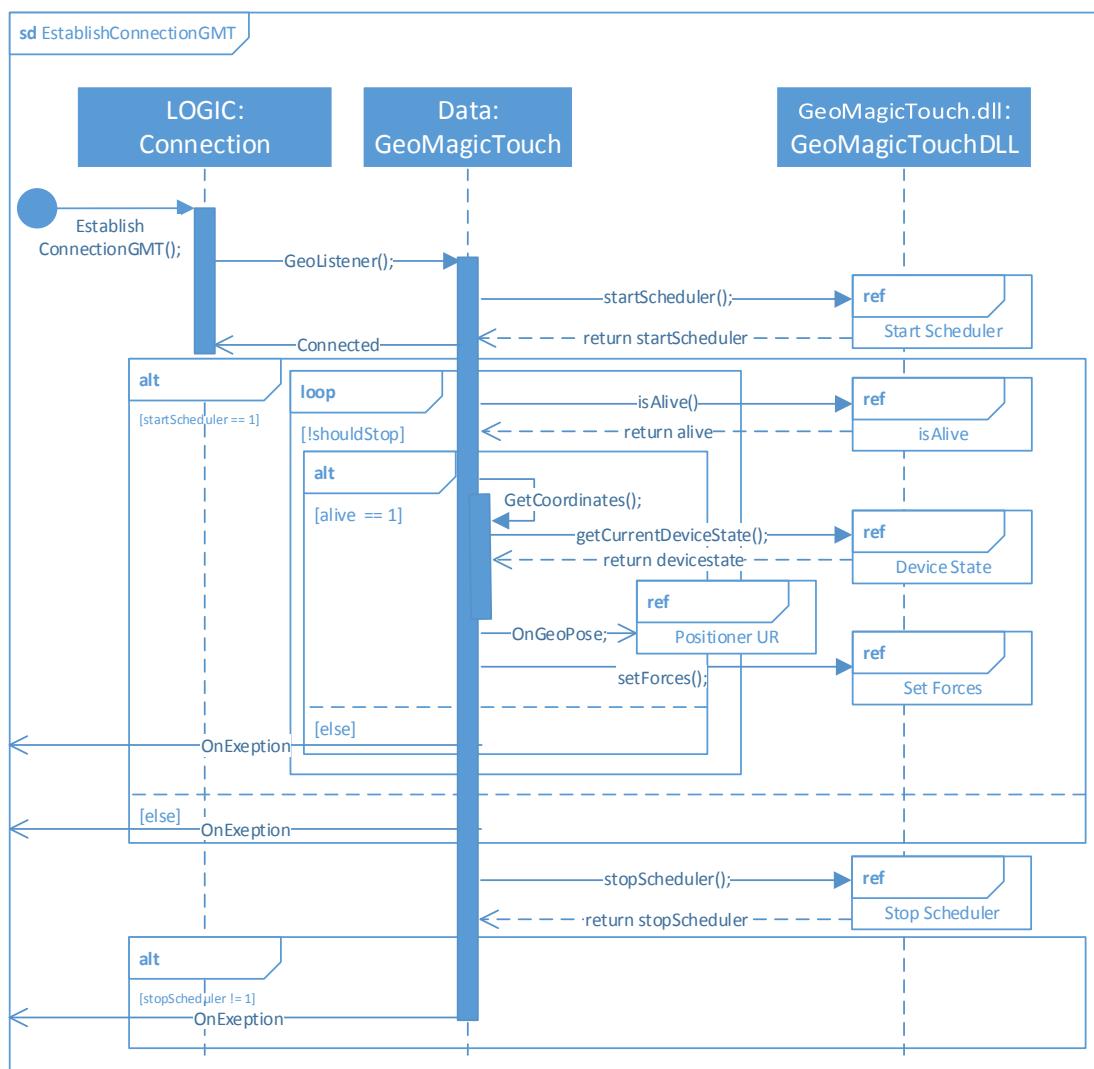
Nedenstående sekvensdiagram viser hvorledes der forsøges at oprette forbindelse til VideoFeedServer. Her vil VideoFeedServer prøve at starte en server og hvis dette lykkes informeres VideoFeed om at der er oprettet forbindelse og kan derved notificere Connection. Hvis der ikke kan startes en server i VideoFeedServer informeres VideoFeed om dette og notificerer herefter Connection om at der ikke er oprettet forbindelse.



Figur 3.11. Sekvensdiagram for EstablishConnectionVF

EstablishConnectionGMT

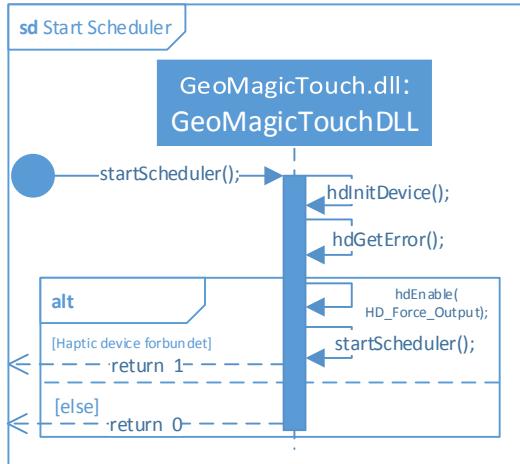
Sekvensdiagrammet viser sekvensen for at oprette forbindelse til GMT og hvor der kontinuerligt sendes og hentes data. Først kaldes Start Scheduler (Se figur 3.13) Hvis denne returnerer 1 informeres Connection om at der er forbindelse til GeoMagicTouchDLL og derved GMT. Herefter startes et loop der kalder isAlive (Se figur 3.14) for at tjekke forbindelsen til GMT i hver iteration. Hvis der er forbindelse kaldes Device State (Se figur 3.15), som henter informationer omkring GMT. Disse informationer bliver kastet til Positioner UR (Se figur 3.24), som håndterer positionering af UR10. I hver iteration kaldes Set Force (Se figur 3.16) for at sætte forcen på GMT. Hvis loopet afsluttes bliver Stop Scheduler (Se figur 3.17) kaldt for at afbryde forbindelsen til GMT. Hvis denne ikke returnerer 1 bliver der kastet en OnException som håndteres andensted.



Figur 3.12. Sekvensdiagram for EstablishConnectionGMT

Start Scheduler

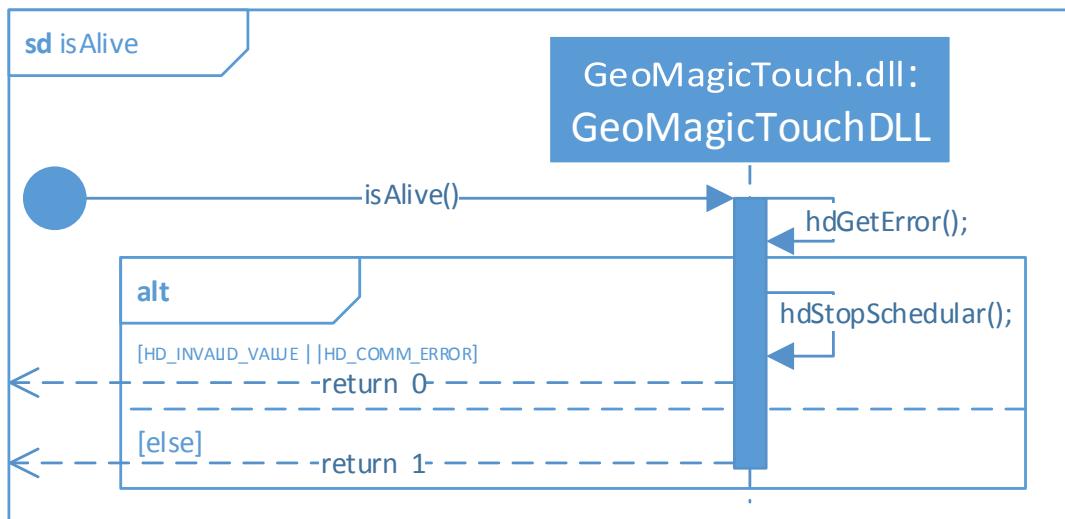
Nedenstående sekvensdiagram viser opstartssekvensen for GMT. Her initialiseres et nyt GeoMagic Device og forceen aktiveres. Hvis der er forbundet et GeoMagic Device returneres 1 ellers returneres 0.



Figur 3.13. Sekvensdiagram for Start Scheduler

isAlive

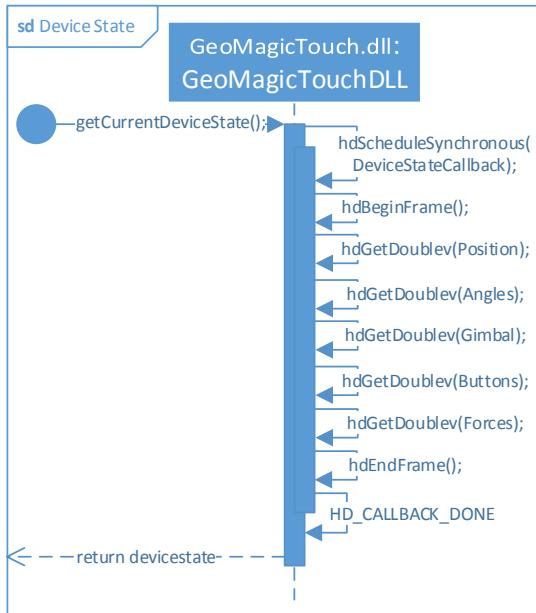
Sekvensen viser hvordan der tjekkes om der er opstået fejl under kommunikationen med GMT. Her anvendes GMT's egen fejlhåndtering til at tjekke om der er en forbindelsesfejl eller at der er forsøgt at sætte en ukorrekt værdi. Hvis dette skulle ske stoppes scheduleren og der returneres 0 ellers returneres 1 for succes.



Figur 3.14. Sekvensdiagram for isAlive

Device State

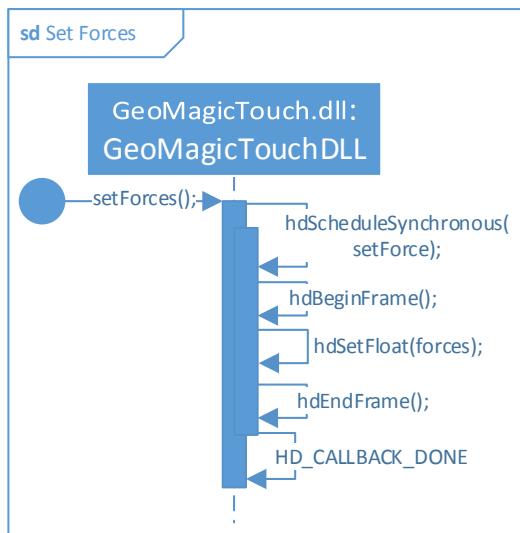
Sekvensdiagrammet nedenfor viser hvorledes der hentes information fra GMT. Her arrangeres et callback til Scheduleren, hvor der startes et nyt frame. I dette frame hentes position, angles, gimbal, buttons og forces før framet afsluttes. Når framet er afsluttet returneres værdien devicestate som indeholder alle fornævnte værdier.



