

Designing Interactive Systems II

Computer Science Graduate Programme SS 2009

Prof. Dr. Jan Borchers RWTH Aachen University

http://hci.rwth-aachen.de



Model-View-Controller

- Central concept behind Smalltalk-80 and its first multiwindow GUI interface
- View: Manages graphical/textual output
- Controller: Interprets user input (mouse,kbd) and tells model and/or view to change
- Model: manages data and behavior of application domain, responds to (View) requests about its state and (Controller) requests to change its state



Model-View-Controller

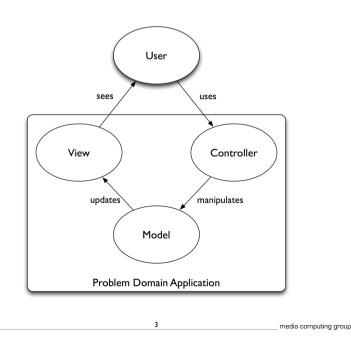
media computing group

- Smalltalk-80 has abstract objects Model, View, Controller
- View & Controller need little added code; offer standardized display and input techniques
- *Models* cannot be standardized that way; any object can be a model

4

• Example: String as model for a simple editor

Jan Borchers



Jan Borchers

Passive Models

Active Models

- Simplest case
- Controller responsible for notifying the view of any changes, because it interprets user input
- Model not responsible for triggering anything, unaware of the MVC triad

- Most models cannot be so passive
- Need to inform all(!) dependent views when the model's state changes (by sending *update* msg.)
- When a View is given its model, it registers itself as a dependent of that model



View and Controller

- Each view is associated with a unique controller and vice versa (through instance variables *controller* and *view*)
- They also both have a *model* instance variable
- The view is responsible for establishing these links

7

Subviews

media computing group

• Views are nested

Jan Borchers

Jan Borchers

- topView has StandardSystemController for moving windows (→ Window Manager task)
- subViews have associated controllers for their particular purpose (buttons,...)
- Bidirectional pointers (*subViews, superView*) establish tree structure

8



Apps
UITK
WM
BWS
GEL
HW

- Common ancestor of all window systems
- Alan Kay (PARC, early 70's): Dynabook
- Influenced by Simula, Sketchpad (DIS I), Logo
- Initially on 64K Alto
- Used in 70s to teach OO to school children...
- Introduced windows, scrolling, pop-up menus, virtual desktop, MVC



Smalltalk: Architecture

• Squeak: Recent open-source implementation (since 1996) by Alan Kay and others

П

- Smalltalk is a purely object-oriented and simple language
- Messages are sent to objects
 - Result := Object message: Parameters

Smalltalk: Architecture

- Machine-dependent Virtual Machine (byte-code interpreter)
- Machine-independent Virtual Image (Smalltalk classes)
- Complete universe, simplest WS archit.
 - OS, language, WS, tools: single address space, single process structure, communicate with procedure calls
 - Initially, OS & WS merged, on bare machine
 - Later, WS on top of OS, but still "universe"



Morphic

media computing group

- User interface construction environment for Smalltalk
- Originally devised in the mid-90s [Maloney'95]
- Directness

Apps

UITK

WM

BWS

GEL

HW

- Change look&feel of widgets by pointing at them
- No separate "GUI editor view"
- Liveness

Jan Borchers

- UI is always active and working
- No separate "edit" and "run" modes
- Reduces UI development time, lowers cognitive load, real-world analogy

12

• Supports multiple users working simultaneously(!)



Jan Borchers

Morphic: Structural Reification

- Widgets are called morphs
- Any morph can be a container (hold submorphs)
- Submorphs managed through container, handle events first

Jan Borchers _____ I3 ____ media computing group

Morphic: Layout Reification

- Layout morphs automatically and continuously lay out their children and make layout policy tangible—Layout Reification
 - Row & Column Layout morphs
- Find compromise for submorph space requests, pass single space request on to parent
- Minimum size and resizing policy as attributes, H&V independent

15

• rigid, space fill, shrinkwrap

Morphic: Structural Reification

- Advantage: entire dynamic widget tree consists of real morphs—Structural Reification—, enabling directness since every part of the widget tree can be manipulated directly
 - E.g., turn labeled button into button with movie on it
 - Extreme case: Editor with every character as morph
- Applications are just big composite morphs built by direct manipulation, including connections between control and target morphs(!)

