Garfixia Software Architectures

Presentation-Abstraction-Control

This architecture is a further development of the Model-View-Controller architecture. The MVC is restricted to simple GUI's with one or more views on the same model. If the model consists of substructures that all require they own special way of interaction, a more complex GUI architecture is in order. The PAC architecture does not have the model as its core component, but a hierarchical structure of PAC components. Each PAC component consists of these items:

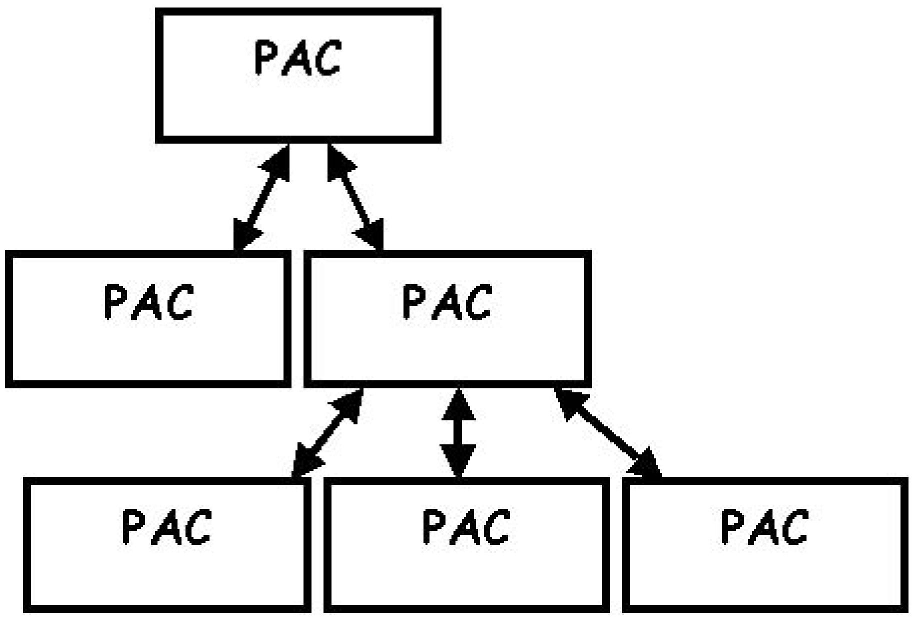
1. Presentation
2. Abstraction
3. Control

Control is somewhat similar to the Controller in the MVC architecture. It processes external events and updates the model. It also directly updates the Presentation part. Yet it is different from the C in MVC in that it passes the changes being made to its parent PAC component.

Abstraction contains the data, like in MVC. However, it may be just part of the complete data structure of the application, and it does not play an active role in the notification of changes.

Presentation is exactly like the View of MVC. It displays the information from the Abstraction.

PAC components are connected in a hierarchical fashion, thus:



Examples

- Most modern compound GUI applications are *loosely* based on this architecture. However, the architecture is in practice usually much more complicated.

Where does it come from?

Joëlle Coutaz first described it in a 1987 article (see Links).

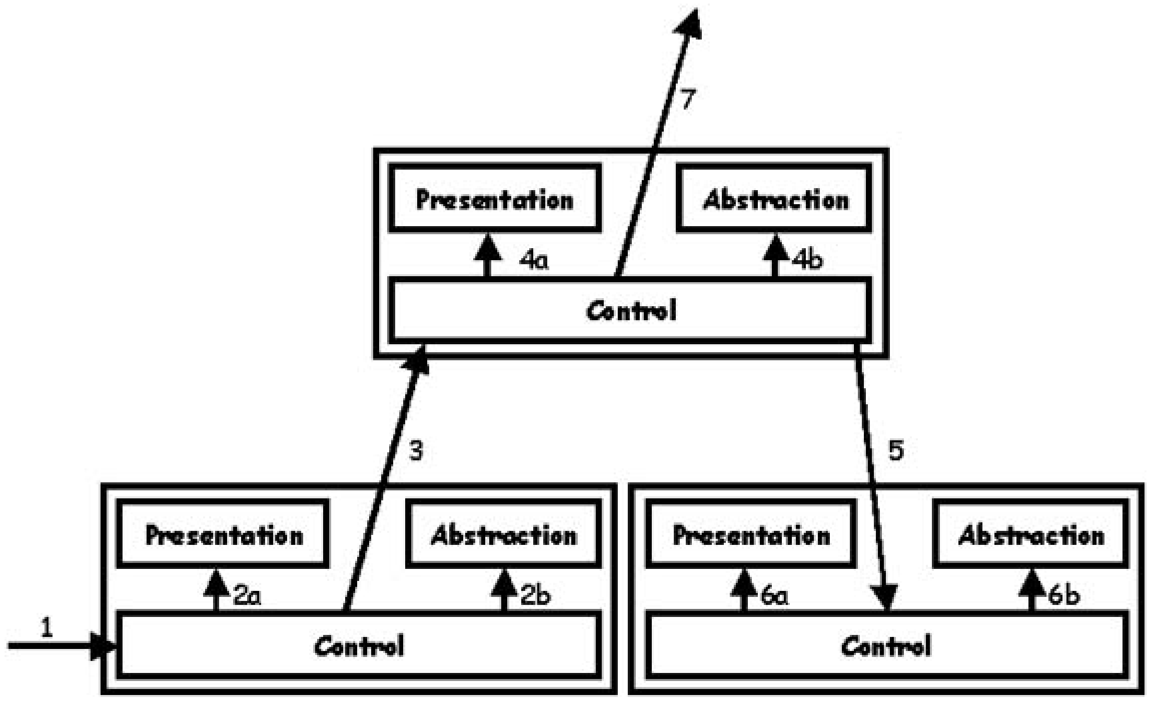
When should you use it?

Use it to get a first idea of how GUI's *should* be built. Use it if you create an entirely new GUI framework.

How does it work?

The parent Control creates its child PAC elements, either at program startup, or dynamically at run-time.

When the control of a PAC element receives a (user) event (1), it may update its Presentation (2a) and/or its Abstraction (2b). Then it sends a change event to its parent (3). The parent updates its children (but not the child where the change originated) (5), which all update their Presentation (6a) and/or Abstraction (6b). After the children have been updated, the parent is updated (7). This ends when all necessary PAC elements have been updated.



Children and parents may send very specific update events to their neighbors. That way, the PAC elements may decide the extent of the effect of the change. Small changes need not be propagated through the entire hierarchy.

Problems

The existing visual programming tools are somewhat related to this architecture, but have all kinds of quirks and exceptions. So you may try to recognize the architecture in visual tools, but don't try to hold on too much. Also, most tools claim to be based on the MVC architecture, which isn't exactly true either.

Common implementation techniques

- The Control is modeled by the Mediator Design Pattern.
- The Presentation is modeled by the Strategy Design Pattern (?)

Links

- Coutaz, J. PAC, an Object Oriented Model for Dialog Design. In Rullinger, H. I. and Shackel, R. (eds), Human-Computer Interaction - INTERACT ’87. Elsevier Science Publishers, 1987, pp 431-436
- Pattern: Presentation-Abstraction-Control
- Presentation-abstraction-control Pattern
- From Single-User Architectural Design to PAC*: a Generic Software Architecture Model for CSCW
- IntrosPAC In French, but look at the video.