Figur 3.15. Sekvensdiagram for Device State

Set Forces

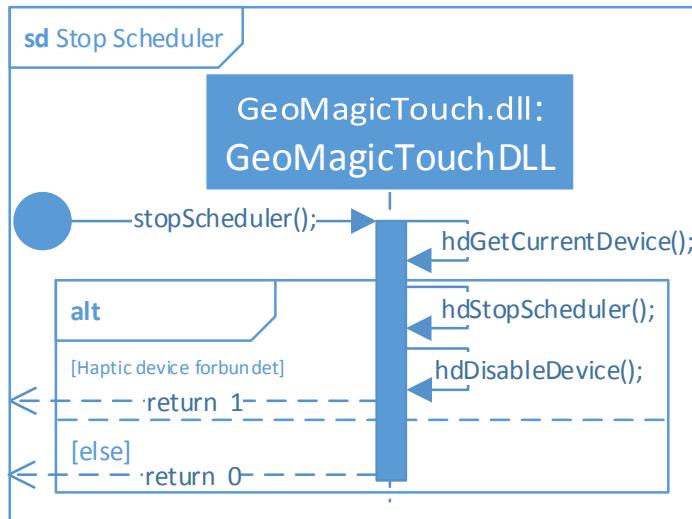
Sekvensen nedenfor viser hvorledes forceen sættes på GMT. Dette gøres ved at lave et callback til Scheduleren, som starter et nyt frame. I dette frame sættes forceen og framet afsluttes.



Figur 3.16. Sekvensdiagram for Set Forces

Stop Scheduler

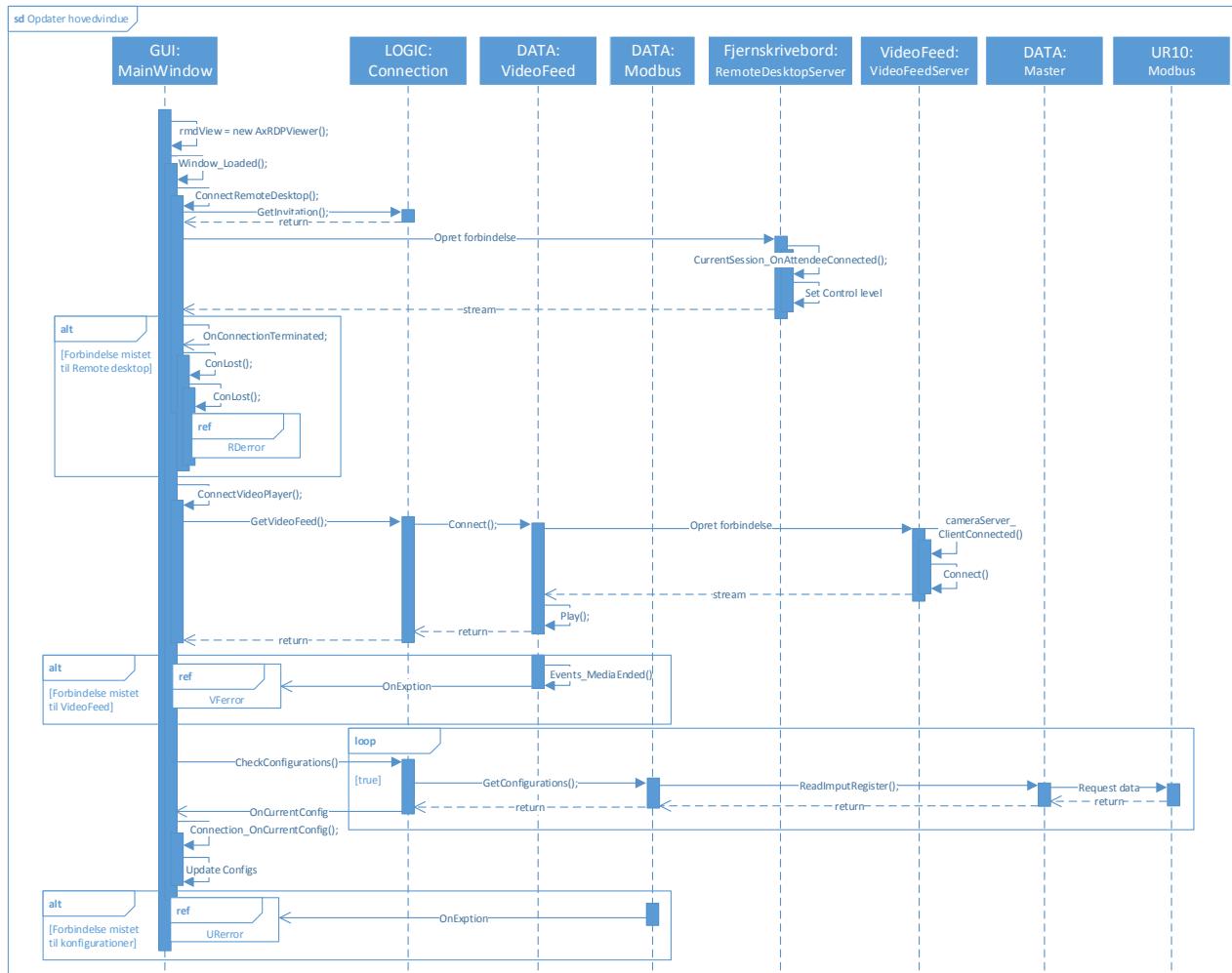
Nedenstående sekvensdiagram viser sekvensen for at afbryde forbindelsen til GMT. Her hentes det nuværende tilsluttede device. Hvis der er tilsluttet et device stoppes scheduleren og devicet frakobles og der returneres 1 ellers 0.



Figur 3.17. Sekvensdiagram for Stop Scheduler

3.3.2 Opdater hovedvindue

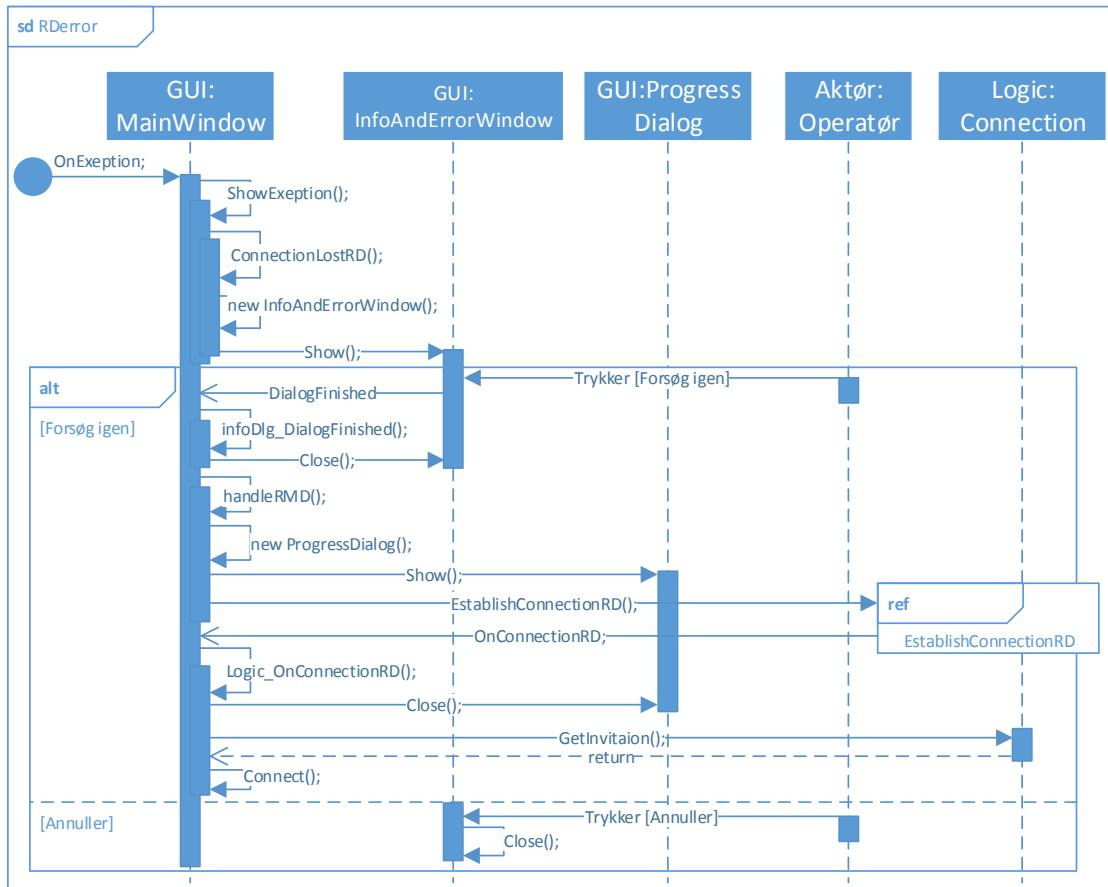
Sekvensdiagrammet nedenfor viser sekvenser for at opdatere hovedvinduet. I diagrammet ses hvorledes MainWindow starter en sekvens der forsøger at oprette forbindelse til Fjernskrivebord, VideoFeed og Modbus. For at oprette forbindelse til Fjernskrivebord hentes en Invitation i Connection hvorefter der forbindes til RemoteDesktopServer, som registrerer en ny bruger og indstiller Control level for den nye bruger. Hvis der skulle mistes forbindelse til RemoteDesktopServer håndteres fejlen i RDerror (Se figur 3.19). For at oprette forbindelse til VideoFeed åbnes et nyt medie i form af et stream fra VideoFeedServer som herefter afspilles. Hvis der mistes forbindelse informeres VideoFeed, som herefter kan informere VFerror (Se figur 3.20), som håndterer fejlen. Herefter ses hvorledes der kontinuerligt hentes informationer fra UR10 ved at hente data ud fra modbussen. Hvis der ikke kan hentes informationer opdateres MainWindow ellers kastes OnException, som bliver håndteret i URerror (Se figur 3.21).



