Assignment 2: Software Engineering – Mine Pump Control System (MPC).

Problem Scenario

You have been tasked to develop the architecture for a mine pump control system, designed to monitor and pump flood water out of mine shafts. As underground mining operations take place far below the water table, flooding into mine galleries and shafts is an ever-present danger. Excessive flooding is clearly a safety hazard for workers, but also has profitability implications ranging from equipment damage to productivity delays, to mine closures in extreme circumstances.

The system to be developed will be required to monitor the water level in a given mine shaft using two sensors. A high water sensor that measures the maximum acceptable level of flooding in a shaft before pumping begins, and a low water sensor, which measures the minimum level of acceptable flooding and pumping stops. These sensors are used to start a mine pump. When the flooding level exceeds the level determined by the high water sensor the pump is switches on. When the water has been pumped out and the minimum level of acceptable flooding has been reached, as measured by the low water sensor, the pump switches off.

In addition to flooding mining is often hindered by methane pockets, where gas seeps into the shafts and galleries triggering an evacuation. Again, this is a safety hazard, the mining staff won't be able to breathe, and even more critically, operating equipment may generate sparks which will cause the methane to ignite. Therefore, the system will include a methane sensor that will be used to trigger an evacuation alarm in the presence of dangerous levels of methane (measured in N parts per million), and also switch off the pump regardless of the current water level.

The system is used by two key roles. The first is the Operator. This role is required to log in to the system with a username and password. Following a successful login, the operator is able to start or stop the pump if, and only if, the water level is between the high and low sensor limits. The second role is the Supervisor. A supervisor must verify their security credential as per the operator above. Following a successful login, they are able to switch the pump on, or off at any time. For example, a supervisor could run the pump after the flood level has dropped below the level set by the low water sensor. They could also switch the pump off if the water level goes over the maximum high level of flooding. In these cases, the supervisors' actions override the automatic behavior of the pump. A supervisor is required to "reset" the pump system in order to re-establish automatic behavior.

Finally, to meet Federal monitoring standards a persistent log is required to capture the following events:

- Pump switched on by high water sensor
- Pump switched off by low water sensor
- Pump switched on or off by operator or supervisor

- Evacuation alarm triggered by methane sensor
- The reading of the methane sensor every 30 minutes

The reading of the methane sensor (for the last 24 hours) can be read by the operator. All readings (up to 30 days) can be read by the supervisor. The supervisor also has the capability to add a “note” to any specific log event that occurs within 24 hours.

Complete the Following Software Engineering Tasks

1. Your company has decided to utilize “best practices” in requirements elicitation and iterative development practices. As a consequence, for the problem scenario described above, perform a problem analysis and capture high-level requirements to be saved in a Requirements Specification Document (in MS Word). See the SRS Template. Items 2 and 3 below will be inserted into the SRS Document.

2. Perform requirements model engineering and specify a set of essential and real use cases describing the functionality that your business solution/system will exhibit. To start, identify goals/sub goals, actors and their dependencies, and high-level tasks. Perform scenario modeling and group scenarios as needed to identify essential use cases. Using essential/real Use Case diagrams identify the following:

- Actors – who/what interacts with the business solution/system?
- Use Cases – what behaviors does the business solution/system exhibit?

For each use case you have identified, fill out the Use Case Specification (Description) Template, one for each use case. Document all of the Use Case Specifications/Descriptions using MS Word. This will become the reference listed in the SRS, item 1.3. Use the following template for each Use Case Description.

Use Case Specification (Description) Template

Use Case ID: *{This should be coded to identify the level of the use case}*
Use Case Name: *{Short descriptive phrase}*
Relevant Requirements: * *{Reference to relevant requirements document.}*
Primary Actor: *{Main sub-system/entity that initiates use}*
Pre-conditions: *{Requirements on the state of the system prior to this use being valid.}*
Post-conditions: *{This describes the state of the system following the successful completion of this use. Effects on other systems and actors may also be described.}*
Basic Flow or Main Scenario: *{Numbered flow of events: 1 The user initiates an action by... 2 The system responds by...}*
Extensions or Alternate Flows: *{This section presents variations on this use case. It presents those use cases that have an extends relation with the current use case.}*
Exceptions: *{This section describes all error conditions that can arise in the use case.}*
Related Use Cases: {use cases that are either usually performed just before or after the current use.}

*Note: Establish proper requirements traceability to the Use Case requirements. Integrate the Use Case in the template with the Requirements Specification Document.

3. Draw the UML Use Case Diagrams using a UML modeling tool of your choice. You may also draw them by hand. Document all of the UML Use Case Diagrams using MS Word. This will become the reference in the SRS, item 1.3.

4. Brainstorming:
Brainstorm a list of object candidates that model objects you consider key abstractions necessary to solve the problem.
Consider the following questions:

- "What do I know"? – what data will the object candidate encapsulate?
- "What do I do"? – what methods (functions) will the object support?

5. Create Class Diagrams. Use the set of object candidates identified in step 4 and develop a set of class diagrams that will document the classes and their relationships in a more formal manner. Diagram all identified classes. This will become the reference in the SRS, item 1.3.

6. Create Sequence Diagrams: Use the flow you have identified (in step 4) at the Use Case and CRC level between objects and develop a set of sequence diagrams to represent the flow of control from a given actor(s) and its execution of the methods you have identified with your emerging model.

At the completion of this assignment you will have the following to submit:

SRS Document – Step 1
Use Case Specification Document – Step 2
UML Use Case Diagrams Document – Step 3
Class Diagrams – Step 5
Sequence Diagrams – Step 6

The above documents should be merged into a single document to submit.

Upload your ORIGINAL work to Canvas before the due date and time. No Google, Cheg, 'collaborations', or any unoriginal work will be accepted.