6
Prototype evaluation

The steps described so far in Chapters 4 (Drawing the interface) and 5 (Drawing the behavior) relate to the **designer's** perspective, the one who creates the prototype. However, the sketched mockup should also be evaluated according to the **end user's (client's)** perspective.

In order to do so, UISKEI provides a **simulation mode**, which presents a functional version of the prototype that the user can interact with, in a similar fashion to the final product. The prototype keeps its sketchy look-and-feel, since studies show that testing with rough prototypes does not interfere with the discovery of usability problems when compared to more finished ones (Lin, Thomsen, & Landay, 2002). This early stage prototyping can help designers obtain invaluable feedback, with potential value in the later stages as well (Hundhausen, Balkar, Nuur, & Trent, 2007).

A login screen example can be seen in Figure 28.When the user hovers the pointer over an element, it becomes blue, to give feedback about where the pointer is. In the example, it is possible to see that when the user clicks on a textbox, a text input field appears, allowing the user to enter any text. The `TextEntered` event is only triggered when the element loses focus, so the pattern matching only occurs after all text is written and considered finished. At this time, there is no special treatment for the textbox input. For instance, there is not a flag indicating whether the textbox is a "password textbox". In the example, the text entered in the password field was actually a sequence of asterisks.

Then, when the user clicks on the "Groups" dropdown, the dropdown items appear. Again, the `SelectedItem` event only happens after the user makes a selection. Finally, when the user clicks on the "Login" button, it triggers the `Clicked` event, which is associated to an action to display a default message.

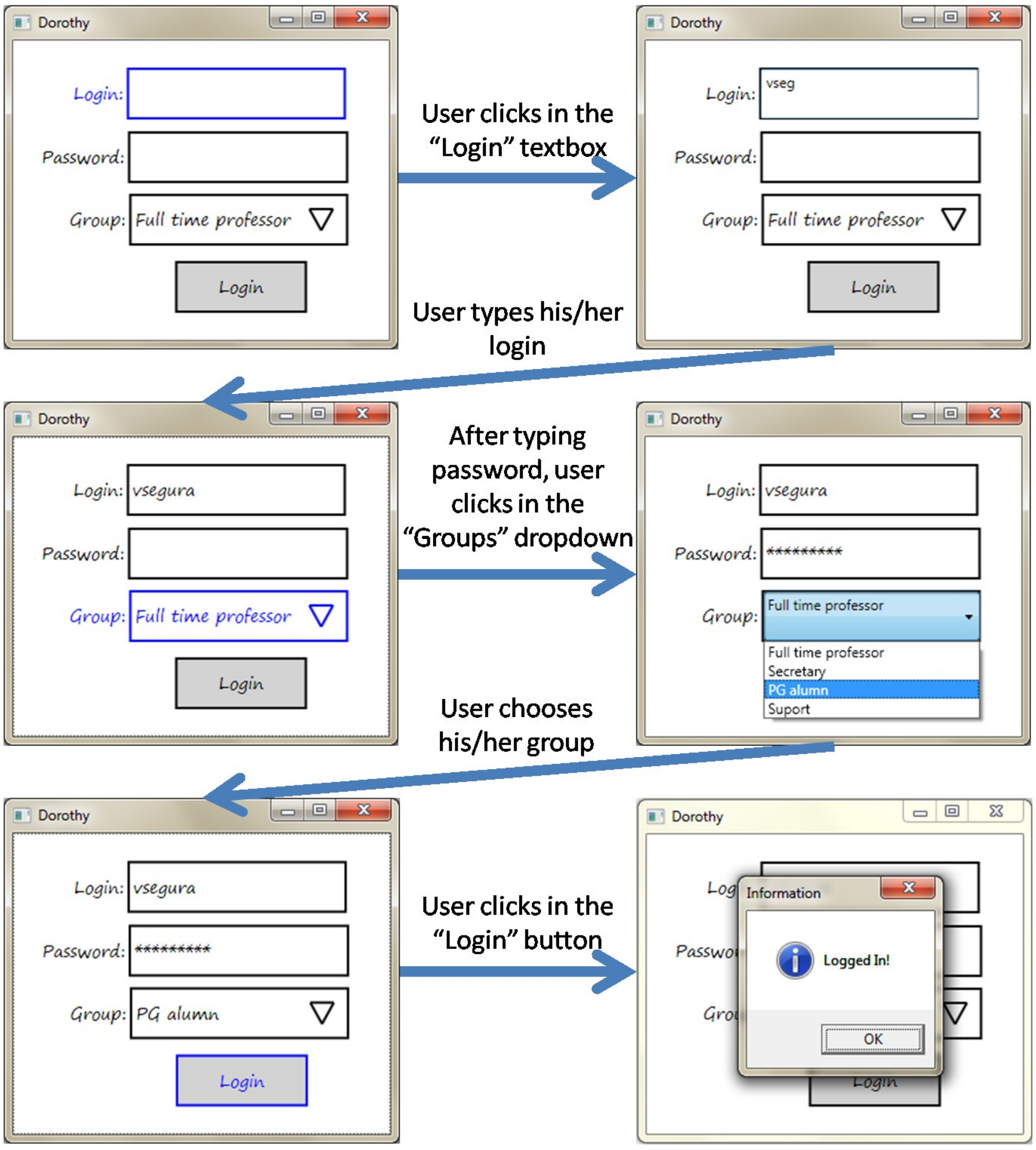
Dorothy
Login:
Password:
Group: Full time professor
Login
User clicks in the "Login" textbox
Login: vseg
User types his/her login
Login: vsegura
After typing password, user clicks in the "Groups" dropdown
Password: *********
Full time professor
Secretary
PG alumn
Suport
User chooses his/her group
Group: PG alumn
User clicks in the "Login" button
Information
Logged In!
OK

Figure 28: Prototype evaluation example.

Each simulation has its own manager, `SimulationManager`, so it is possible to run more than one independent simulation at the same time. This `SimulationManager` references the current's project `ECAMan`, so if changes are made to ECAs on the designer view during the simulation, they are reflected in the simulation. This was planned as a basis for a future Wizard of Oz approach, which will be discussed later in Section 9.4. Besides updating `ECAMan`, it could also reference the current presentation unit, so changes in the elements' list could be reflected on-the-fly.

Despite being a functional prototype, UISKEI lacks some basic features, such as switching the focused widget with the `TAB` key, for example. At the present time, only the events discussed in 5.1.1 are handled. Similar to the element descriptors, the simulation representation is hard-coded according to the event type. If new widgets were added, that share the same already existing events (for example, a list should also have the `SelectedItem` event), this simulation dynamics and representation should also be defined in the descriptor as well. Another reason for delegating this to the descriptors is because a widget can handle more than one event, depending on how or where it was activated. For example, a numeric spinbox can have the `TextEntered` event, when the user inputs the value through the keyboard in the text input area, and the `ValueChanged` event, when the user changes the value by using the up/down arrows.

As could be observed, the simulation does not follow the pen-based interaction paradigm and rather falls in a mouse-based paradigm. Although this may be a valid critique, UISKEI's main intent is to explore the pen during the designer's phase, not during the prototype evaluation. So, the paradigm break is acceptable, particularly because it happens together with the designer/end user perspective change.