The Edition Manager

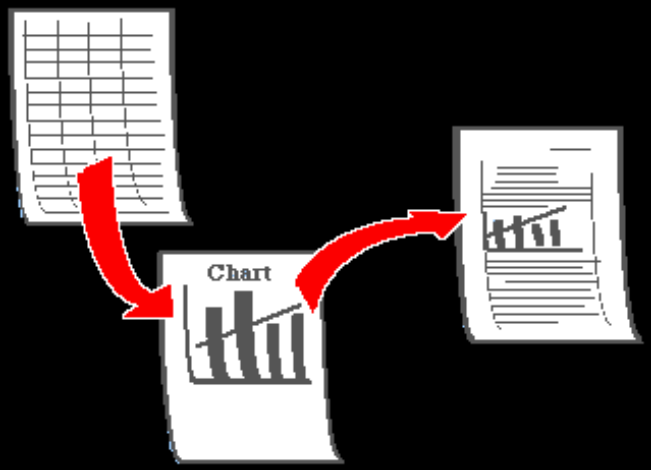
Overview

Tom "My wife just had a kid" Ryan

Engineering Manager

Overview

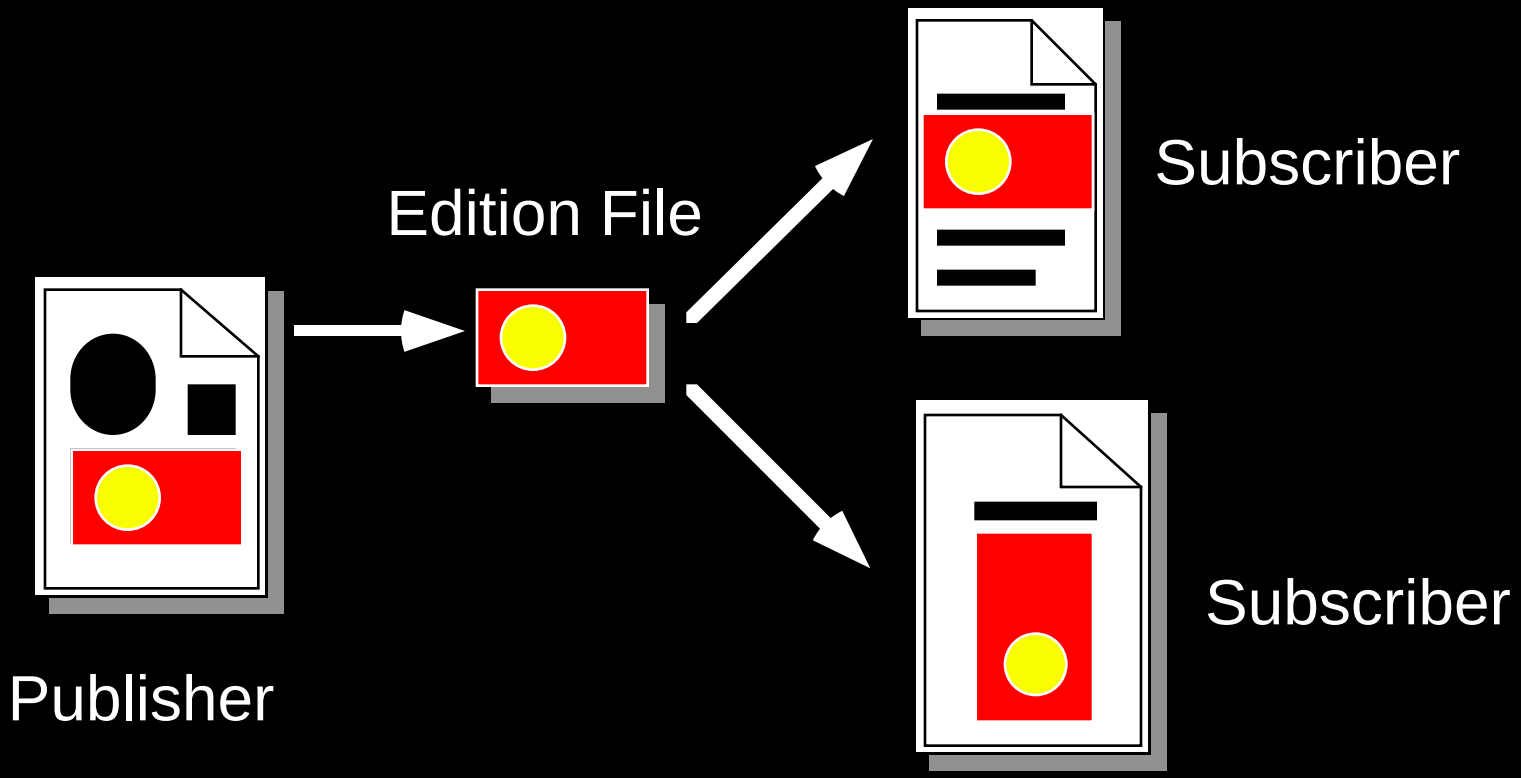
- Analogous to Copy and Paste
- Complementary to the Clipboard
- Based on publishing/subscribing metaphor



