CPF Software Design Review

Gene Massion

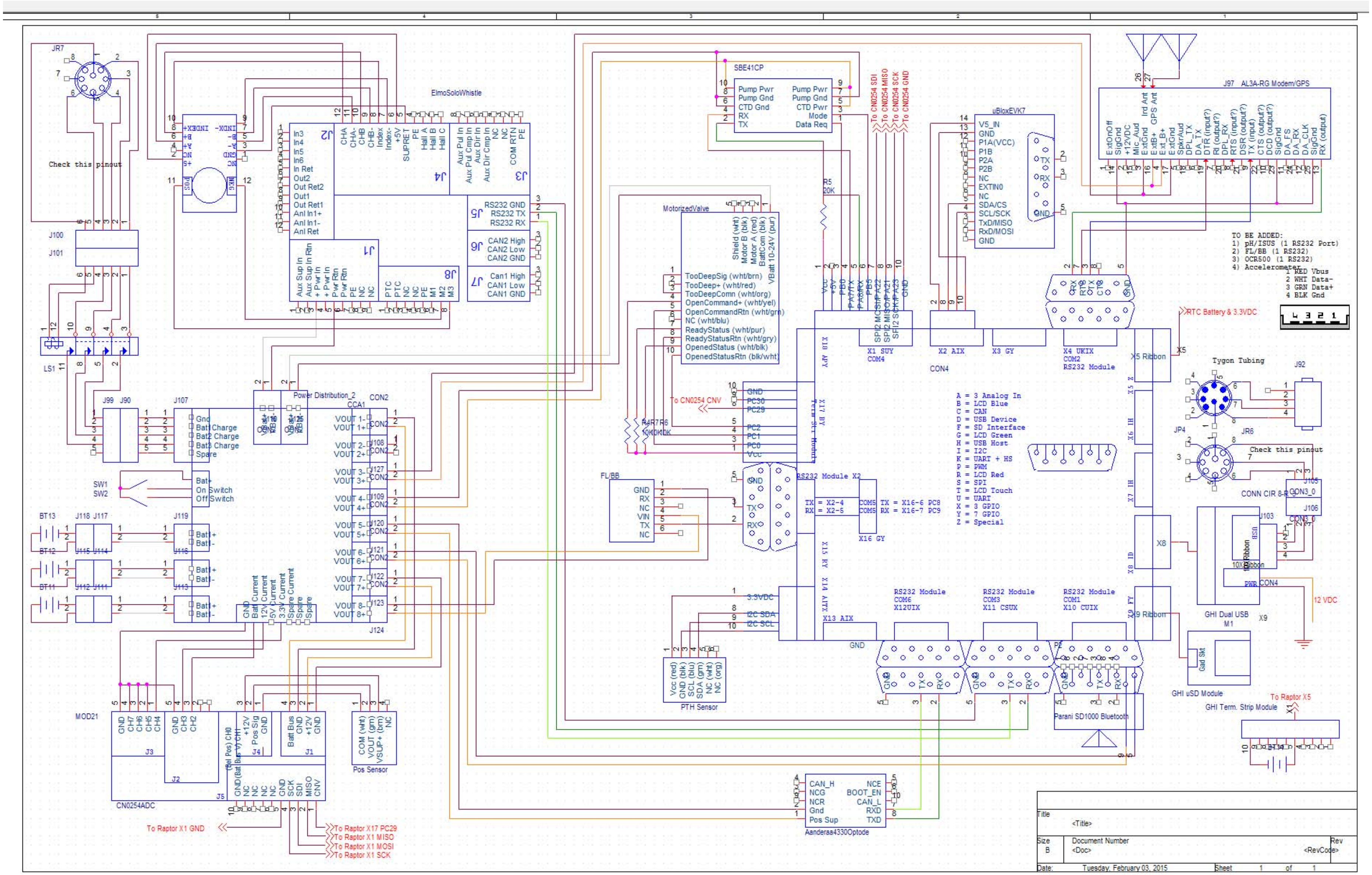
2015 March 13 (Friday)

