

Lecture No 10

Prototyping and GUI Design

Adding user interface details in the SRS is controversial. The opponents of this argue that by adding GUI details to the SRS document, focus shifts from what to how – GUI is definitely part of the solution. On the other hand many people think that, it is still what not how and hence it should be made part of the SRS document. By adding the GUIs in the FS, requirements can be solidified with respect to scenario contents. It is my personal experience that the client appreciates more the contents of the SRS document if our SRS document contains the GUI details than if we don't have them there. This document is also going to be used as the base line for design, user manual, and test planning among other things. Presence of the UI details imply that these activities can start right after SRS is accepted and signed-off. Emergence of rapid GUI drafting tools has made the task a lot simpler than it used to be. Exploring potential user interfaces can be of help in refining the requirements and making the user-system interaction more tangible to both the user and the developer. User displays can help in project planning and estimation. A user interface might highlight weaknesses in addressing some of the non-functional requirements (such as usability), which are otherwise very hard to fix later on. If you cannot freeze the RS until UI is complete, requirement development process takes a longer time. However, we need to be very careful when we use GUIs to the SRS document. It is a very common mistake to use UI layouts as substitute of defining the functional requirements. We must remember that these are supplementary information and cannot replace other components of the SRS document. Any change in the requirements entails a change in the UI. If the requirements are not stable, this can mean a lot of rework.

Motivation for GUI

- System users often judge a system by its interface rather than its functionality
- A poorly designed interface can cause a user to make catastrophic errors
- Poor user interface design is the reason why so many software systems are never used

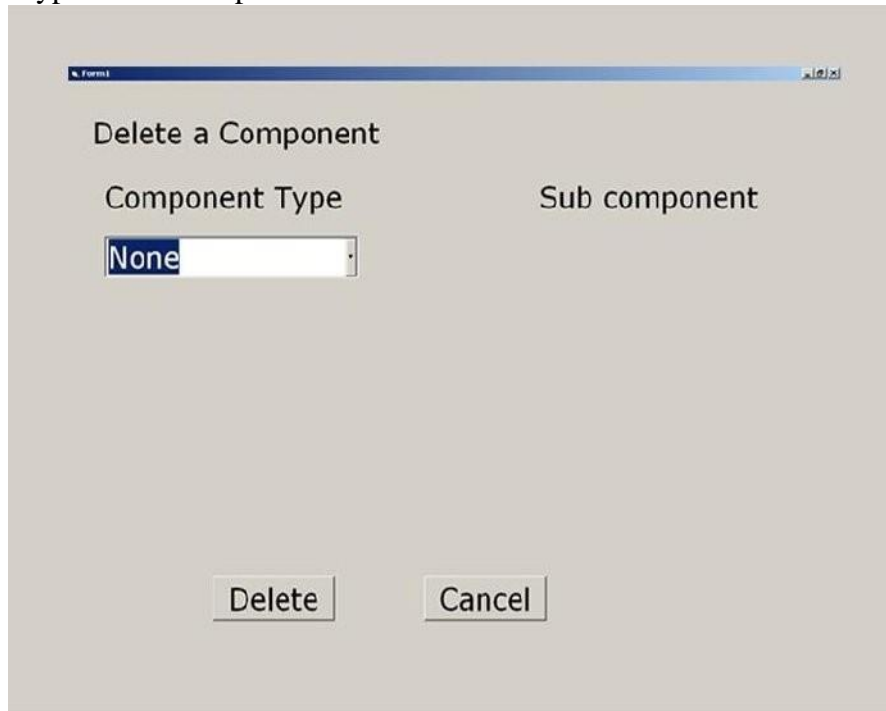
Pitfalls of using GUIs in Functional Specifications

- UIs distract from business process understanding (what) to interfacing details (how)
- Unstable requirements cause frequent modifications in UIs
- An extra work to be done at the requirement level each time a GUI change has to be incorporated