Figur 3.18. Sekvensdiagram for Opdater hovedvindue

RDerror

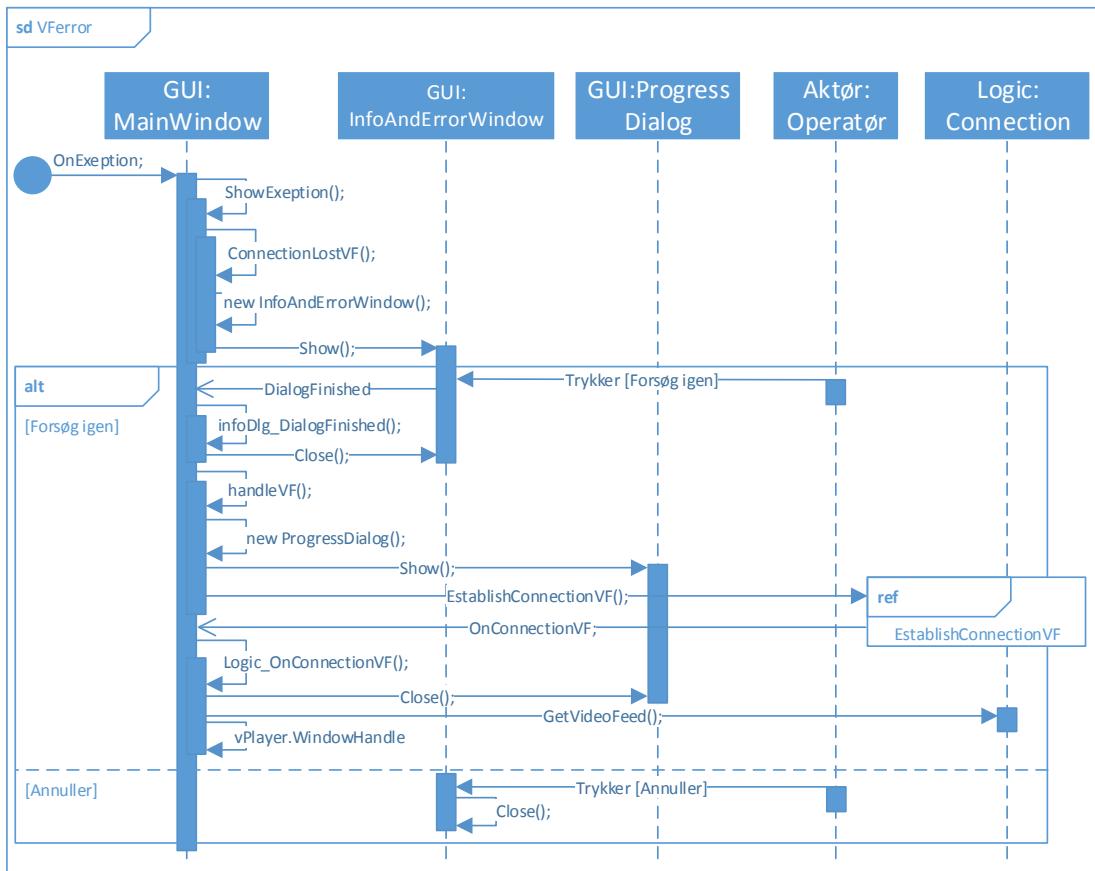
Sekvensdiagrammet nedenfor viser sekvensen der håndterer hvis PC applikationen skulle miste forbindelse til fjernskrivebordet. Her informeres Operatøren med en fejlmeddeelse hvor der er mulighed for at prøve at oprette forbindelse igen eller annullere forsøget. Hvis Operatøren ønsker at oprette forbindelse igen startes EstablishConnectionRD (Se figur 3.9).



Figur 3.19. Sekvensdiagram for RDerror

VFerror

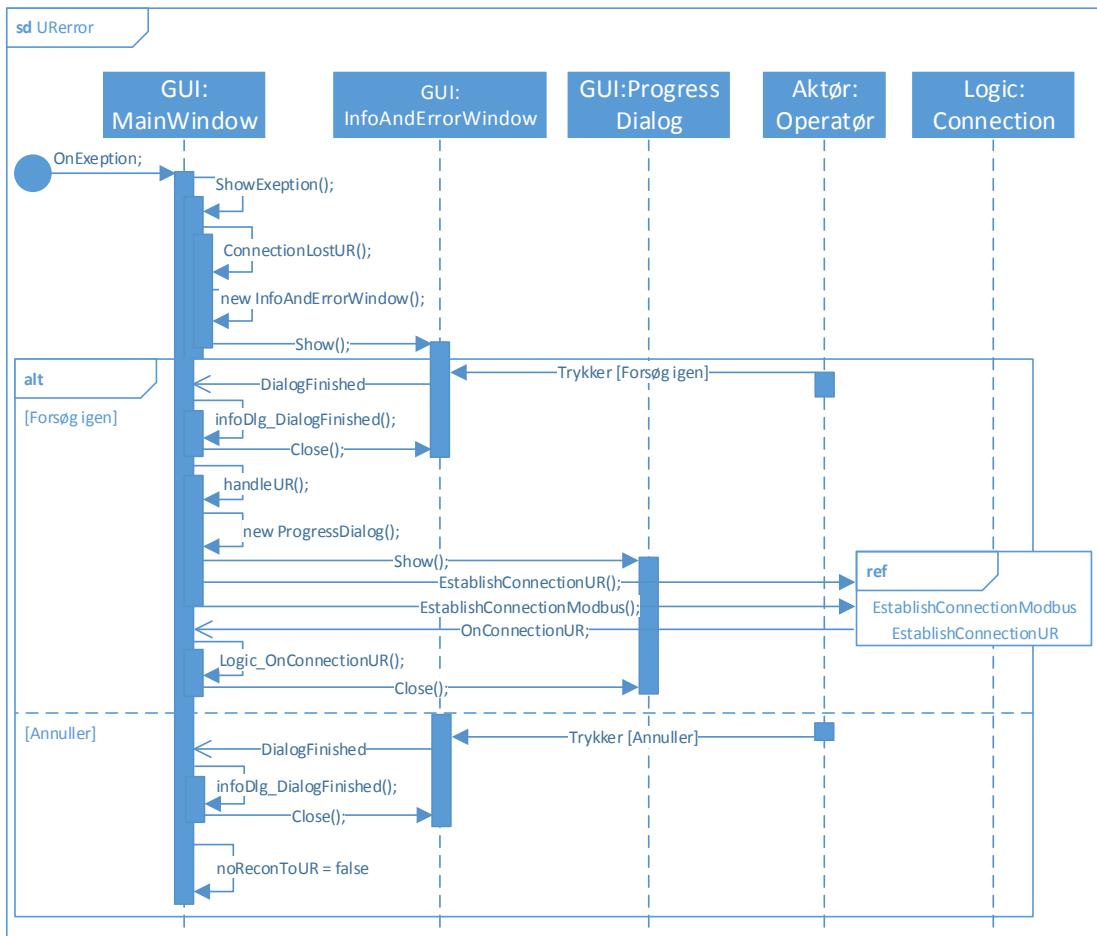
Sekvensdiagrammet nedenfor viser hvorledes det håndteres hvis PC applikationen mister forbindelsen til VideoFeed. Her bliver Operatøren informeret om at der er opstået en fejl og har herefter mulighed at trykke på [Forsøg igen] eller [Annuler]. Hvis han trykker [Forsøg igen] vil InfoAndErrorWindow lukke, ProgressDiaglog vil åbne og EstablishConnectionVF (Se figur 3.11) vil håndtere at oprette forbindelse til VideoFeed.



Figur 3.20. Sekvensdiagram for VFerror

URerror

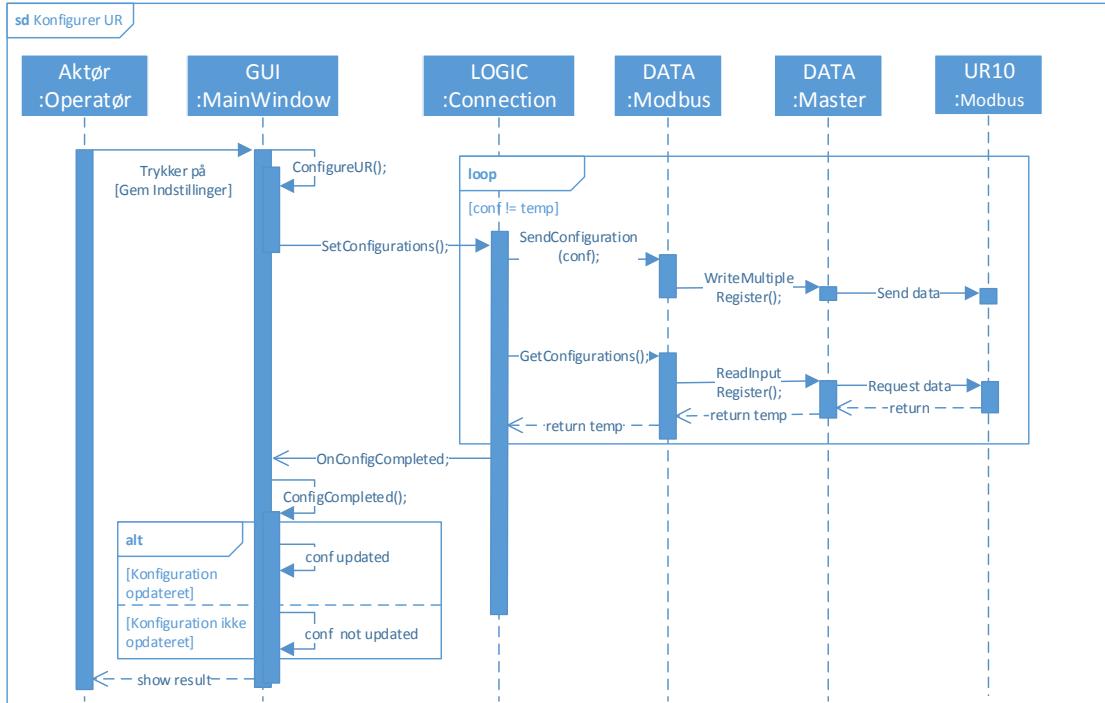
Nedenstående sekvensdiagram viser en sekvens hvor systemet har mistet forbindelse til UR10 og Operatøren har mulighed for at handle på fejlen. Her har Operatøren mulighed for at trykke [Forsøg igen] og MainWindow vil sikre at der bliver oprettet forbindelse, som håndteres i EstablishConnectionModbus (Se figur 3.10) og i EstablishConnectionUR (Se figur 3.8). Hvis Operatøren trykker på [Annuler] bliver der ikke oprettet forbindelse og der returneres til MainWindow.



