Extensibility in GNUstep & Étoilé

GNU Hackers 2011

http://www.gnustep.org

http://www.etoileos.com
Objective-C & GNUstep
Objective-C

• Created by Brad Cox and Tom Love in 1986 to package C libraries in Smalltalk-like classes

• Comes with dynamic features such as
  • message forwarding
  • categories to extend existing classes
  • resolve methods lazily etc.
@interface Person : NSObject

- (void) sleep;
@end

@implementation

- (void) sleep {
    NSLog(@"Zzzz!");
}
@end
@interface Person (Talktative)
    - (NSString *) talk;
@end

@implementation Person (Talktative)
    - (NSString *) talk {
        return @"poumpoumpidoum";
    }
@end
Objective-C Runtime

• No virtual machine, but a small runtime library
  • class_getSuperclass()
  • class_setSuperclass()
  • class_replaceMethod()
  • method_getArgumentType() etc.

• Provides type infos for C types such as structs, unions, pointer etc.
Class Transform

- Dynamic implicit subclass creation
- Many Use cases
  - Persistency (Fast Portable Orthogonally Persistent Java)
  - Change Notifications (Key Value Observing)
  - Prototypes (Google V8, libobjc2)
  - Faulting, State Machine, AOP etc.
Composition of Class Transforms

• Multiple transforms create several implicit subclasses…
• Methods can be overridden several times
  • Composition order matters
• How to be sure the resulting behavior is correct?
  • No well-known model to support composition
Safe Composition of Class Transforms

• V8, libobjc2 and Foundation approach
  • restricts the supported transforms to the core language or library level
  • hides the implicit subclass

    id obj = [A new]
    objc_setAssociatedReference(obj, key, value, retainPolicy)
    [[obj class] isEqual: object_getClass(obj)] // A in both cases
Class Cluster

• Variation on the Abstract class idea
• A single public Class
• Multiples concrete implementation classes
• The public class initializer and copy methods choose the class of the returned object
• For example… NSSet, NSArray, NSNumber, NSString etc.
NSString Class Cluster

• GSPlaceholderString
• GSString
  • GSCString
    • GSCBufferString
    • GSCInlineString
  • GSCSubString
• GSUnicodeString (same subclasses than GSCString)
Class Cluster

• In theory very nice :-)

• In practice…

• Poorly documented API contracts by Apple

• No way to register new implementations and control how the concrete classes are choosen
Class Registration

• Extra classes loaded on-demand to provide new abilities e.g.
  • reading/writing new document format

• Registration API involves method such as
  • +registerClass:
  • +unregisterClass:
  • +registeredClasses
NSImage Example

```objc
[NSImageRep registerImageRepClass: [MySVGImageRep class]];

NSImage *img =
    [[NSImage alloc] initWithContentsOfFile: @"~/tiger.svg"];

// [img representations] contains a MySVGImageRep instance
```
Drawing Backend Example

- GNUstep imaging is based on the DisplayPostScript model
- NSGraphicsContext is the public API and an abstract class
- Concrete subclasses adapts the DPS model to various drawing libs e.g. Cairo, Xlib, GDI
- CairoContext, XGContext, WIN32Context
• NSGraphicsContext is part of the AppKit framework

• While each concrete subclass is located in a bundle that is chosen at launch time

• System/Library/Bundles can contain `libgnustep-xlib.bundle` or `libgnustep-cairo.bundle`

• `defaults write MyApp GSBackend libgnustep-cairo`
Étoilé
Étoilé

A desktop environment built around

• Pervasive Data Sharing & Versioning
• Composite Document
• Collaboration
• Light & Focused Applications (1000 loc max per app)
Étoilé Today

Well, presently more or less a development platform centered around

- LanguageKit
- CoreObject
- EtoileUI
Small in the long run

• An entire desktop environment in 150 000 loc
  atop GNUstep and some other dependencies such as LLVM, FFmpeg, TagLib etc.

• Most frameworks are between 2 000 and 6 000 loc

• Only two frameworks are above 10 000 loc
  • LanguageKit
  • EtoileUI
LanguageKit
LanguageKit

• A framework to build dynamic languages based on the ObjC object model
• Small and modular
  • ~ 15 000 loc
• Fast…
Already Fast...

- LLVM built-in passes
- Small objects hidden in pointers (e.g. efficient integer computation)
- The new GNUstep runtime comes with
  - various extra passes
  - type feedback to generate profiling infos related to call sites...
ObjC Runtime Passes

- Cached lookup
  - fragile instance variable and class access
  - classes messages
  - messages in a loop
- Method inlining
  - class methods
  - speculative
Benchmarks

- Almost the same speed as C integer arithmetic e.g. Fibonacci benchmark ran the same speed as GCC 4.2.1
- With all optimizations, can be faster than C in some micro-benchmarks
- Probably 5 to 10 times the speed than an open source Smalltalk such as Squeak
- Floating point still slow but will become fast soon :-}
Primitive Support

- Automatically box and unbox primitive types such as int, float etc.
- Integer operations/methods as C functions, compiled to bitcode and inlined by LLVM
- $4 + 4$ in Smalltalk is as fast as C
- As a bonus, C direct library access without FFI, just do C sqrt: 42 for sqrt(42)
Modular

Composed of several components in separate libraries

• An AST geared towards dynamic languages bundled with an AST interpreter

• A code generator-based on LLVM (JIT or static compilation)

• Two language front-ends (Smalltalk, EScript currently)
Mixing Languages

- Methods written in EScript, Smalltalk and ObjC
- can belong to the same object
- can call each other
- You can clone an object in EScript then pass it around to some Smalltalk or ObjC code
- ObjC and Smalltalk blocks are interchangeable
Surprisingly Small

• Found on Digg (in 2007)…

• Konqueror itself is really a surprisingly small app: approx 40k lines of code. Not tiny, by any stretch of the imagination, but way, way smaller than people seem to think it is.

• 40x what is allowed in Étoilé :-/

From: http://digg.com/linux_unix/Nautilus_vs_Dolphin_vs_Konqueror
Code Compression

• Étoilé Generic Object Manager
  • 700 loc

• Étoilé Model Builder
  • 1 000 loc
Model Builder

Editing a package & browsing a repository
Post-WIMP?

• From the whole screen to a single row in a list view…

• It’s just an uniform tree structure

• No special window, list or row node
An existing application should be easy to retarget

- personal computer
- mobile phone
- tablet
- web
Why a “new” UI toolkit?

• Everything can be changed at runtime
• Simple, compact and highly polymorphic API
• Write less code and develop faster
• Feeling of manipulating real objects
What does it solve?

- Generic protocol for Structured Document
- Building blocks for Graphics Editor
- Custom widget development
- As little code as possible
- Plasticity
Separation of Concerns

• No monolithic view/widget, but rather…

• UI aspects
  • Styles, Decorators, Layouts
  • Tools, Action Handlers
  • Widgets
  • Model Objects, Controllers
Turtles all the way down

Many things are just layout items

• selection rectangle
• handles
• shapes
• windows
• layers