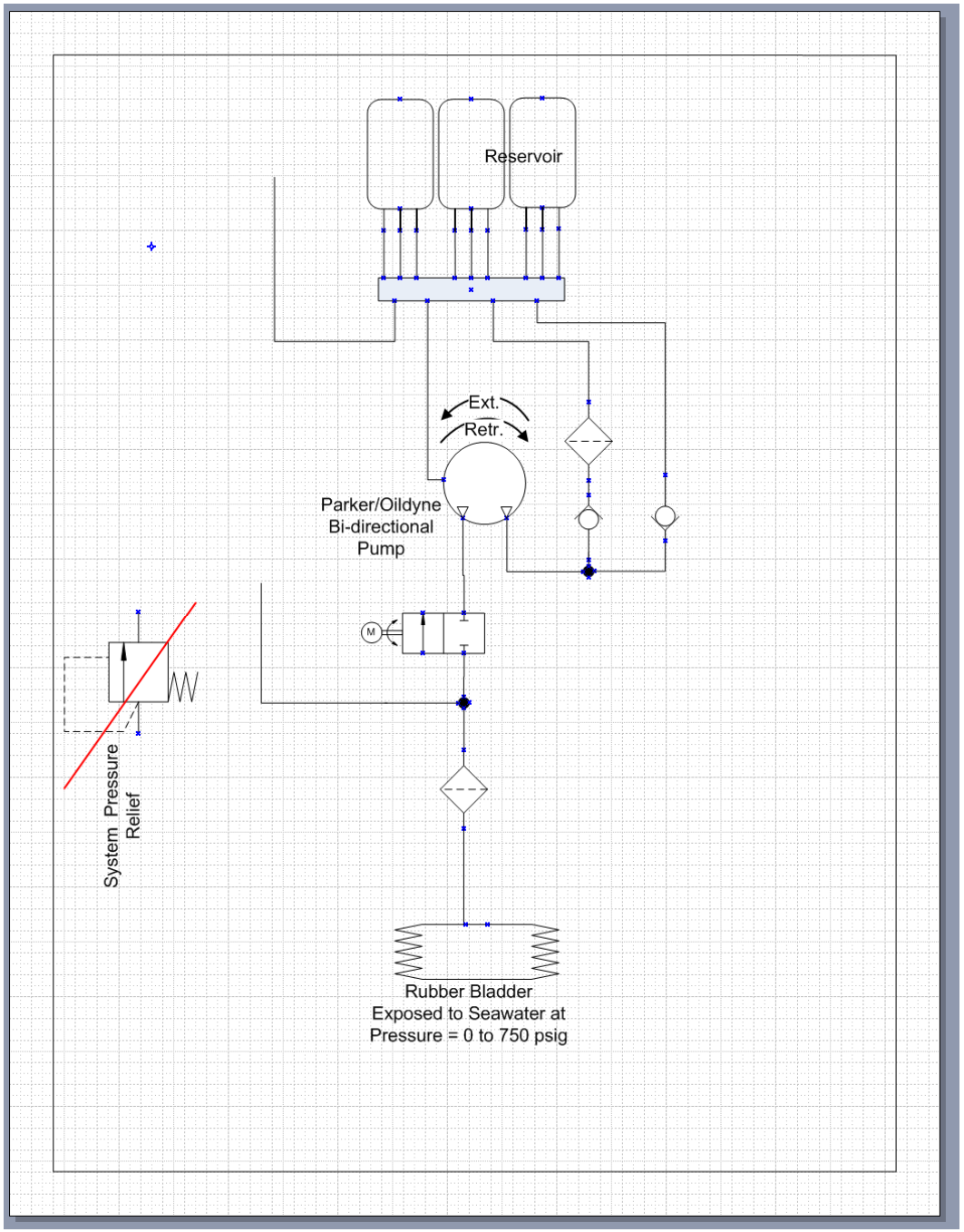
Goals

- Review the architecture and identify areas that need more work
- Pay particular attention to further developing an effective test strategy
- Pay particular attention to the right amount of error detection and correction

Outline

- Quick overview of the system
- Top level scan of the software
- Line by line review of Program.cs
- Overview of the other classes with particular attention to key methods
- We'll keep track of action items as we go through the design review (that would be you Laughlin)