Figur 3.21. Sekvensdiagram for URerror

3.3.3 Konfigurer UR

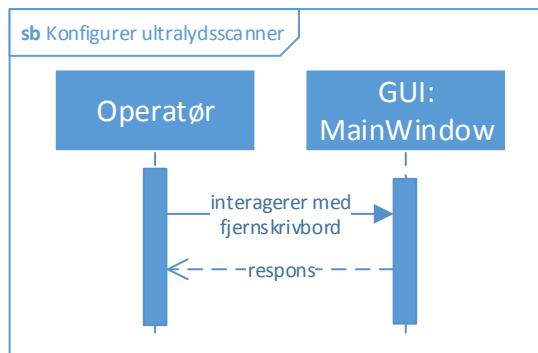
Sekvensdiagrammet nedenfor viser sekvensen hvor Operatøren trykker på [Gem indstillinger]. Her vil der startes et loop der vil blive ved med at overføre konfigurationer fra DATA:Modbus til UR10:Modbus indtil der er overenstemmelse mellem de konfigurationer der hentes ud af UR10:Modbus. Herefter bliver Operatøren informeret om det har været muligt at opdatere konfigurationer eller ej.



Figur 3.22. Sekvensdiagram for Konfigurer UR

3.3.4 Konfigurer ultralydsscanner

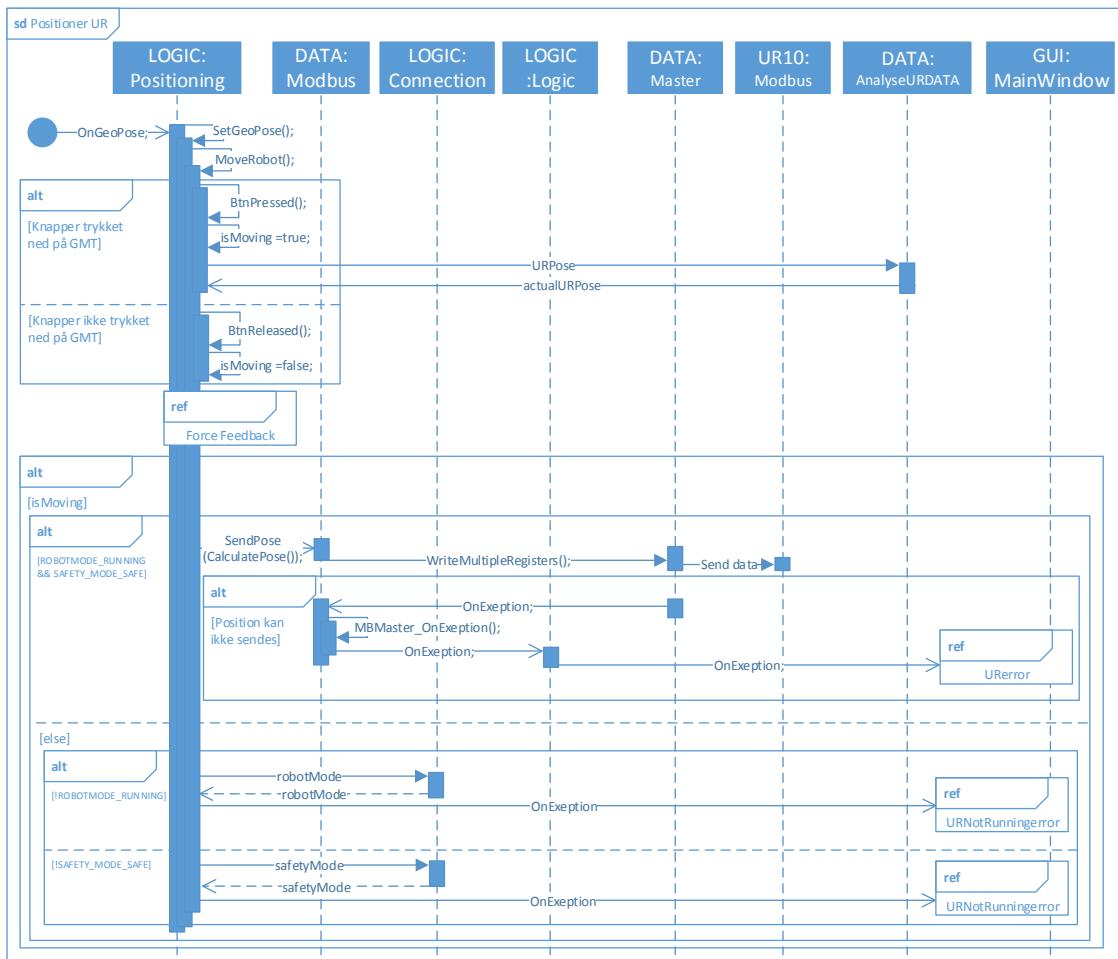
Sekvensdiagrammet nedenfor viser hvorledes Operatøren interagerer med fjernskrivebordet og herefter får en respons fra interaktionen.



Figur 3.23. Sekvensdiagram for Konfigurer ultralydsscanner

3.3.5 Positioner UR

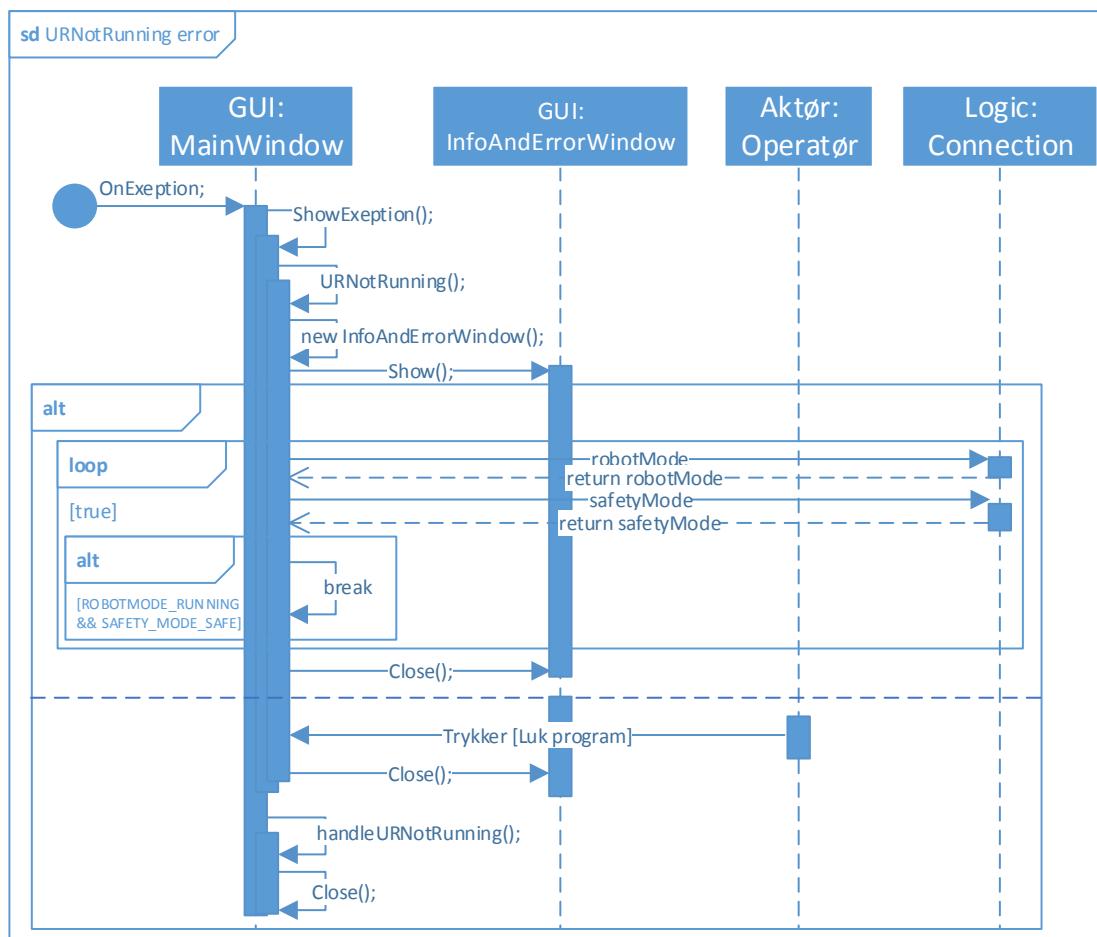
Sekvensdiagrammet nedenfor viser hvorledes UR10 positioneres. Først tjekkes om knapperne er nede på GMT hvorefter Force Feedback (Se figur 3.26) håndterer forcen mellem UR10 og GMT. Hvis knapperne er trykket ned på GMT hentes en ny position for UR10'en som anvendes til at beregne en ny position der sendes til UR10:Modbus. Hvis der ikke kan sendes en position opstår der en fejl som håndteres af URerror (Se figur 3.21). Herefter tjekkes om UR10's robotmode og safetymode kan godkendes ellers opstår en fejl som håndteres af URNotRunningerror (Se figur 3.25).



Figur 3.24. Sekvensdiagram for Positioner UR

URNotRunningerror

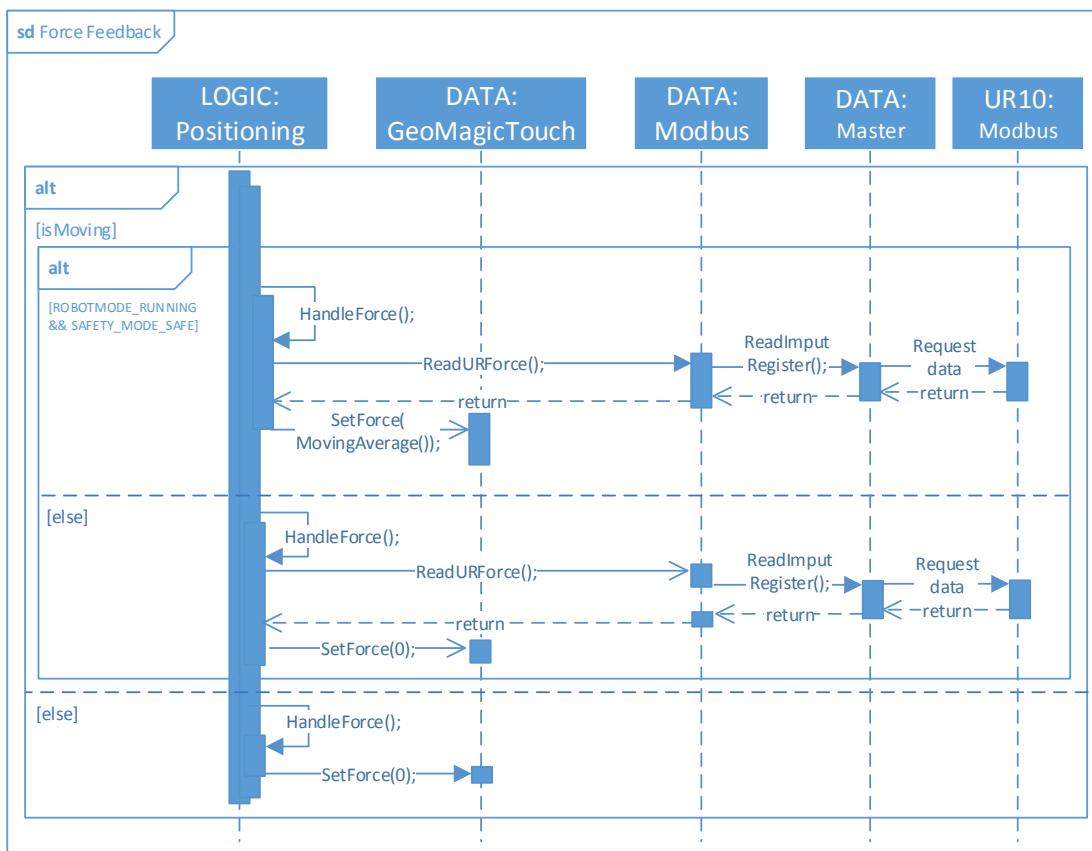
Nedenstående sekvensdiagram viser hvorledes en fejl håndteres når UR10's robotmode eller saftymode ikke er som forventet. Her bliver Operatøren informeret i InfoAndErrorWindow om at der er opstået en fejl og har nu mulighed for at afslutte programmet eller afvente at robotmode og safetymode ændrer sig. Herefter startes et loop der i hver iteration henter robotmode og safetymode for at tjekke om disse skulle ændre sig. Når robotmode og safetymode igen er korrekte vil loopet blive afsluttet og InfoAndErrorWindow bliver lukket og der returneres til MainWindow.



