Part I
Usability Documentation

Overall, Slicker is just that. A slicker way of using your computer. It must be unobtrusive and it must be easy to use. This is why this documentation is needed already at this early stage of the development process. This document is a guide to developers in how to create applets and otherwise for Slicker in a consistent way, so the end user will feel at home fast and not need to think too much about how it works. Remember: A user is just that, he is not a mechanic and does not wish to know what is beneath the bonnet of the proverbial car.

The motto of this document must thus be the design principle known as KISS - Keep It Simple, Stupid.

1 Usability test

Still to be decided on, but should include the following:

- Create a card
- Move a window to another desktop and switch to that desktop
- Start program
- Use Slider's PIM functionalities

2 Usability FAQ for Slicker Developers

2.1 Naming

2.1.1 Development

The names of the different structures in Slicker are spelt and cased as follows:

- Slicker: The entire system. Consists of an arbitrary, userdefined amount of EdgeWidgets, which can be one of the following:
  - Task bar: The task bar
  - Slider: An extended form of system tray, containing tiny programs called mini-applets. Mini-applets can extend Drawers
    - Mini-applet: A tiny program, no more than 64 by 64 pixels
    - Drawer: Window that to the user will seem to slide out from under the Slider, containing further options for the mini-applet
      - For example: Desktop pager. Tiny pager showing all the user's desktops in a grid format, which when the mouse is hovering over the pager will show a drawer containing a larger preview of all the desktops in a list with the desktop names as well as which windows are on each desktop.
  - Stack: A stack of cards. Can be both a standard stack and a "rolodex" (explain further), which is selected in the Cards configuration dialog
    - Card: A tiny application holding one or more Applets
      - Applet: Small program which will do one single task (such as for example show the user's in-box, show a list of application shortcuts, search options and so on). Should never be larger than 200 by 200 pixels unless absolutely nessecary

2.1.2 User

All names (as seen by the user) should be self-explanatory. For example an applet which is the application menu part of the K Menu should be called Application Menu (if there is already one such, either help the developer with that, or if yours is substantially different (for example task based) call it such; Task Based Application Menu).

2.2 User Input and Commands

2.2.1 Commands

- Toggle Show/Hide...
  - All Slicker components
  - All Cards
  - Taskbar
  - Slider
- New card
- Add applet to card
- Configure Slicker

2.2.2 Global keyboard shortcuts defaults

- Toggle Show/Hide...
  - All Slicker components - Win+H
  - All Cards - Win+C
  - Taskbar - Win+T
  - Slider - Win+S
- New card - Win+N
- Add applet to card - Win+A
- Configure Slicker - Win+P

2.2.3 Mouse gestures

Is this something that we need to look into? In that case, which commands need the gestures relate to?

2.3 Applets

2.3.1 How should applets attract attention?

- The applet should at all times use a maximum of one of these methods at any one time.
- The applet should always use the same type of attention attractor for the same type of event.

2.3.1.1 If the card containing the applet is closed...

- Open the card to show the applet and close the card again.
- Flash inverted colours on the card tab (3 times maximum)

Implement globally available function to do this

2.3.1.2 If the card containing the applet is open...

- This can be considered a small change (see bellow: "How should applets notify the user of a small change").
- Flash inverted colours on the applet (3 times maximum).

2.3.1.3 Regardless of card state

- Use XOSD or similar program to write a message on-screen (possibly using box-shape zooming effect to and from the card containing the applet)

2.3.2 How should applets notify the user of a small change?

- The applet can register one or more icons with it's container card upon applet startup.
- This icons can be enabled and disabled individually on demand.
- When one or more icons are enabled, the card will cycle through them with an interval of 0.5 seconds.
- If only one icon is enabled, an empty icon will be inserted into the cycling to assist in notification.

