**Exercise: Hospital Bed, Producer - Consumer**

In these exercises, you will work with C# threads, the producer – consumer design and network communication.

Your job is to design and implement software for a hospital bed.

The caretakers at the hospital have trouble with patients leaving their beds, when they are not supposed to.

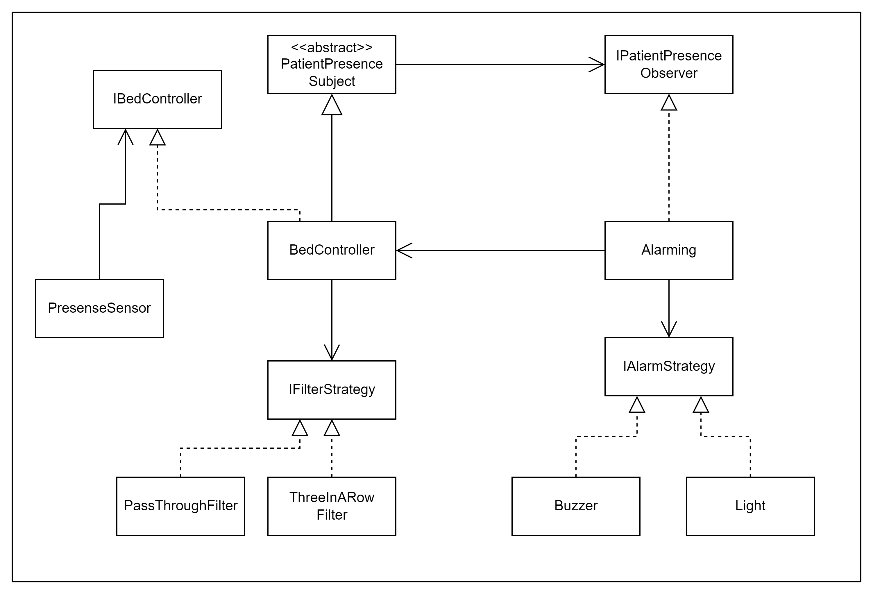
The hospital has hired *you*, a brilliant young engineer, to solve their problem.

You came up with a solution:

Install a presence sensor and a buzzer in the beds.

If the presence sensor detects, that there is no patient in the bed, alarm the caretakers using the buzzer.

In a previous exercise, you simulated the sensor, created a BedController, which received readings from the sensor and used the GoF Strategy pattern to choose how to notify the caretakers. The design probably looked much like this:



We will now focus on the interaction between the PresenceSensor and the BedController to create a more decoupled and robust design.

**Exercise 1:**

The PresenceSensor invokes a method in the BedController, every time a new sensor reading has been made. This could be a problem, because long processing time in the BedController and the rest of the system will delay new readings.

Change the design of the interaction between the PresenceSensor and BedController, so they each run in their own thread and are decoupled by a BlockingCollection.

Create a SensorReading object, which holds information about if a patient is present in the bed or not.

The PresenseSensor should add SensorReading objects to the BlockingCollection and the BedController should take SensorReading objects from the BlockingCollection.

Create a UML class diagram, a UML Sequence diagram and any other diagrams, you find relevant for the design.

Explain your design to one or two of your fellow students.

**Exercise 2:**

Implement your design as a Console application.

**Exercise 3:**

In the real world, the sensor will often be implemented on a small microcontroller, which sends the readings to a more powerful system running on a PC, on which the different alarming types can be controlled.

Modify your design, so the sensor runs as its own console application and sends data to a BedController application, which listens to data on a network port. To start with, you can use localhost and run both applications on the same PC.

Create a UML class diagram, a UML Sequence diagram and any other diagrams, you find relevant for the design.

Explain your design to one or two of your fellow students.

**Exercise 4:**

Implement your design.

**Exercise 5:**

Did you use TCP or UDP for the communication?

Provide the arguments for and against both choices (TCP vs. UDP).

**Exercise 6:**

Create a wifi-hotspot and run the Sensor application and BedController application on two different PC’s.