CmpE 202 – Software System Engineering

Problem Statement for Team Project (3)

(Where: Problem Statement = a Set of User Requirements Template)

Themes:

- Social Network
- Human Rights
- Ethics
- Content Management
- Industrial Design
- Smart Robots
- Consumer Report
- Business Rules
- Calligraphy
- Unified Software Engines
- E-menia

Electronic submissions only. Check submission Guidelines:

Criteria

When submitting a problem (set of user requirements) description, you should use the following structure as a guide and write your problem statement with the following challenges in mind:

1. No computer of any kind in the analysis phase.
   - No user interface
2. Three or more roles or two or more actors or both
3. At least 10 – 12 Entities (System Classes)
   - Classes = Data + Operations
4. Three to Five operations per Class
5. 10 to 12 Use Cases and make sure two or more roles per use case.
6. Significant
7. Block Diagram with 4-6 blocks where Blocks = Subsystems
8. Unlimited number of scenarios
9. Final products are very important and must be thought out during the analysis &
design phase and lead to s/w architectures on demand including testing activities

10. Synectics for the chosen scenario --

For each problem statement provide the following:

• Title, Team Name, Team members and their e-mail, semester, course # & section #
• Abstract
  This concise presentation of your problem will be the basis for the participants to choose
  the problem they want to work on.
• Introduction (1)
• Description of Problem domains (2)
  This description can be accompanied by pictures, images, text or whatever. The purpose
  of this description is to explain the context in which the problem is placed. The text
  should define any terminology and should be understandable by laymen. All
  abbreviations must be explained, if possible in a glossary.
• Block diagrams (3)
  A block diagram of the problem domain, or any other type of diagram which may serve
  as a basis for the analysis and design, should be included here.
• Description of the program that is wanted (Optional) (4)
  This should include the goals of the program and the required functions. It should state
  what the program should achieve without enforcing any solutions.
• Detailed functional requirements (Hint: Sub-headers = Block Names) (5)
  Each team must create a more detailed 2+ page of detailed functional requirements.
  Because of time limitations these should of course be restricted. An indication of
  performance and capacity requirements should be given.
• Detailed non-functional requirements (6)
  Identify the ultimate non-functional requirements or the best quality factors of the system
  that will be derived from the functional requirements.
• Synectics (7): Describe briefly applications or case studies of expected outcome
  architecture.
• Use cases and User Context (8)
  Each team must provide description of what your users do, how their need for the system
  arose, and what they would like to see to solve their problem. Use your imagination and
  put yourself in the user’s context. Describe all the use case in your system or meaningful
  handful of use cases. Use cases are well suited to provide hints for both the analysis and
  design processes, as design teams tend to work on them first. Description of the work
  context for the user/customer’s point-of-view: why a system is needed, and what the
  system will do for them. Remember you need to come up with 10-12 significant use
  cases.
• Interfaces (9)
  Descriptions of interfaces to other systems. This is of course optional when no interfaces
  to other systems are necessary.
• References for further study (10)
  These are references to articles, other similar systems or implementations.
Sample Problem Statements or Requirements: Check OOPSLA’s DesignFests

http://designfest.acm.org/Problems/Welcome.htm &

Please remember that your team’s problem statement will be subject to submission to OOPSLA ’07 DesignFest. Check:

http://www.oopsla.org/2006/designFest.html

The following properties should be considered when creating your problem statement:

1. **Uniqueness**: new and no duplicates
2. **Originality**: Creative effort invested by an author into the requirements that gives them a new quality or character. The problem statement contains indicators for freshness of aspect, design, or style.
3. **Completeness**: The state of being complete.
4. **Readability**: The ease in which any party can read and understand the problem statement.
5. **Creativity**: Artistic or intellectual inventiveness. We can define creativity as innovation which refers to the introduction of novel things.

**Grading Criteria at the end of the semester as part of the final project**

<table>
<thead>
<tr>
<th>Item</th>
<th>% of Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Illustrations</td>
<td>20%</td>
</tr>
<tr>
<td>Completeness of requirements</td>
<td>35%</td>
</tr>
<tr>
<td>Organization</td>
<td>15%</td>
</tr>
<tr>
<td>Creativity</td>
<td>30%</td>
</tr>
</tbody>
</table>

1. **Illustrations**: This refers to any illustrated models, such as pictures, images, diagrams, that provide clear definition of the problem domain and/or system architecture. **Properties: Readability and Creativity.**
2. **Completeness of requirements**: This refers to how completely the group has described the user context and requirements. Make sure all the possible scenarios or use cases are covered, including error recovery. For example, a requirement document for an e-mail system that leaves out the ability to reply to mail messages would be downgraded (of course, it may be a requirement that the user cannot reply to mail -- but that needs to be part of the requirement). **Properties: Completeness.**
3. **Organization**: This refers to how well-organized and readable the requirements document is. If related requirements are strewn throughout the document, or the document is written poorly, it will be downgraded. Good organization that clarifies the requirements is desired. **Properties: Readability.**
4. **Creativity**: Creative requirements will be rewarded. Try to come up with some good ideas that fit the user's context. **Properties: Uniqueness, Originality, and Creativity.**