Figur 3.25. Sekvensdiagram for URNotRunning error

3.3.6 Force Feedback

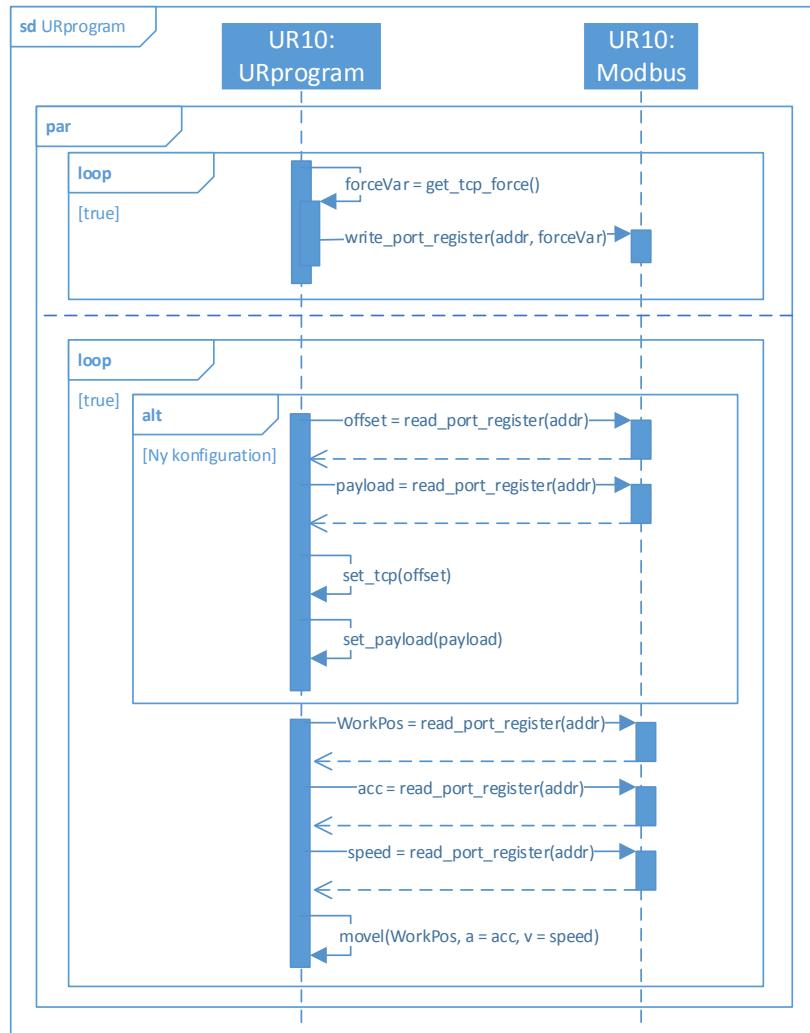
Nedenstående sekvensdiagram viser sekvensen for at håndtere forcefeedback. Diagrammet viser hvorledes forcen læses fra UR10'en og omregnes til en passende kraft, som sendes til GMT og derved påvirker Operatøren. Derudover viser diagrammet hvordan force påvirkningen håndteres i forskellige scenarier.



Figur 3.26. Sekvensdiagram for Force Feedback

3.3.7 URprogram

Nedenstående sekvensdiagram viser sekvensen for programmet der kører på UR10. Når programmet er startet kører der to parallelle loops, hvor det ene står for at aflæse den aktuelle force på Tool Center Point, og skrive den ind i modbusregisteret. Det andet loop aflæser den ønskede position, acceleration og hastighed for derefter vha. *moveL* bevæge robotten ud i den ønskede position. På den måde står robotten hele tiden i den position der er skrevet ind i modbusregisteret. Såfremt der bliver sendt nye konfigurationer fra PC applikationen, aflæses værdierne i registeret og robotten indstilles til de angivet værdier.



Figur 3.27. Sekvensdiagram for URprogram

3.4 Detaljeret specifikation af klassediagrammer

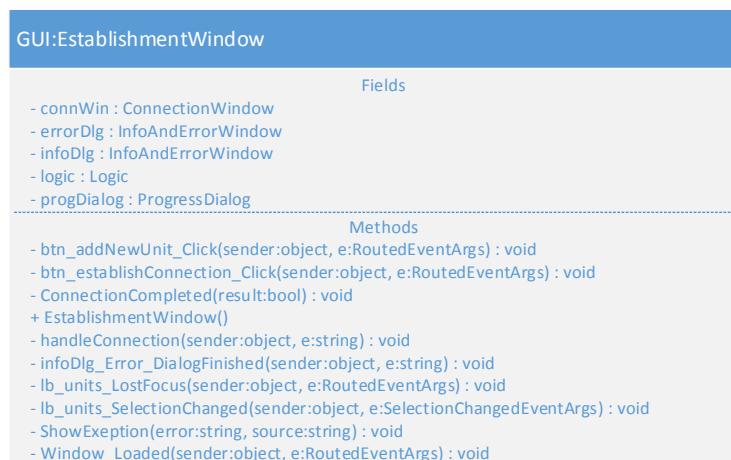
Formålet med dette afsnit er at vise detaljerede specifikationer af tidligere definerede klassediagrammer på baggrund af forrige afsnits sekvensdiagrammer. For at se en sammenhæng mellem klasserne kan der anvendes figur 3.2 eller Bilag 15 for et fuldt klassediagram med attributter, metoder og events.

GUI:ConnectionWindow



Figur 3.28. Detaljeret specifikation af ConnectionWindow

GUI:EstablishmentWindow



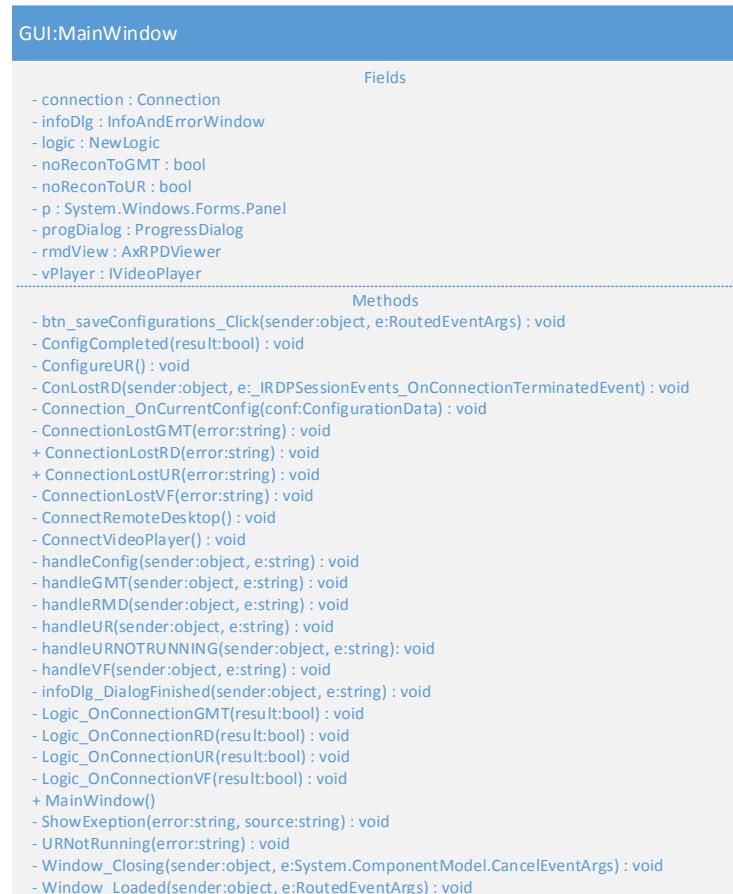
Figur 3.29. Detaljeret specifikation af EstablishmentWindow

GUI:InfoAndErrorHandler

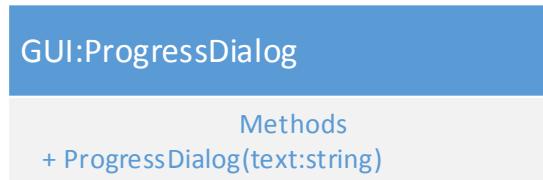


Figur 3.30. Detaljeret specifikation af InfoAndErrorWindow

GUI:MainWindow



Figur 3.31. Detaljeret specifikation af MainWindow

GUI:ProgressWindow

Figur 3.32. Detaljeret specifikation af ProgressDialog

LOGIC:Logic

Figur 3.33. Detaljeret specifikation af Logic

LOGIC:Connection

LOGIC:Connection	
	Fields
- configThread : Thread - gmtThread : Thread - invitation : string - isConnected : bool - logic : Logic - positioning : Positioning - rd : RemoteDesktop - socketTread, gmtTread : Thread - timeout : int - videoFeed : VideoFeed - videoThread : Thread - waitHandles : AutoResetEvent[]	
	Properties
- gmt : GeoMagicTouch - modbus : Modbus - robotMode : string - safetyMode : string - socket : URSocket	
	Methods
+ CheckConfigurations() : void + CloseConnection() : void + Connection () : void + EstablishConnection() : void + EstablishConnectionGMT() : void + EstablishConnectionModbus() : void + EstablishConnectionRD() : void + EstablishConnectionUR() : void + EstablishConnectionVF() : void + GetInvitation() : string + GetVideoFeed(vPlayer: IVideoPlayer) : void + SelectConfigurations(configData:ConfigurationData) : void - SetRobotMode(robotMode:string, controlMode:string) : void - SetSafetyMode(safetyMode:string) : void	
	Events
+ OnConnectionCompleted : ConnectionCompletedEventHandler + OnConfigCompleted : ConfigurationEventHandler + OnConnectionRD : RDConnectionEventHandler + OnConnectionGMT : GMTConnectionEventHandler + OnConnectionUR : URConnectionEventHandler + OnConnectionVF : VFConnectionEventHandler + OnCurrentConfig : CurrentConfigEventHandler	
	Nested Types
+ ConnectionCompletedEventHandler(result:bool) : void + ConfigurationEventHandler(result:bool) : void + CurrentConfigEventHandler(conf:ConfigurationData) : void + GMTConnectionEventHandler(result:bool) : void + URConnectionEventHandler(result:bool) : void + VFConnectionEventHandler(result:bool) : void + RDConnectionEventHandler(result:bool) : void	