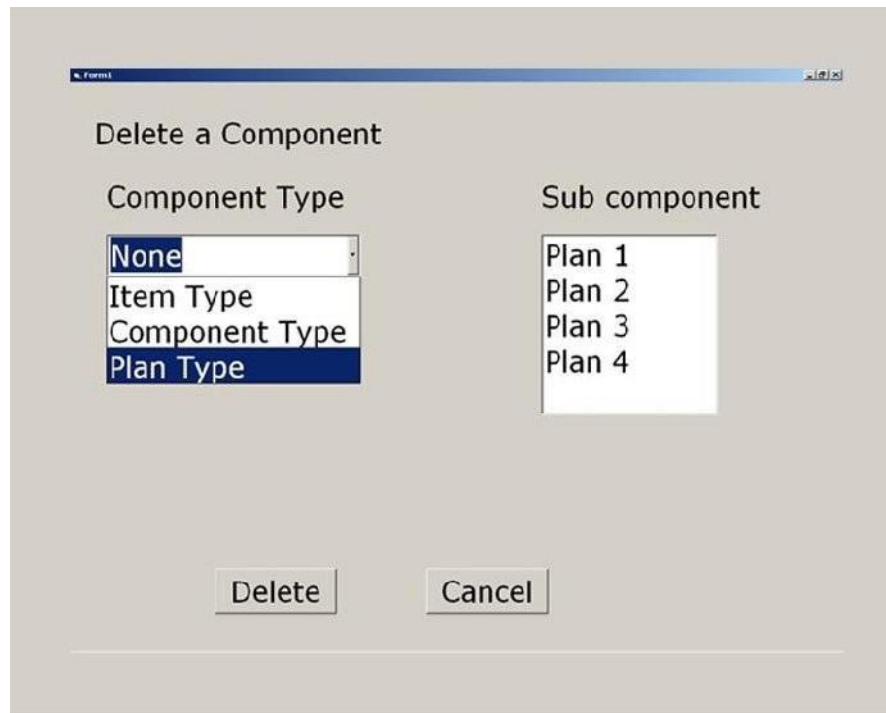
In the following we shall discuss how unstable requirements cause difficulties in preparing GUIs

Example

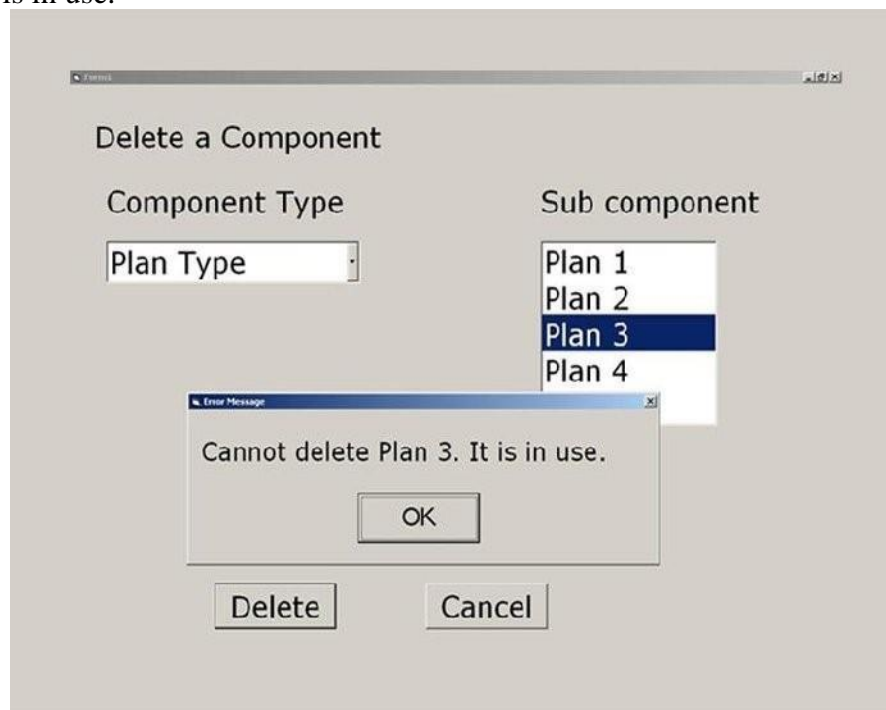
The following GUI implements the delete component use case that we discussed in use case section. The GUI displays a drop down list box that contains a list of component types. The top of the list entry is 'None' where the user can click on the arrow and select the component type whose component he wants to delete.