JR7
Check this pinout
ElmoSoloWhistle
SBE41CP
Pump Pwr
Pump Gnd
CTD Gnd
RX
TX
CTD Pwr
Mode
Data Req
To CN0254 SDI
To CN0254 MISO
To CN0254 SCK
To CN0254 GND
uBloxEVK7
J97 AL3A-RG Modem/GPS
TO BE ADDED:
1) pH/ISUS (1 RS232 Port)
2) FL/BB (1 RS232)
3) OCR500 (1 RS232)
4) Accelerometer
1 RED Vbus
2 WHT Data-
3 GRN Data+
4 BLK Gnd
MotorizedValve
TooDeepSig (wht/brn)
TooDeep+ (wht/red)
TooDeepComn (wht/org)
OpenCommand+ (wht/yel)
OpenCommandRtn (wht/grn)
NC (wht/blu)
ReadyStatus (wht/pur)
ReadyStatusRtn (wht/gry)
OpenedStatus (wht/blk)
OpenedStatusRtn (blk/wht)
RTC Battery & 3.3VDC
Tygon Tubing
J92
JR6
Check this pinout
CONN CIR 8-R
To CN0254 CNV
Power Distribution_2
CCA1
CON2
J107
Gnd
Bat1 Charge
Bat2 Charge
Bat3 Charge
Spare
Bat+
On Switch
Off Switch
SW1
SW2
BT13
BT12
BT11
FL/BB
RS232 Module X2
TX = X2-4
RX = X2-5
TX = X16-6 PC8
RX = X16-7 PC9
X16 GY
A = 3 Analog In
B = LCD Blue
C = CAN
D = USB Device
F = SD Interface
G = LCD Green
H = USB Host
I = I2C
K = UART + HS
P = PWM
R = LCD Red
S = SPI
T = LCD Touch
U = UART
X = 3 GPIO
Y = 7 GPIO
Z = Special
X1 SUY
COM4
X2 AIX
X3 GY
X4 UKIX
COM2
RS232 Module
CON4
X5 Ribbon
X8
X9 Ribbon
GHI Dual USB
M1
12 VDC
RS232 Module COM6 X12UIX
RS232 Module COM3 X11 CSUX
RS232 Module COM1 X10 CUIX
X13 AIX
PTH Sensor
Vcc (red)
GND (blk)
SCL (blu)
SDA (wht)
NC (wht)
NC (org)
GHI uSD Module
To Raptor X5
GHI Term. Strip Module
Parani SD1000 Bluetooth
MOD21
CN0254ADC
Pos Sensor
To Raptor X1 GND
To Raptor X17 PC29
To Raptor X1 MISO
To Raptor X1 MOSI
To Raptor X1 SCK
Aanderaa4330Optode
CAN_H
NCG
NCR
Gnd
Pos Sup
NCE
BOOT_EN
CAN_L
RXD
TXD
Title
<Title>
Size
B
Document Number
<Doc>
Rev
<RevCode>
Date: Tuesday, February 03, 2015
Sheet 1 of 1

