Question 1 – 8 Points.

There are many cases of ambiguity and incompleteness:

**Ambiguity:**
1. If the altitude cannot be determined—how do we “determine” that
2. The detection of a fault—what kind of fault are we talking about
3. Do we turn the do on when we are at the threshold or do we have to be below.
4. Do we indicate a fault after exactly 2 seconds or after two seconds have passed.
5. Etc.

**Incompleteness:**
1. What do we do if two events come in at the same time? This argument could be stretched into several points by comparing different events.
2. The effect of the inhibit on the system in various situations is not clear (at least not to me). For example, what do we do if a fault is detected when the ASW is inhibited and later the inhibit is removed?

Question 2 – 12 points

We accept both requirements and specification statements.

**Functional Requirements Statements:**
1. The apartment must sleep a family of 4.
2. The apartment must be wired for at least 1mbs Internet connection.
3. The dining area must accommodate thanksgiving dinners for at least 12.
4. Etc.

There could be some argument about what is functional versus non-functional in this context. Here are some requirements one could argue are non-functional.
1. The apartment must not be on the ground floor.
2. The apartment must be no more than 5 miles from <work address>
3. The walls must be painted blue.
4. Etc.

We will also accept more specification-oriented statements. For example,
1. There must be four bedrooms
2. The kitchen must have a Y cubic feet convection oven
3. Etc.
Question 3 – 8 points

**Proposed Solution:**
As a client, you naturally want to nail down as many details as possible so you know what you will be receiving from the contractor—this may lead to a severe over-specification of the system.

As a contractor, you want a less detailed requirements document to give you the freedom you need to design and implement a cost effective (for you) product.

Naturally, both these somewhat conflicting goals may lead to conflicts in the process. It does not matter who “wins”, we will most likely have a document that is either too detailed or not detailed enough. Or, we have a document that is a mixture of both—some parts are severely over specified and others are woefully incomplete.

Full score for a convincing discussion, even if it does not bring up this exact point.

Question 4 – 8 points

**Suggested Solution:**
Errors of omission are very difficult to catch. Thus, including a TBD makes it explicit that you have thought of an issue, but you have decided to defer the decision to later. Now it is easy to find and dismiss the TBDs at a later stage.

**Grading:**

Question 5 – 18 (6+12) points

**No solution provided.**

Question 6 – 16 Points.

**Suggested Solution:**
The guarding condition to set the DOI_Command to PowerOn clearly blocks this from happening if we have received a notification that the DOI is on---the condition “DOI_Status is not equal to On” assures this since it is part of a conjunction.

**Assumptions** 1 and 2 guarantee that if the DOI has been powered on, we will know about it. Thus, if the DOI has been powered on, there is no possibility that we will attempt to power it on again. What remains to be determined is what happens at startup when we do not know if the DOI is on or off?
We will turn the DOI on if its status is Unknown. To satisfy Safety Requirement 1 we must assure that if we attempt to turn the DOI on when we do not know its status, it is turned off. Therefore, we need at least one extra assumption:

**Assumption 3:** At startup, the pilot must make sure the DOI is Off. Thus, we can assume that the DOI is Off until the ASW is notified otherwise.

Now we can complete the argument.

In the initial state the DOI will be off according to Assumption 3. Therefore, it is OK to turn the DOI on if DOI_Status is either Off (we know the DOI is off through Assumptions 1 and 2) or UNDEFINED (we know the DOI is off through Assumption 3).

### Question 7 – 10 points

**Suggested Solution:**

*Alternative 1* seems more suitable for this task. The model is closer to the problem at hand and, thus, more likely to be understood by the customer. *Alternative 2* is more abstract, but it relies on some features of UML that are not entirely trivial to understand. For example, the fact that we may have two tanks, one monitoring the other, is not really clear from the model. An instance diagram might solve this problem for the customer.

### Question 8 – 20 points

**Suggested solution:**

1. How many alarm logs are there in the system? - 1
2. More than one operator can operate the switch at one time (true or false)? - False
3. How many operators can review the log? - 1
4. How many switches does the operator have to operate? - 2
5. How many pumps do we have in the fuel system? - One or more
6. Did this analyst model both the large and the small tank (yes or no)? - No
7. How many “pieces”(objects) is an instance of the Fuel System composed of? – 3 or more
8. What kind of sensor is in the system (Bar Sensor, mmHg Sensor, both types, Cannot Tell)? - Cannot tell
9. The system could have both a visible and audible alarm (yes or no)? - No
10. How many switches do we have (total) in this system? – 2