Stable Squeak World Tour

John W. Sarkela
jsarkela@exobox.com
sarkela@home.com

Project Goals

- A Production Quality Smalltalk System for the masses
- Extend the Spirit of Camp Smalltalk

Technology

- Derived from the original Apple Smalltalk-80 license.
- Self hosting VM
  - VM written in Smalltalk
  - Smalltalk to C translator
  - Direct object pointers
  - Incremental Garbage Collector
  - Dynamically loaded named primitives

Technology continued

- Network Support
  – Web Server, Web Browser, Email Client, Chat, Ftp, Telnet, MD5, DES . . .
- Sound Support
  – FM Sound Synthesis, KLATT speech synthesis, MIDI support . . .
- Graphics Support
  – 3D Engine, VRML, Morphic, Wonderland

Why Squeak?

- Great for education
  - It’s free, it runs on all platforms, it has Freecell
- Suitable for embedded devices
  - Runtime may be made small
  - All capabilities written in Smalltalk
- Lots of potential for developers
  - Functionality ready for reuse

What’s my motivation?

- **Tell them, “Ralph sent me.”**

The UIUC summer OO design course used Squeak and XP to build a functional object swiki in four weeks with 6 programmers who also learned Smalltalk at the same time

- **So many things “almost” worked . . .**

So what is the problem?

- Squeak needs a production quality base library
- The core team is more interested in experimentation and exploration
- Squeak may be the first time many new programmers see Smalltalk

Of Software and Social Work

- Most of Smalltalk's problems are not technical in nature
- Lack of success stories is not really the issue
- The Squeak out-of-box experience is enough to prevent anyone from exploring Smalltalk further.

What is the proposed solution?

- Use a Camp Smalltalk style development
- Bring the Camp to developers, whereever they may live
- Work as closely as possible with Squeak Central to incorporate refinements into the base system

Current Plan - Phase 1

- Define a minimal development image
- Refactor this image until
  - All methods may be compiled from source code
  - There are no undeclared references
  - All globals have a known initial state
  - Leverage Camp Smalltalk ANSI tests

Current Plan - Phase 2

- Factor remaining functionality into modules such that
  - There are no method or class redefinitions
  - The module dependency lattice is well defined
  - As many unit tests as possible are generated

Current Plan - Phase 3

- Refactor the base into
  - A headless image with just enough included to be able to bind image segments
  - A set of bindable UI's, including a text based stdin,stdout,stderr UI
    - (Anyone for an emacs browser???)
- Build ImageSegments from defined modules