2.3.3 What should the basic layout of applets be?

- All slicker applets should (if at all possible, sometimes it will be impossible (example: Console card)) be separated into squares no larger than 200 pixels on all sides for easy arrangement on either screen edge.
- It is essential that all applets are very simple, to comply with the KISS principle of UNIX systems[1]. If the applet becomes too large, maybe you shold consider cutting down the features in the applet interface and implement further features in a program the applet could interface with.
  - An example of the above consideration could be an e-mail applet. You should not show messages in an applet, you should only show the inbox (with a combo-box enabling the user to select which mailbox to use), messages would then be opened by double-clicking. If KMail is open, the e-mail could be opened in the existing KMail window.
  - Another example would be the K Menu. In stead of having one applet with the entire KMenu (known from Kicker) in it, we could have several:
    * Recently/ most launched applications
    * Application menu (standard or task based)
    * System menus (Bookmarks, Quick browser, Recent documents, Printer subsystem, Configuration...)
    * Run applet (can this way contain a text field as well as other interesting features, expanding the card and others)
    * Session management/ logout applet (Save session, Lock screen and Logout)

2.3.4 How should applets use the Card Tray?

- The card's Tray should only be used very sparcely, as all applets in a card have access to the same tray and it could potentially become very cluttered.
- For example, an Instant Messenger should show one button only for Connection, one for Away status; all other functions should be contained in the Applet, or in the contect menu.

[1] Many discussions have arisen over what makes a UNIX a UNIX. It is really quite simple: Originally, the principle of programs in UNIX systems was that they should do one thing, and one thing alone. MacOS, NeXTStep and others have gone in this direction, while Windows have gone the exact opposite way. Linux is at a crossroads, and basically, we need to make sure that it stays a UNIX.

2.3.5 Applets spawning new cards

- A card can spawn a new card if the function of the applet contained is sufficiently different and small enough for a card to be used.
  - An example would be an Instant Messenger card spawning a new card containing a chat dialog.
  - Another example would be a Download Manager applet spawning a new card containing an expanded download progress dialog.

2.4 What should Slicker do in case of...

2.4.1 First startup

- Show First Start wizard and give the user a number of options to choose between (office, games, education, others?)

2.4.2 Slicker finds no cards set up on startup, and Taskbar and/ or Slider are enabled

- Do nothing. Some users may wish to use only Slicker's Taskbar and/ or Slider.

2.4.3 Slicker finds no cards set up on startup, and Taskbar and Slider are disabled

- See above: "First Startup".

2.4.4 User attempts to remove last card

- Ask for comfirmation ("Are you sure you wish to remove the last card?")

2.5 Adding new applets

2.5.1 From Configuration dialog

- Give a list of all current cards.
- List applets contained in each card.
- Give the option to add an applet to any card, or to add a new card and add an applet to it.

2.5.2 From context menu

2.5.2.1 Card context menu

- Give the option to add an applet to the current card ("Add applet to this card")
- Give the option to add an applet to another card ("Add applet to..." submenu with list of all current cards)
- Give the option to add an applet to a new card ("Add applet to a new card")

2.5.2.2 Taskbar and Slider context menus

- As Card context menu
- Do not give the option to add an applet to the current card (Taskbar and Slider are not cards)

2.5.2.3 New applet dialog

- List all available applets (layout to be decided; tree-based with separate description pane, or simple list), give ok and cancel options only.

3 Documentation

3.1 Links

3.1.1 KDE Usability Project

`http://usability.kde.org`

The clearinghouse for all things related to KDE usability. Includes a list of links, information on the mailing list, and the XML structure for submitting KDE usability reports on-line.

3.1.2 KDE User Interface Guidelines

`http://developer.kde.org/documentation/design/ui/index.html`

A collection of articles related to usability. Lots of great content here. Probably should be your first stop.

3.1.3 Bruce Tognazzini: Ask Tog, Design Section

`http://www.asktog.com/menus/designMenu.html`

Site dedicated to human / machine interaction in general. Particularly relevant to software design are his two articles: "First Principles of Design" and "Maximizing Human Performance" (under the "Basic" header).

3.1.4 Jakob Neilsen: Why You Only Need to Test With 5 Users

`http://www.useit.com/alertbox/20000319.html`

Short summary of a study that studied the usability of usability studies. :) Main findings:

- Many small tests are better than one big one
- Test five users at most to get most bang for buck
- Benefit of testing one user is about 1/3 of total!
- You need to retest after a redesign

3.1.5 Colorado University: Usability testing

`http://www.cu.edu/~irm/stds/usability/`

Detailed but concise description of how to conduct a usability test.