**Assignment 4**

|  |  |
| --- | --- |
| Scenario name | Set\_watch\_2\_min\_ahead |
| Participating actor instances | vk:WatchOwner |
| Flow of events | 1. The WatchOwner presses both Watch buttons simultaneously. 2. The Watch enters “set time” mode and indicates this by blinking the hour digits. 3. The WatchOwner presses the left button once. 4. The Watch stops blinking the hour digits and starts blinking the minutes. 5. The WatchOwner presses the right button twice. 6. The Watch increments the minutes by two. 7. The WatchOwner presses both buttons simultaneously. 8. The Watch stops blinking. |

1. **Multiple Choice Questionnaires**

Multiple choice questionnaires are the ways of getting the requirement from the user. In this the user is given a set of questions with each one of them having several choices. The user must select one choice out of the given ones to specify his requirement.

**Drawbacks of Multiple-Choice Questionnaires**

• **Domain of the answers:** The analyst is required to be enough familiar with the application domain to offer the best set of answers for each question.

This means the analyst must know all possible solutions for a given problem so that the user does not have to bother about finding the suitable choice. If the user does not find any appropriate choice, he might answer with a choice that does not reflect the problem properly and the requirement might be misinterpreted.

• **Bounded choices:** Sometimes it does not matter how efficiently the analyst has selected the choices of questions. There remains the chance for something that user would like to have opened for any answer.

For example, if the question is about a user interface, various users might have their own choices for interface. It is not possible for analyst to enumerate all of them. Even in this case, multiple choice questions lack the transparency.

Hence, for certain cases the multiple-choice questionnaires cannot be a good choice to get the user requirement.

1. Strengths of users:
   * They usually have detailed knowledge of the problem that needs to be solved.
   * They have detailed knowledge of constraints imposed by the environment on the possible solutions.
   * They have detailed knowledge of the application domain.

Weaknesses of users:

* They usually have poor knowledge of the possible solutions.
* They usually have poor knowledge of the formal languages for describing the problem or the solutions.

Strengths of developers:

* They have detailed knowledge of different possible solutions.
* They have detailed knowledge of formal languages to describe the problem or the possible solutions.

Weaknesses of developers:

* They (initially) have poor knowledge of the problem to be solved.
* They can make incorrect assumptions about the problem based on their prior knowledge of different problems.

1. The client contracts the development of a system. The client is responsible for defining the scope of the system and for financing the project. The end users are the participants who uses the system to accomplish their work. In some project, the role of end user and client may be shared by the same participant. In general, however, this is not the case.

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Client** | **End user** |
| 1. | Client is the customer for whom the software is developed by the development team | The end-user is the one for whom the client gets the software prepared. |
| 2. | A client can be a person or a co-operative group. | The end-users are the person the ones who use the software once it is released. |
| 3. | The client needs to specify their software requirements and specifications to the development team. | On behalf of end user, client tells the need of end user to the development team. |
| 4. | The client is the one who pays for the development of the software to the development team/company. | End users do not pay any money to the development team. |

1. A role is assigned to a participant according to their skill and expertise.

* Change a subsystem interface to accommodate a new requirement. [system architect]
* Communicate the subsystem interface change to other teams. [API engineer]
* Change the documentation because of the interface change. [editor]
* Design a test suite to find defects introduced by the change. [tester]
* Ensure that the change is completed on schedule. [project manager or team leader]