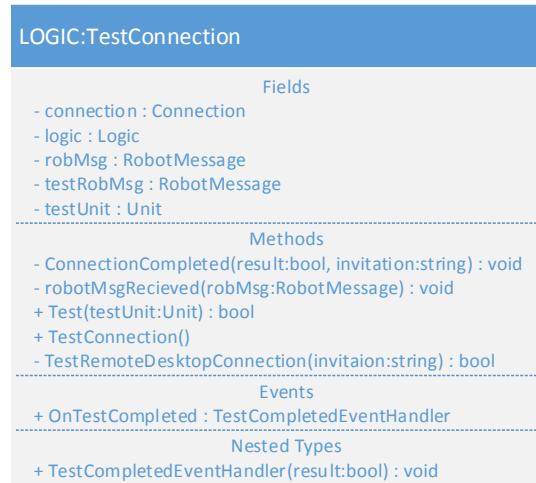
Figur 3.34. Detaljeret specifikation af Connection

LOGIC:Positioning



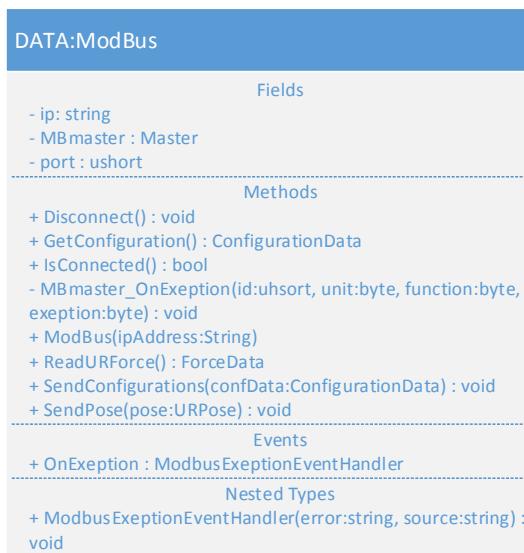
Figur 3.35. Detaljeret specifikation af Positioning

LOGIC:TestConnection



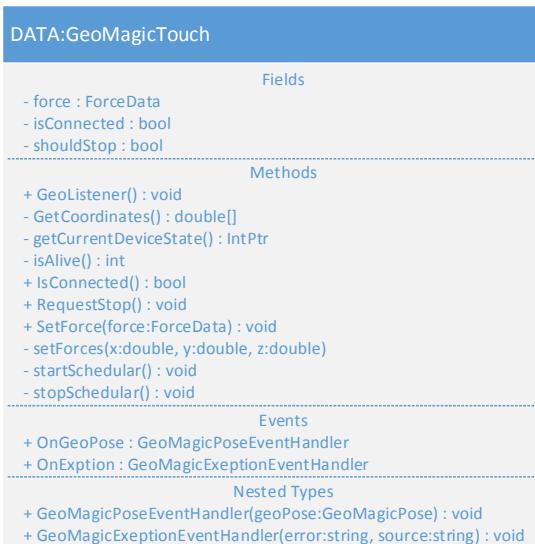
Figur 3.36. Detaljeret specifikation af TestConnection

DATA:Modbus



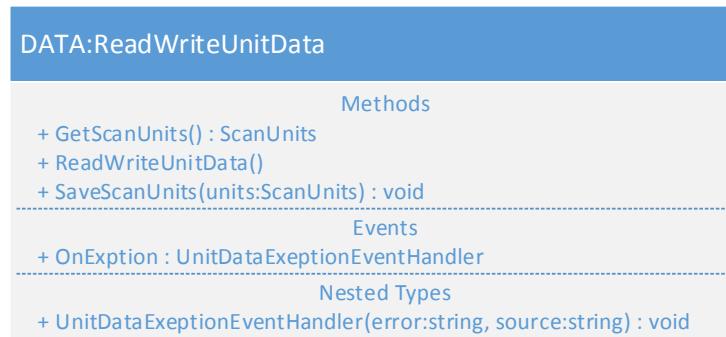
Figur 3.37. Detaljeret specifikation af Modbus

DATA:GeoMagicTouch



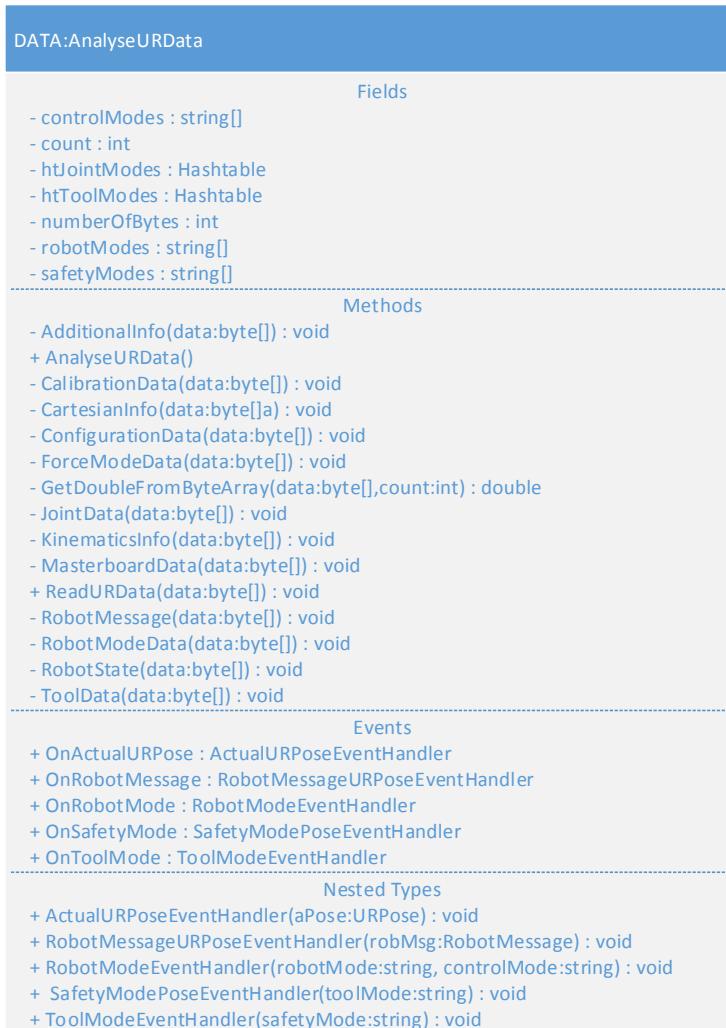
Figur 3.38. Detaljeret specifikation af GeoMagicTouch

DATA:ReadWriteUnitData



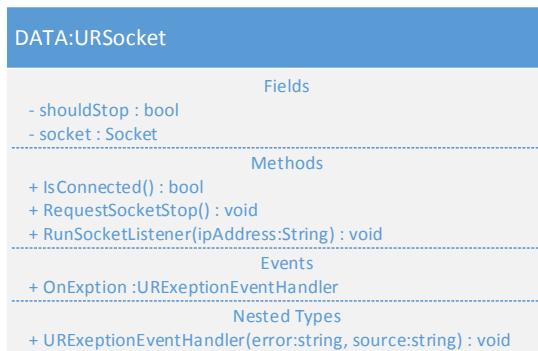
Figur 3.39. Detaljeret specifikation af ReadWriteUnitData

DATA:AnalyseURData



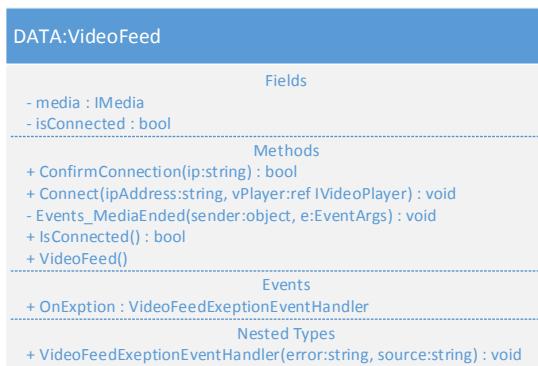
Figur 3.40. Detaljeret specifikation af AnalyseURData

DATA:URSocket



Figur 3.41. Detaljeret specifikation af URSocket

DATA:VideoFeed



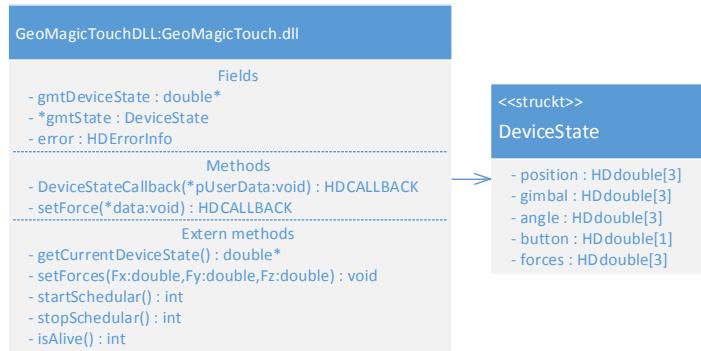
Figur 3.42. Detaljeret specifikation af VideoFeed

DATA:RemoteDesktop



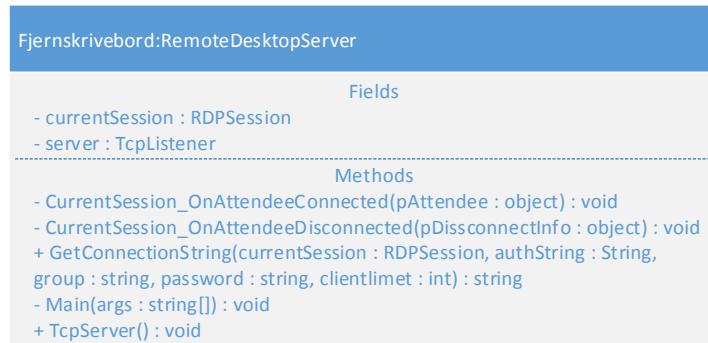
Figur 3.43. Detaljeret specifikation af RemoteDesktop