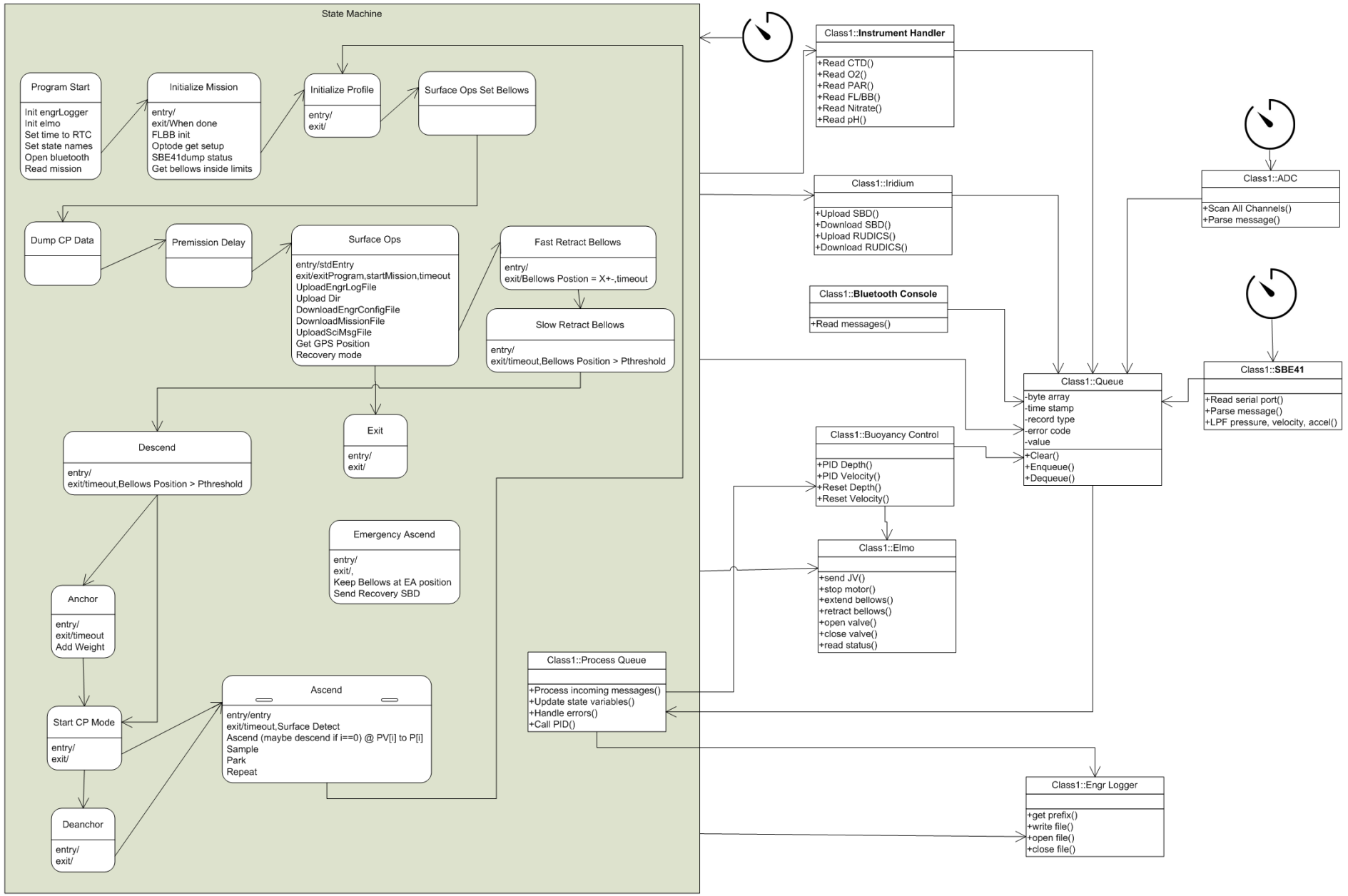

Reservoir
Ext.
Retr.
Parker/Oildyne
Bi-directional
Pump
M
System Pressure
Relief
Rubber Bladder
Exposed to Seawater at
Pressure = 0 to 750 psig

Design “Rules”

- Assume GM is responsible for maintaining the code forever i.e., find the simplest possible solution
- We have a pretty good understanding of the requirements i.e., we don’t need to design for some killer unknown future need
- Everything must run-to-completion
  - No “blocking” calls
  - Short Thread.Sleep(<1 sec) may be acceptable
  - Every state should transition when done or after a defined timeout period
- Log everything and decimate when required
  - Reading and analyzing log file should be nearly effortless

Garbage Collection (GC)

- C# doesn't require (or support) memory allocation/deallocation but it does GC
- GC happens whenever it wants and can take minutes if there is a lot of garbage
- Value types aren't subject to GC (I'm pretty sure)
  - int, float, double, bool, char and struct
- Reference types are subject to GC if they aren't referenced anymore
  - Any variable that requires a "new" definition: arrays, StringBuilder, DateTime, TimeSpan, objects, …
- Declaring a reference type as static prevents GC
- Strings are just plain bad, const or readonly strings are OK
- StringBuilders are OK if they don't have to get redimensioned


State Machine
Program Start
Init engrLogger
Init elmo
Set time to RTC
Set state names
Open bluetooth
Read mission
Initialize Mission
entry/
exit/When done
FLBB init
Optode get setup
SBE41dump status
Get bellows inside limits
Initialize Profile
entry/
exit/
Surface Ops Set Bellows
Dump CP Data
Premission Delay
Surface Ops
entry/stdEntry
exit/exitProgram,startMission,timeout
UploadEngrLogFile
Upload Dir
DownloadEngrConfigFile
DownloadMissionFile
UploadSciMsgFile
Get GPS Position
Recovery mode
Fast Retract Bellows
entry/
exit/Bellows Postion = X+-,timeout
Slow Retract Bellows
entry/
exit/timeout,Bellows Position > Pthreshold
Descend
entry/
exit/timeout,Bellows Position > Pthreshold
Exit
entry/
exit/
Emergency Ascend
entry/
exit/,
Keep Bellows at EA position
Send Recovery SBD
Anchor
entry/
exit/timeout
Add Weight
Start CP Mode
entry/
exit/
Ascend
entry/entry
exit/timeout,Surface Detect
Ascend (maybe descend if i==0) @ PV[i] to P[i]
Sample
Park
Repeat
Deanchor
entry/
exit/
Class1::Instrument Handler
+Read CTD()
+Read O2()
+Read PAR()
+Read FL/BB()
+Read Nitrate()
+Read pH()
Class1::Iridium
+Upload SBD()
+Download SBD()
+Upload RUDICS()
+Download RUDICS()
Class1::Bluetooth Console
+Read messages()
Class1::ADC
+Scan All Channels()
+Parse message()
Class1::SBE41
+Read serial port()
+Parse message()
+LPF pressure, velocity, accel()
Class1::Queue
-byte array
-time stamp
-record type
-error code
-value
+Clear()
+Enqueue()
+Dequeue()
Class1::Buoyancy Control
+PID Depth()
+PID Velocity()
+Reset Depth()
+Reset Velocity()
Class1::Elmo
+send JV()
+stop motor()
+extend bellows()
+retract bellows()
+open valve()
+close valve()
+read status()
Class1::Process Queue
+Process incoming messages()
+Update state variables()
+Handle errors()
+Call PID()
Class1::Engr Logger
+get prefix()
+write file()
+open file()
+close file()

- Line by line review of the main Program.cs and critical classes e.g. EngrLogger, StructQueue, SBE41 and maybe a few others