Identified Pain Points

User Problematics List

This page lists the "user problematics" or "pain points" identified through the walkthrough exercises and from other sources. Where possible, each item in the list should link to a description of the problem it represents, any principles that apply, and the process through which it was identified -- e.g. the walkthrough exercise that revealed it. If there is a relevant Component Solution, that too should be referenced.

Everyone is invited to contribute to the list. While desirable, descriptions are not required.

My distillation of initial pain points - note that this is "to some extent" from a developer perspective, and is not from a walkthrough, but just "off the top of my head":

- **Preferences**: The preferences system is currently "hard-baked", and has a hard dependency on each application which exports preferences. As a result there are really only about 4 apps which have bothered to do this since each further one creates a dependency risk and involves committing to core code. From the UI perspective, other than the simple inhibition on expressing any UI for preferences at all, we have the risk that preferences cannot be reaggregated. It **should** be possible for them to appear both contextually at the point of application, **as well** as in some central aggregated ("synoptic", that dirty word) overview of a single users entire preferences. And, of course, from the developer perspective should be hosted in the tool/component itself, rather than in any kind of central "Preferences Service".
- **Permissions**: The permissions system similarly has a single "one-size-fits-all" view full of a huge heap of tickboxes. These cannot be placed "close to the tools", unless the tool defines within itself a completely custom view, which once done, conversely cannot be dispatched back to a central location! Steve Githens has been doing work on a "permissions helper" but the scope of this should ultimately be expanded.
- **"Resources"/Portal**: "Resources" has to be the central target. The best we have been able to do so far in "reaggregating" is the system of "helpers", whereby one tool briefly takes completely control of the render pane, until such time it decides it has finished at which point it dispatches back. These things are generally so arcane and nonstandard that again few people have managed to create any unless impelled by really strong "pain points" of design. Each one in general uses its own URL mapping and back-dispatching strategy... I am expecting that one of the core deliverables we will construct in Fluid is an exhaustive catalogue of the different UX patterns that could result from "includable" units of resource choosing, resource presenting, resource manipulating applications. An ultimate goal is to "fold" the portal navigation structure to more closely resemble the resources structure. What is currently the Resources "Tool" should be capable as acting as both consumer and exporter of UI estate. "Tools" should appear within Resources, and "Resources pickers/viewers" within tools. Right now we can do neither.

Portal Views

Portal views refer to the main ways in which the portal contents are displayed to the user.

Definitions

Portal
An aggregator and distributor of pages and tiles based on user identity.

Tile
The abstract concept of a channel or portlet; the most primitive box of content.

Page
A page is a container of one or more tiles. Pages are usually the primary navigation element (and are often rendered as and referred to as tabs).

Portability
The concept of tiles being self-contained and portal-agnostic, meaning that any tile should be able to be dropped into any portal that complies with the proper tile specification.

Page View

The primary view, where multiple tiles are stacked into a page.

Pain
As expanded below, each tile may have different size, view, context, complexity, functionality, interface, and content. This is both the value and curse of the portal. One tile might contain an integration to a large complex application like an SIS, while another tile is simply a list of URLs. Another tile might be web content pulled from outside the portal. Even between two tiles of similar type there is variance in approach. In the case of application integration - email for instance - one email tile implementation might make the default display only a summary of the number of new messages, whereas another might try to display the full inbox. An inbox is a large, complex interface. Imagine several large, complex interfaces stuffed into tight boxes, sprinkled with tiles of miniscule content swimming adrift in a sea of white space, shouldering alien content teleported in from the ends of the Web, all tiled together rather hodge-podge and pell-mell. Thus is the page view of most portals.

The portal really throws down the gauntlet when it comes to the user experience, a content Frankenstein that has yet to be tamed.

Focused View

A secondary view, where the page real estate is given solely to one tile.

Pain
In a quest for portability of tiles, the portal and the tile contents have no communication. The portal can signal a focused view to the tile, but in many cases the tile does not pass that user action on to the tile. As a result, you get a focused tile view in the portal, but the tile contents remain in the same state they were in from the page view. Thus the end result is often a "maximized" tile view with "minimized" or "cameo" tile content view. In other words, the tile knows it has the full real estate, but the tile contents are dumb to the matter, resulting in a waste of interface. This problem also occurs in reverse. When leaving the focused view and returning to the page view, the tile knows to become "cameo", but the tile contents remain in whatever state it was in when focused.

The tile contents should grow with the focused view and shrink in returning to the page view; keep the tile contents context aware.

Tile Views

Each tile may have several views. The current standard for portlets, for instance, identifies four modes (View, Edit, Help, Custom) and four states (Normal, Maximized, Minimized, Custom)

Pain
Views, modes, and states are open to interpretation and are often optional, resulting in great variance and inconsistency of tile content between tiles in the same page and portal. User interaction (example clicking on "Edit") does not produce a similar result in all tiles.

Tiles placed into a shared environment (portal), should have similar and cohesive views.

Portal Content Hosting

A portal provides a point of integration with disparate applications. Often, the integration point is merely a few links or superficial data from the application. Because there is no further integration, trying to access other application functionality or data requires pushing the user into the application itself.