GeoMagicTouchDLL:GeoMagicTouch.dll



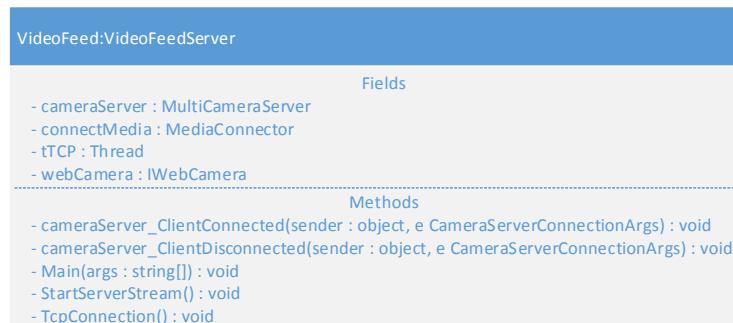
Figur 3.44. Detaljeret specifikation af GeoMagicTouch.dll

Fjernskrivebord:RemoteDesktopServer



Figur 3.45. Detaljeret specifikation af RemoteDesktopServer

VideoFeed:VideoFeedServer



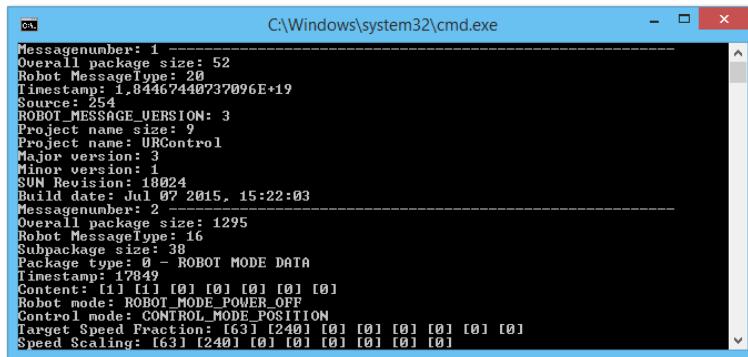
Figur 3.46. Detaljeret specifikation af VideoFeedServer

Unitests 4

Der er udført unitests af de forskellige elementer der indgår i kommunikationen mellem PC applikationen og eksterne enheder i systemet, for at teste systemet i mindre dele før det implementeres i det samlede.

4.1 Kommunikation med UR10

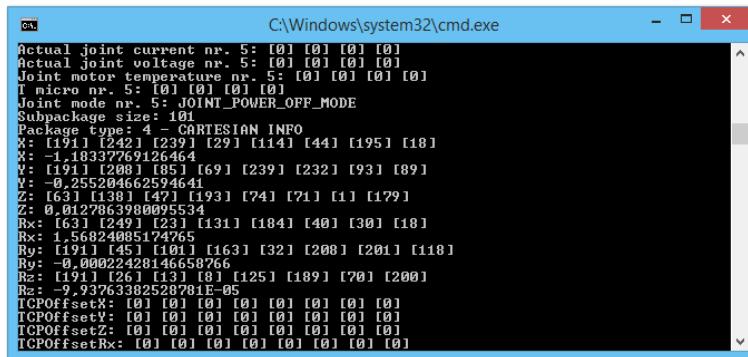
Når der kommunikeres med UR10'eren bliver der sendt data ud på en stream, som skal analyseres så indholdet kan bruges videre i programmet. En unittest viser indholdet af de pakker der modtages. Den første der sendes på streamet når forbindelsen er oprettet er en versionsbesked, som kan ses på figur 4.1. Her kan man se at indholdet fortæller om den installerede software på UR10'eren.



```
C:\Windows\system32\cmd.exe
Messagenumber: 1 -----
Overall package size: 52
Robot MessageType: 20
Timestamp: 1.84467440737096E+19
Source: 254
ROBO_MESSAGE_VERSION: 3
Project name size: 9
Project name: URControl
Major version: 1
Minor version: 1
SUN Revision: 19824
Build date: Jul 07 2015, 15:22:03
Messagenumber: 2 -----
Overall package size: 1295
Robot MessageType: 16
Subpackage size: 38
Package type: 0 - ROBOT MODE DATA
Timestamp: 17849
Content: [1] [1] [0] [0] [0] [0] [0]
Robot mode: ROBO_MODE_POWER_OFF
Control mode: CONTROL_MODE_POSITION
Target Speed Fraction: [63] [240] [0] [0] [0] [0] [0]
Speed Scaling: [63] [240] [0] [0] [0] [0] [0]
```

Figur 4.1. Unitest af modtagelsen af første pakke fra UR10 indeholdende Versionbesked.

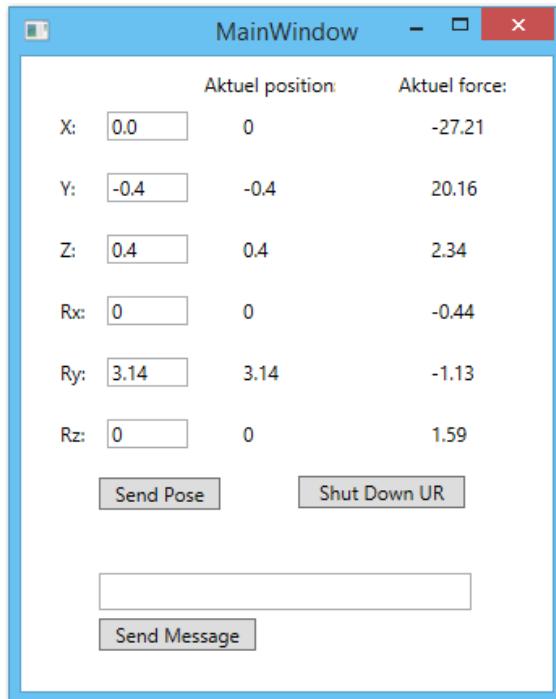
Den efterfølgende pakke der modtages indholder data omkring UR10'ens tilstand. På figur 4.2 kan man se et udsnit af denne pakke, som er underpakke nr. 4 - Cartesian Info og indeholder UR10'ens aktuelle position.



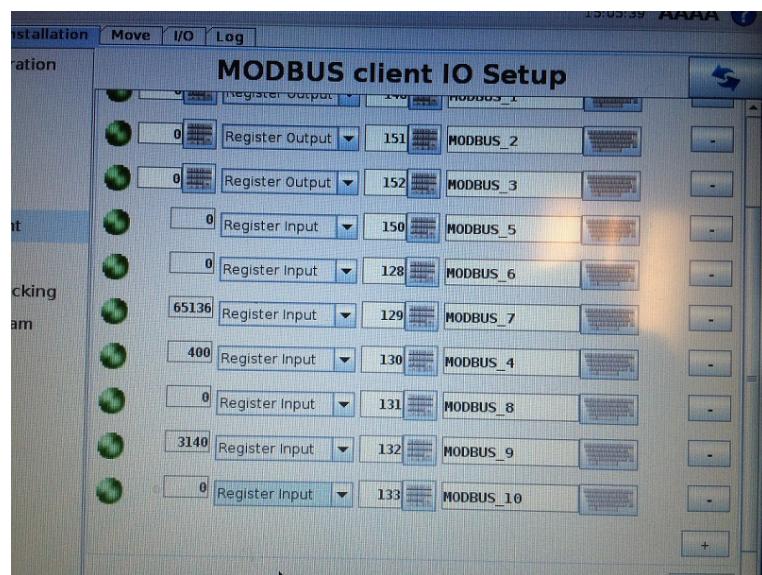
```
C:\Windows\system32\cmd.exe
Actual joint current nr. 5: [0] [0] [0] [0]
Actual joint voltage nr. 5: [0] [0] [0] [0]
Joint motor temperature nr. 5: [0] [0] [0] [0]
I micro nr. 5: [0] [0] [0] [0]
Joint mode nr. 5: JOINT_POWER_OFF_MODE
Subpackage size: 101
Package type: 4 - CARTESIAN INFO
X: [191] [242] [239] [29] [114] [44] [195] [18]
X: -1.18337269126464
Y: [191] [208] [85] [69] [239] [232] [193] [89]
Y: -0.255284662594641
Z: [63] [138] [47] [193] [74] [71] [1] [179]
Z: 0.0127863980095534
Rx: [63] [249] [23] [131] [184] [40] [30] [18]
Rx: 1.56824085174765
Ry: [191] [45] [10] [163] [32] [208] [201] [118]
Ry: -0.002242814665876
Rz: [191] [26] [13] [8] [125] [189] [70] [200]
Rz: -9.93763382528701E-05
TCPOffsetX: [0] [0] [0] [0] [0] [0] [0] [0]
TCPOffsetY: [0] [0] [0] [0] [0] [0] [0] [0]
TCPOffsetZ: [0] [0] [0] [0] [0] [0] [0] [0]
TCPOffsetRx: [0] [0] [0] [0] [0] [0] [0] [0]
```

Figur 4.2. Unitest af modtagelsen af UR10 position via socket.

Når der skal sendes en ny position til UR10'en sker dette vha. modbus. Der er været anvendt et wpf-program til at teste denne funktion. Figur 4.3 viser programmet, hvor det er muligt at indtaste koordinater og sende dem til modbusen. Figur pic:ModtagetPoseModbus viser så brugergrænsefladen på UR10'ens tilhørende skærm, hvor det er muligt at se indholdet på de enkelte registre i modbusen. Her kan det ses at værdien på de to figure er den samme blot med faktor 1000 til forskel, da modbusen arbejder med integers op til 16 bit, for at sikre præcisionen i koordinater.



Figur 4.3. Unitest af at sende en position vha. indtastede koordinater.



Figur 4.4. Billede af UR10'en modbus set på dens egen UI.

Den sidste del der indgår i positioneringen af UR10'en er registrering af forceen på dens Tool Center Point. Et program på UR10'en skriver hele tiden den aktuelle force ind i modbusen,

og figur 4.5 viser så et program der aflæser denne force kontinuert. Her ses det også at forceen variere meget selv om UR10'en står stille i denne test.



```
Connecting to ModBus
ModBus connected
ForceData: [5.0]   [-0.5]   [-4.94]   [-0.72]   [0.39]   [1.44]
ForceData: [2.08]  [-0.3]   [-3.63]  [-0.49]  [0.15]  [1.67]
ForceData: [2.241]  [0.46]  [-0.60]  [-2.65]  [0.61]  [1.84]
ForceData: [3.31]  [0.071]  [-6.59]  [-2.46]  [1.09]  [1.49]
ForceData: [1.471]  [-2.01]  [1.61]  [-3.16]  [-1.1]  [1.66]
ForceData: [2.651]  [-0.59]  [-6.51]  [-2.22]  [1.24]  [1.98]
ForceData: [1.331]  [-2.39]  [1.54]  [-3.22]  [-0.82]  [1.25]
ForceData: [2.191]  [-2.52]  [1.81]  [3.07]  [-0.89]  [1.32]
ForceData: [2.441]  [-0.15]  [-4.85]  [-1.09]  [0.89]  [1.61]
ForceData: [2.651]  [-0.06]  [-3.52]  [-0.59]  [0.32]  [1.93]
ForceData: [1.221]  [-2.13]  [0.37]  [2.29]  [-0.65]  [1.66]
ForceData: [0.961]  [-3.26]  [2.94]  [4.15]  [-1.25]  [1.79]
ForceData: [1.821]  [-1.29]  [-1.88]  [0.83]  [0.05]  [1.81]
ForceData: [1.571]  [-2.92]  [2.81]  [3.86]  [-0.91]  [1.61]
ForceData: [3.1]    [-0.43]  [-6.71]  [-2.65]  [1.25]  [1.49]
```

Figur 4.5. Unittest af den aflæste force fra modbusen.