Overview

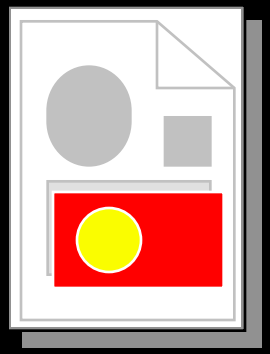
- Allows sharing of data between documents
- Keeps documents up-to-date
- User interface guidelines with Toolbox support

The Edition model



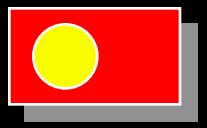
The Publisher

- Source of data to share
- Writes to Edition File
- Updates Edition File when data changes
- One publisher, one Edition File



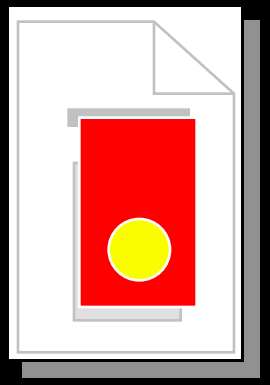
The Edition File

- A file with a name and a Finder icon
- Holds the data to share
- Always contains the latest official edition
- Meeting point for subscribers and the publisher



The Subscriber

- Recipient of shared data
- Reads from the Edition File
- Subscribers are generally not editable
- Ways to navigate from subscriber to its publisher



Benefits of the Edition model

- Shared data across the network
- More user control
- Many subscribers to one publisher
- Updates subscriber next time it's opened

Publish/Subscribe vs. Copy/Paste

- Copy/Paste
  - For data that does not change
  - To rearrange a document
- Publish/Subscribe
  - Repetitive copy and paste
  - Sharing data over a network
  - Small cooperative applications

Apps should not use Editions for:

- "Real-time" updating
- "Navigational links" between documents

Human Interface Guidelines

Scott Jenson

Human Interface Group

The Ground Rules

- New interface behavior
- Designed for ALL users
- Applies across all applications
- Very important to get the basics right