14

In-Class Exercise: Implementing Layout

media computing group

- Algorithm to determine the layout of a morph that includes a tree of submorphs?
- Answer:
- Ist pass: Compute minimum size of all submorphs bottom-up
- 2nd pass: Distribute available space between submorphs top-down
- **Optimizations?**
- Deferred layout: Don't layout until visible
- Pruning: Maintain layoutOK flag for subtrees, do not compute subtree layout if flag ok and required space available
- Site Selection: Try to limit recomputation to subtree up to next likely stable (e.g. rigid) morph

16

Jan Borchers

Review

media computing gro

- What is the difference between Smalltalk, Squeak, and Morphic?
- How did the original Smalltalk implement the window system layer architecture?
- What are the most particular qualities of Morphic as a UI toolkit?
- What are morphs, and what is special about them?
- How does Morphic implement widget layout?



Morphic: Ubiquitous Animation

- Multiple animations active concurrently
- Animations can be composited concurrently or sequentially, abort by user possible (e.g. delete file)
- Increases Liveness, allows objects to observe others

19

Morphic: Ubiquitous Animation

- Morphs can have autonomous behavior, usually appearing as animation (clock,...) (intrinsic step method, triggered by system each frame or less often, from activity list)
- Also, animation behaviors (move, scale, change color) can be assigned to any morph (as external activity, frame- or time-based, several pacing options, triggered from activity list n times)

18

• These two are orthogonal



Managing redraws

- Damage List
 - Add bounding box of each changed morph to list (at both locations if moving)
 - Each frame, redraw all morphs intersecting each bounding box in damage list, back-to-front off screen, then copy to screen (double buffering)
- Improvements
 - Merge overlapping bounding boxes when reported
 - Prune submorph drawing to damage rectangle (works well with Row&Column morphs)
- Don't draw occluded morphs (requires each morph to fill its bounding box)

Jan Borchers

media computing group



Morphic: Live Editing

- No edit/run modes
 - +: No mode changes, no cognitive load, works with n users
- How distinguish operating from editing gestures?
 - Context-sensitive meta menu on right click
 - Includes access to code for morph, decomposable
- Special commands to access submorphs (spatial demultiplexing), specify additional operands



× ■ Getting Started... equè XI 0 queak is a work in progress based on Smalltalk-80, with which it Gt7 s still reasonably compatible. Every Squeak release includes all urce onde for the Squeak system, as well as all source onde for s Virtual Machine (VM, or interpreter, also written in Smalltal dismiss this men Blue items in this window are active text. If an item contains a package browser method finder \bigcirc it will require internet access and may take a while to load workspace file list The Worlds of Squea Not only is all source code included, and changeable at will, it is also completely open and free. The Squeak system image runs bit-identically across all platforms, and VMs are available for jus transcript inner world about every computer and operating system available. The history of the Squeak project can be read at simple change so dual change sort email reader web browser IRC chat The Squeak license and most other relevant information can be ound on the Squeak Home Page mvc project morphic projec http://www.Squeak.org Morphie enders of add:afterIndex: [4] This release of Squeak uses th queak also includes an MVC a (YM OrderedCollection hierarchy (see the world menu add: add:after: Object Collection opving adding SemienceableCol add:before 44411 Process Browser GraphicSymbo dd 411Firet dAllLast rivate instance class esting dFirst di 3/15/1999 14:01 • adding • 1 impleme change (1 2 3 4) includesAllOf: #(2 3) → true (1 2 3 4) includesAnyOf: #(2 3) → tr (1 2 3 4) windowReqNewLabel: #(2 3) (1 2 3 4) ^{*=} #(2 3) → true (1 2 3 4) ^{*=} #(2 3) → true (senders)(implementors)(versions)(inheritance)(hierarchy)(inst vars)(class add: newObject afterIndex: index "Add the argument newObject as an element of the receiver. Put it in the sequence just after index. Answer newObject ype a fragment of a selector in the top pane. Accept i self insert: newObject before: firstIndex + index r, use an example to find a method in the system. Type ↑ newObject eceiver, args, and answer in the top pane with periods between ne items. 3. 4. 7 Supplie

Smalltalk: Evaluation

- Availability: high (Squeak,...)
- Productivity: medium (depending on tools, libs used)
- Parallelism: originally none, now external
 - But linguistic crash protection
- Performance: medium (high OO overhead since everything is an object)
- Graphic model: originally RasterOp
- Style: flexible (see Morphic, for example)
- Extensibility: highest (full source available to user, code browser)

23

Smalltalk: Evaluation

- Adaptability: low (no explicit structured user resource concept; although storing entire image possible)
- Resource sharing: high
- Distribution: none originally, yes with Squeak
- API structure: pure OO, Smalltalk language only
- API comfort: initially low, higher with Squeak&Morphic

24

- Independency: High (due to MVC paradigm)
- Communication: flexible (objects pass messages)

A.

Jan Borchers

media computing group