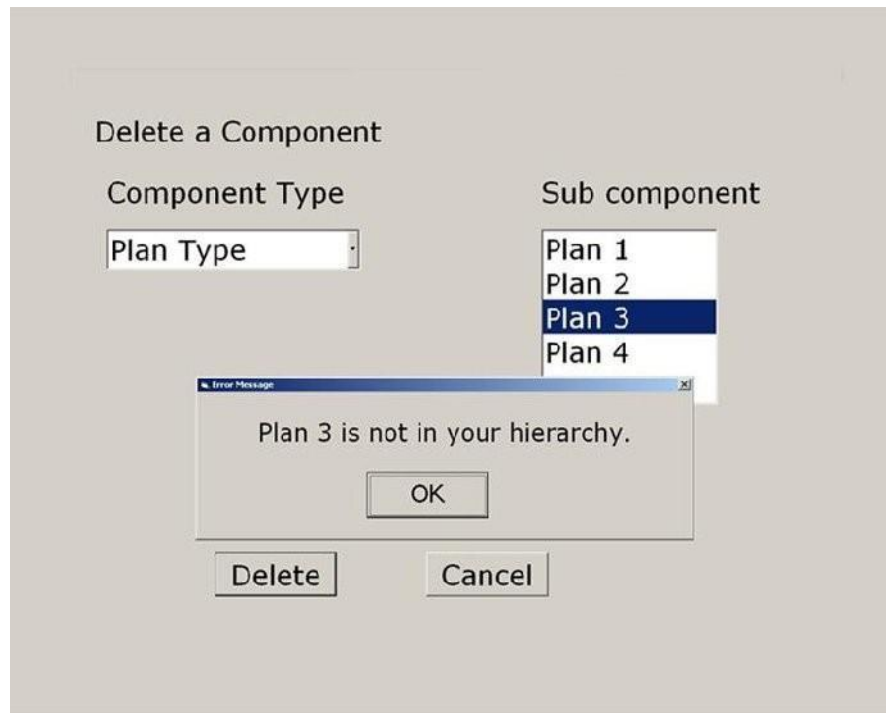
The next GUI implements the scenario when user has clicked over the arrow and a few component types are populated in the list. User then selects a component type 'Plan Type'. Corresponding plans are populated and displayed in the list box at the right side of the GUI.



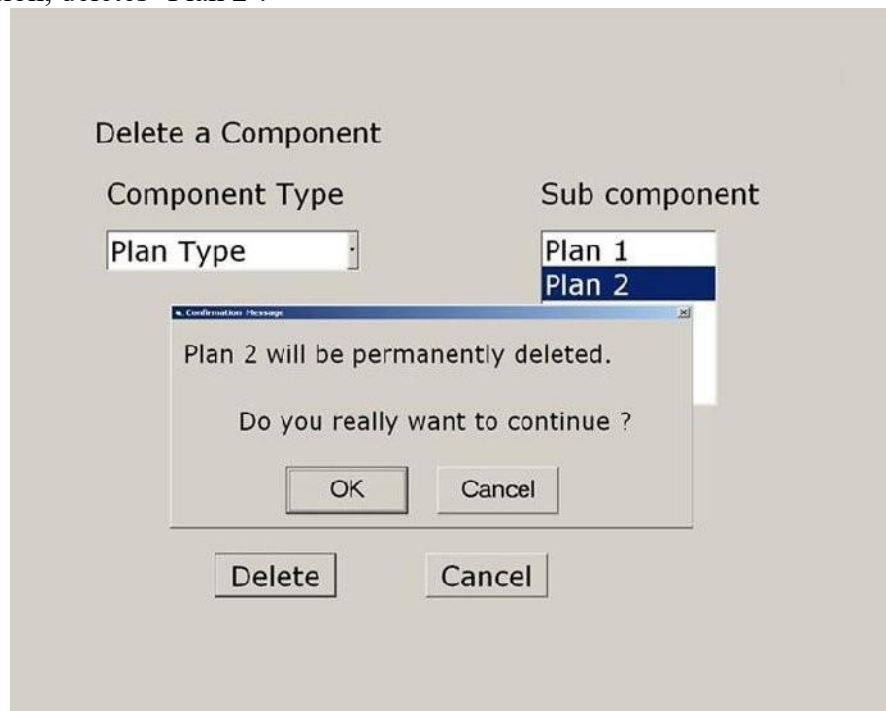
Following GUI depicts the scenario when user selects a particular plan 'Plan 3' and clicks on the 'Delete' button. Now assume that 'Plan 3' is currently being used. So, the application displays a dialog box to the user informing him that he cannot delete this plan as it is in use.



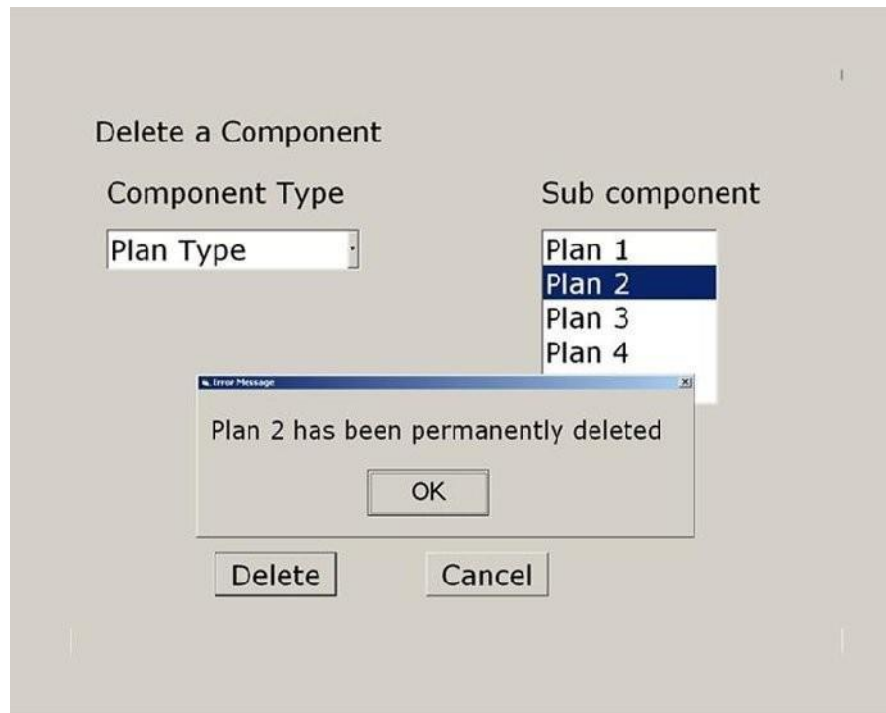
The next GUI, another dialog box is shown in which user is getting another message from the system. It says that Plan 3 is not in his hierarchy.