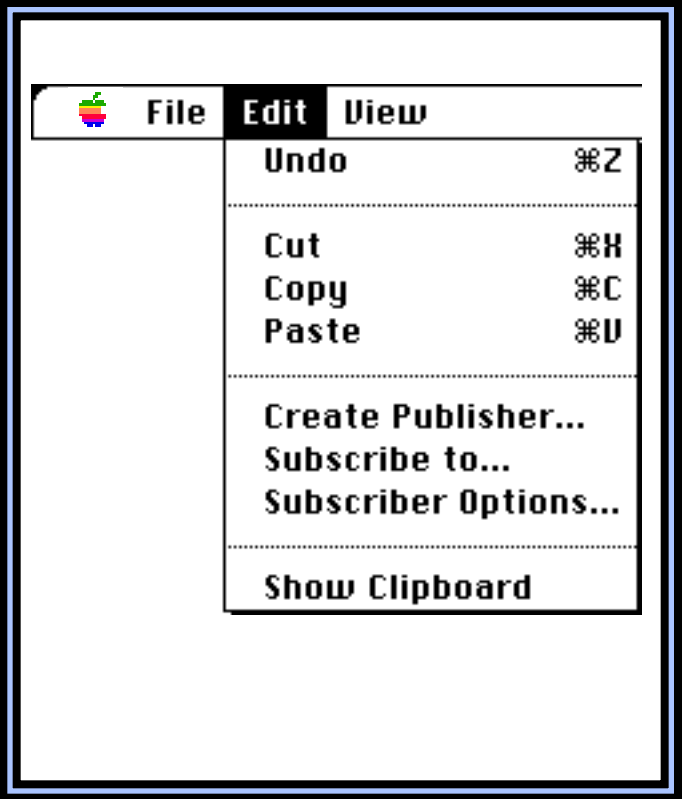
Publish and Subscribe should “just work”

- It’s the details of the user interface that make it work
- There are quite a few, but they aren’t difficult
- The user never sees these details

Terminology

- Three new words for the user
  - Publisher
  - Subscriber
  - Edition
- Don’t ever use the words “Section” or “Edition Manager"

The Edit Menu



- Three new menu items
- Subscriber/Publisher Options
- No Command Keys

Showing Borders

- Borders are “dynamic”
- Appear whenever the contents are selected
- Disappear whenever the user clicks outside
- “Show Borders” item is optional

Border Design

- Publisher: 3 pixel, 50% gray
- Subscriber: 3 pixel, 75% gray
- Borders are outside the contents

Word Processors: Basic

- Universal receiver, primarily subscribes
- Borders move with their text
- Publisher borders grow and shrink as you edit

Word Processors: Positioning the cursor

- The border exists between characters
- Publishers have “gravity”
  - Clicks move towards the Publisher

- End case: other publishers, page breaks, etc.

SpreadSheets

- Primarily Publishers
- Subscribers don’t replace linking
- Subscribers bring in “outside” stuff
- Borders are similar to word processors

Object Drawing

- Not as much structure as previous types
- Subscriber is simple
- Publisher isn’t tied to document contents
- It “floats” over the document
- Borders are no longer dynamic
- Show/Hide borders is necessary

Bitmap

- Primarily as Publishers
- Handling Subscribers forces Bitmap applications to become more Object-like.
- Show/Hide borders is necessary

Other Types

- You should be able to get the basic behaviors out of these examples
- This functionality is a base to build on
- When in doubt: MACINTERFACE

Publish/Subscribe Dialog Boxes

- Last thing published is first subscribed

- Preview gives content hint
- But also give type information

Options Dialog Boxes

- Can add specific controls to the bottom of the Options dialog
- The OK and Cancel button move automatically
- Ensures future compatibility

Subscribers:

- Generally, Subscribers are read only
- Can allow searching and selecting

- Can allow editing through Options dialog
- Warn users before Subscriber updates

Backward Compatibility in 6.0.x

- Disable Publish/Subscribe/Options items
- Display Publishers and Subscribers
- Allow editting