4.2 Kommunikation med Geomagic Touch

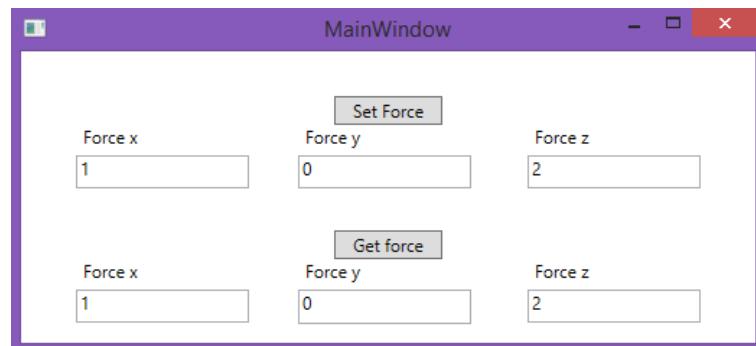
For at kommunikere med Geomagic Touch er det nødvendig at kunne hente og skrive information til devicet. Derfor er der lavet to separate unitest, som er nærmere beskrevet nedenfor.

Nedenstående figur viser hvorledes der er udviklet en consol-applikation med det ene formål at hente information fra Geomagic Touch og udskrive disse i en consol. Der ses på figuren at hver gang der hentes informationer fra Geomagic Touch bliver positionen, angles, buttons og gimbal angles hentet ud.

```
C:\Users\rasmus\Desktop\setForce\Debug\structConsole.exe
*****
This is the DEVELOPER EDITION of OPENHAPTICS, commercial distribution is prohibited
Please contact 3D Systems to obtain a commercial license
*****
Device position: 0.000 -65.511 -88.114
Device Angles: 0.000 0.269 -0.371
Buttons: 0.000
Gimbal angles: -0.001 0.799 0.455
*****
Device position: 0.000 -65.511 -88.114
Device Angles: 0.000 0.269 -0.371
Buttons: 0.000
Gimbal angles: -0.005 0.799 0.468
*****
Device position: 0.000 -65.511 -88.114
Device Angles: 0.000 0.269 -0.371
Buttons: 0.000
Gimbal angles: 0.000 0.796 0.483
*****
Device position: -39.891 45.866 -7.161
Device Angles: 0.243 1.026 0.814
Buttons: 3.000
Gimbal angles: 0.238 -0.231 -0.195
*****
Device position: -53.754 83.011 23.608
Device Angles: 0.273 0.967 1.186
Buttons: 2.000
Gimbal angles: -0.002 -0.839 -0.392
*****
Device position: -56.817 88.079 25.903
Device Angles: 0.285 0.958 1.231
Buttons: 0.000
Gimbal angles: 0.101 -0.876 -0.413
*****
```

Figur 4.6. Unitest af informationslæsning på Geomagic Touch.

På nedenstående figur ses hvorledes der er udviklet et testprogram der udelukkende har til formål at teste forceen på Geomagic Touch. Her er der lavet tre felter hvor der kan indskrives en ønsket force som Geomagic Touch skal anvende og en knap hvorpå der påføres. Derudover er der en knap der henter forceen fra Geomagic Touch og udskriver dem i tre felter. Herefter kan der verificeres at den force der er blevet sat stemmer overens med den udlæste force.



Figur 4.7. Unitest af forceen på Geomagic Touch

4.3 Fjernskrivebord

På nedenstående figur ses et billede af test af fjernskrivebordets server. Her bliver der startet en server, registreret et nyt request om forbindelse, oprettet forbindelse til fjernskrivebordet og til slut frakoblet fjernskrivebordet. På figuren ses at denne syntax foretages flere gange.

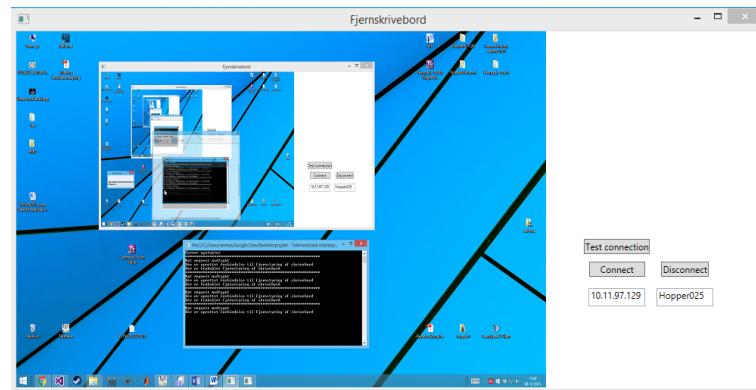
```

file:///C:/Users/rasmus/Google Drev/Bachelorprojekt - Telemedicinsk robotstyring
Server opstartet
*****
Nyt request modtaget
Der er oprettet forbindelse til fjernstyring af skrivebord
Der er frakoblet fjernstyring af skrivebord
*****
Nyt request modtaget
Der er oprettet forbindelse til fjernstyring af skrivebord
Der er frakoblet fjernstyring af skrivebord
*****
Nyt request modtaget
Der er oprettet forbindelse til fjernstyring af skrivebord
Der er frakoblet fjernstyring af skrivebord
*****
Nyt request modtaget
Der er oprettet forbindelse til fjernstyring af skrivebord

```

Figur 4.8. Unittest af server på fjernskrivebord.

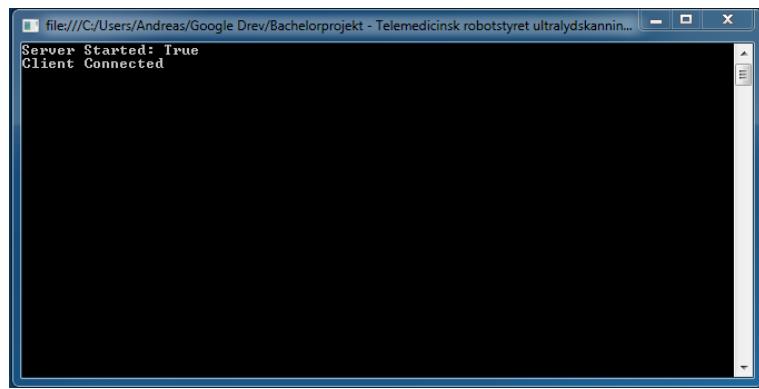
På nedenstående figur ses et billede af testprogrammet til at teste visning af fjernskrivebord. På figuren ses to tekstfelter hvor der kan indskrives ip og lokation. Derudover ses tre knapper hvor den ene tester forbindelsen uden der bliver oprette forbindelse. En anden knap opretter forbindelse til fjernskrivebordet og den sidste afslutter forbindelsen til fjernskrivebordet.



Figur 4.9. Unittest af viewer på fjernskrivebord.

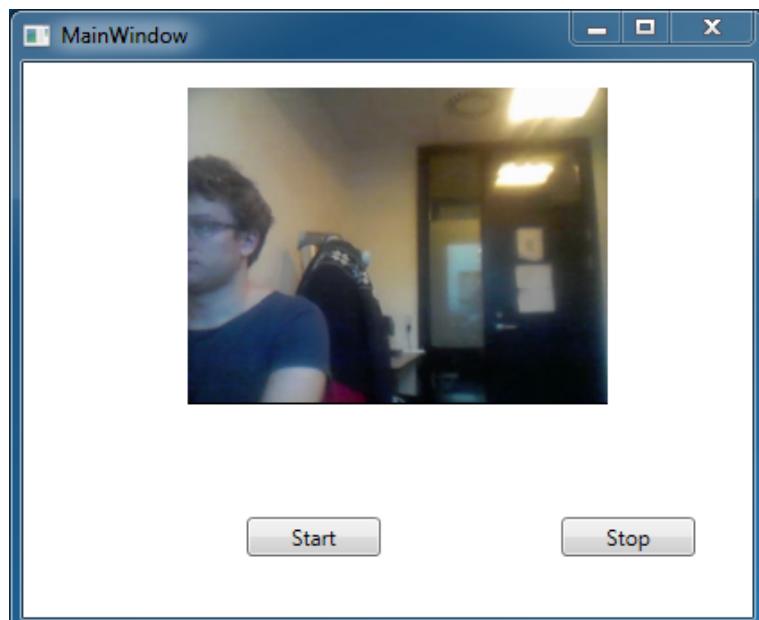
4.4 Videofeed

Der er nedenfor vist et billede af hvordan serveren ser ud når den er opstartet, figur 4.10. Serveren tilkobles et webcam, som den starter. Webcam optager løbende når serveren er startet. Efter opstart af serveren lytter den på et event, for tilkoblede clients. I tilfælde af en client forbinder til serveren, vil videofeedet fra webcamet blive streamet til clienten.



Figur 4.10. Unittest af VideoFeedServer

Der kan nedenstående figur 4.11 vist et billede af det testprogram der er benyttet i forbindelse med unittest af videofeed afspilleren. Det kan ses der er et videofeed kørende fra den VideoFeedServer som clienten er tilsluttet. Der er anvendt et delay på 150ms i afspilleren. Knapperne start og stop anvendes til henholdsvis at starte streamet og stoppe det igen.



Figur 4.11. Unittest af VideoFeed afspiller

Bilag

- 1 - 15138 Telemedicinsk robotstyret ultralydsskanning
- 2 - Feasibilitetsstudie
- 3 - OpenHaptics_ProgGuide
- 4 - OpenHaptics_RefGuide
- 5 - Medical Device Direktiv 93-42-EØF - 2007
- 6 - client_interface
- 7 - Interview Sonograf
- 8 - ModBus Register Overview
- 9 - ModBus server data
- 10 - Modbus_Application_Protocol_V1_1b3
- 11 - scriptmanual_en
- 12 - software_manual_en_polyscope
- 13 - UR10_User_Manual_da_Global
- 14 - Videodemonstration
- 15 - Komplet klassediagram
- 16 - Tidsplaner
- 17 - Milestones template