The user then selects 'Plan 2' and deletes it. System confirms the user and upon confirmation, deletes 'Plan 2'.

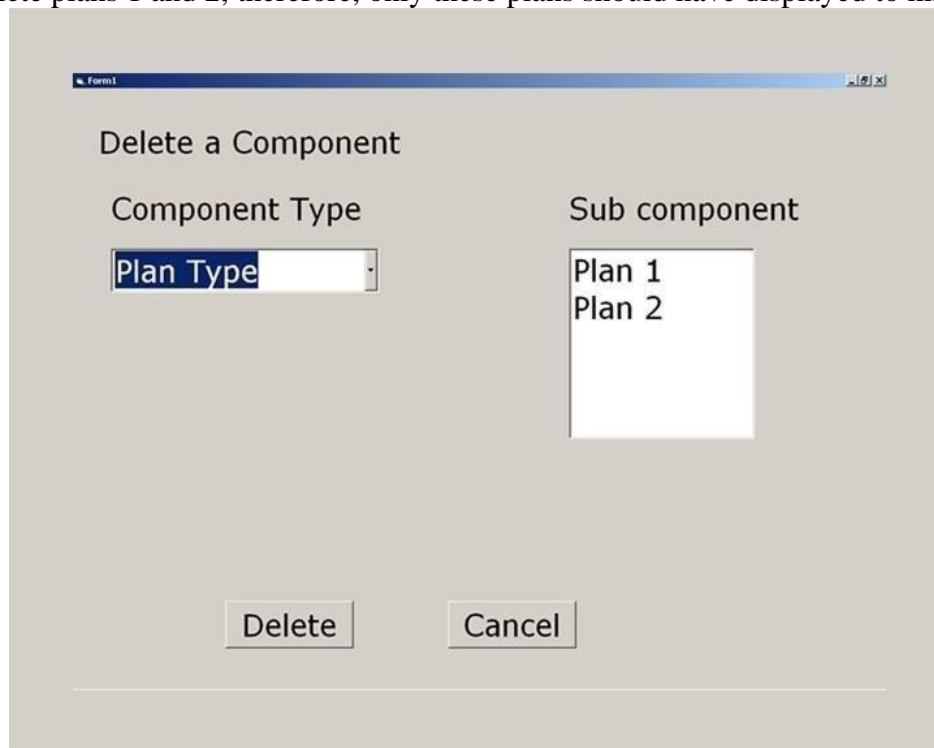


After deleting 'Plan 2', it displays the message that Plan 2 has been permanently deleted. Whereas, 'Plan 2' is still visible in the list.



However, it should be noted that, all the above GUIs presented two major mistakes about the GUIs. First, if a plan is currently in use, it should not have been displayed in the list at the right. Secondly, instead of displaying two messages separately in two dialog boxes, it would have been appropriate to combine them in one message.

The following GUI displays what this GUI should have displayed ideally. As, user can only delete plans 1 and 2, therefore, only these plans should have displayed to him.



In the above example, it is evident that if requirements are partially generated a number of changes have to be made and sometimes the frequency of these changes rise so much that it takes all of the requirements and design time just in finalizing GUI

Prototype

Prototyping is yet another technique that can be used to reduce customer dissatisfaction at the requirement stage. The idea is to capture user's vision of the product and get early feedback from user to ensure that the development team understands requirements. This is used when there is uncertainty regarding requirements. Sometimes, even the customer does not know what he/she actually needs. This happens when there is no manual solution.

A prototype is not the real product. It is rather just a real looking mock-up of what would be eventually delivered and might not do anything useful. However, the presence of a prototype makes a new product tangible. It brings use cases to life and closes gaps in your understanding of the requirements. From a user's perspective, it is easier to play with a prototype and try it out than to read SRS.

