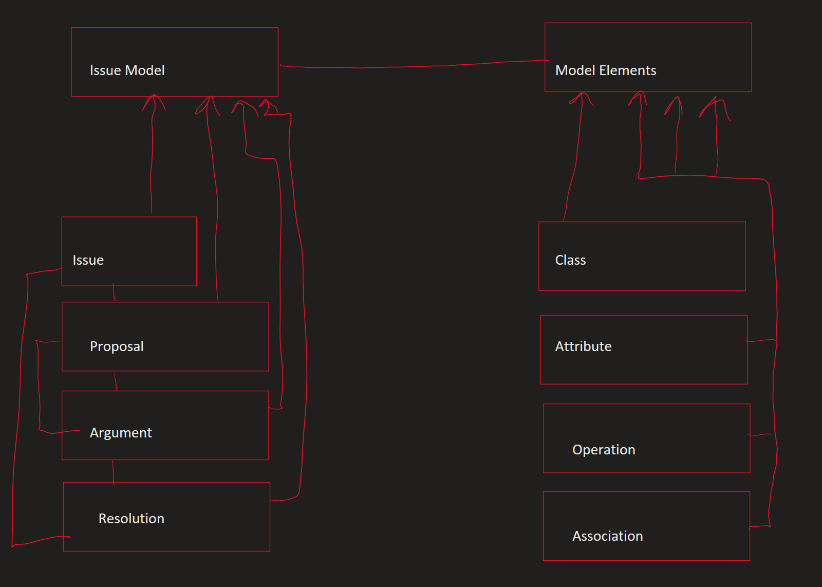
12-2 You are developing a UML modeling tool. You are considering the integration of

rationale into the tool. Describe how a developer could attach issues to different model

elements. Draw a class diagram of the issue model and its association with model

elements.

There are several ways a developer could attach issues to different model elements. Here are a couple examples: first being issue tracking. You could integrate with systems like GitHub for fast and issue tracking of the software development. To build on that, you can integrate issue creation. To continue in the context of GitHub, issues can be raised from the tool and assigned appropriately from there. Another example to allow developers to navigate between the different model elements for different issues. Able to view the linked issues quickly for further investigation.



12-3 Below is an excerpt from a system design document for an accident management

system. It is a natural language description of the rationale for a relational database for

permanent storage. Model this rationale with issues, proposals, arguments, criteria, and

resolutions, as defined in Section 12.3.

* Issue: Fundamental issue in database design is the database engine realization
* Proposals
  + Object oriented database as the engine
  + Relational database as the engine
  + Using a file system for specific data types as the engine
* Arguments & Criteria
  + Advantage of OO DB
    - Ability to handle complex data relationships
    - Buzzword compliance
  + Disadvantage of OO DB
    - Sluggish performance with large amounts of data
    - Integration issues with COBRA
  + Advantages of Relational DB
    - Robust engine
    - Higher performance characteristics
    - Large pool of experience and tools
    - Integration with CORBA
  + Disadvantages of Relational DB
    - Limited support for complex data relationships
  + Advantages of file system
    - Easy archival solution
    - Ability to handle large amounts of data
  + Disadvantages of file system
    - Code must be written from scratch, including serialization of access
* Resolution
  + Based on the requirement to use COBRA and simplicity of relationships between the systems’ permanent data, the decision is made to use a relational database design.

14-10 Linux, developed using the bazaar model, is more reliable and more responsive than

many operating systems running on Intel PCs. Make a case why the bazaar model

should or should not be used for the Space Shuttle control software.

I would say that, in the context of the Space Shuttle control software, one should NOT choose to use the bazaar model with a Linux system. My main reasoning for this is the fact that it is open source. While open-source development can be faster, and often times provides better work, the possibility for error (accidentally or otherwise) far out ways any benefits. When it comes to human life and hundreds of millions of dollars, any person working on the project should be appropriately vetted. With open-source software, all it takes is one person with malicious intent to cause harm on a global scale, in the context of the Space Shuttle.

14-11 Organize the project participants into teams of four people. Each team has the

following resources available: 2 eggs, 1 roll of TESA film, 1 roll of toilet paper, a cup

with water, a bucket with two liters of sand, 20 foam balls (each about 1 cm diameter),

1 table whose surface is about 1 meter above the ground. Each team has 25 minutes

time to build and test an artifact that allows an egg to be released 75 cm above the table

such that the egg falls on the table without cracking. Each team has another 5 minutes

to demonstrate the artifact to project management.

From slack: “What I envision for this question is you have 4 people that will be performing this egg drop task and demoing to the Project Management board. What roles would you want filled by the 4 people in the hypothetical group.”

For me, the roles being assigned would be highly dependent on my knowledge of the background of the 4 participants as individuals and as a group. For example, if I knew that the 4 participants had worked together before, and worked well together, I would not feel it necessary to assign tasks to anybody because they, as a team, already know each other’s strengths, weaknesses and so on. With this knowledge of each other, they would already know who is doing what and what to do individually. Now, to flip this, if this is the first time these 4 people are meeting with no prior knowledge of each other, that changes the tactics a bit. That said, I would probably skip the managerial role for the 4 participants. In a group setting like this, there should not be a single person making decisions. The ideal setting would be group collaboration and a consensus on decisions. That said, with there only being an even number of people, this can lead to a “voting” issue delaying progress on the project. Which is why, this hypothetical project having a short time limit, I don’t think that will play too much into it (yes, there can still be differences of opinion causing delays but there isn’t much choice. This can tell a perspective leader if this is the right set of team members). Essentially, in this example, I would like to see the concept of egoless programming at work. No wone person is more important than the other, the team fails and succeeds together.