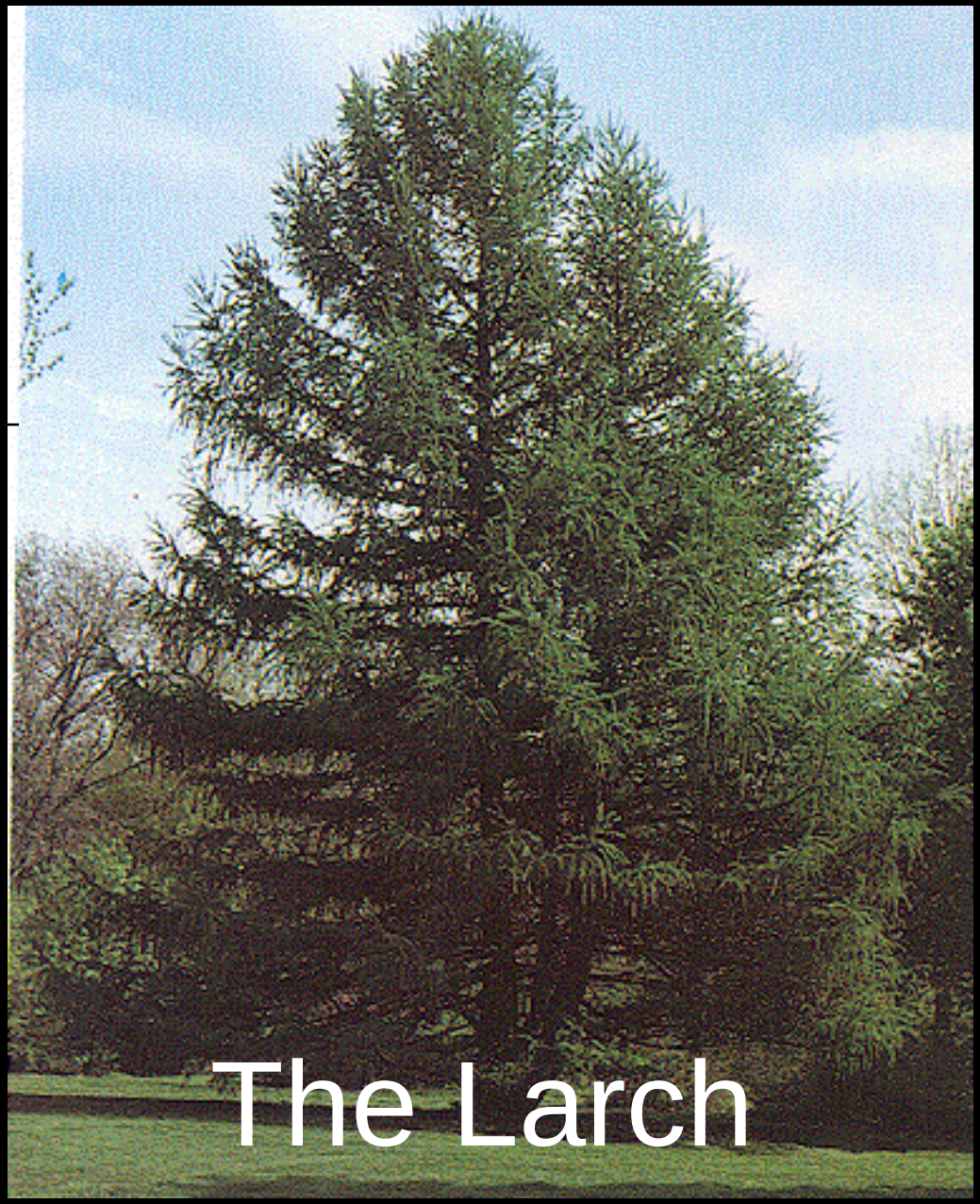
Programming for the Edition Manager

Nick Kledzik

Edition Manager Engineer

How to recognize different trees from quite a long way away

1


The Larch

Overview

- Programmer's model
- Data formats
- When does updating occur?
- Code walk through

Persistent Selection

- Remembering a selectable area of a document, even after editing
- Publishers and Subscribers are special cases of persistent selections

Examples of Persistent Selection

- Named ranges in spreadsheets
- Marks in MPW
- Style variables in word processors

Sections

- Edition Manager term for persistant selection
- Requires a SectionRecord and AliasRecord
- Can register and unregister sections
- Alias associates section to an EditionContainer

EditionContainer

- Abstract term for Subscriber subscribes to and what a Publisher writes to
- The standard EditionContainer is an Edition File
- In the future, other types of EditionContainers may be supported

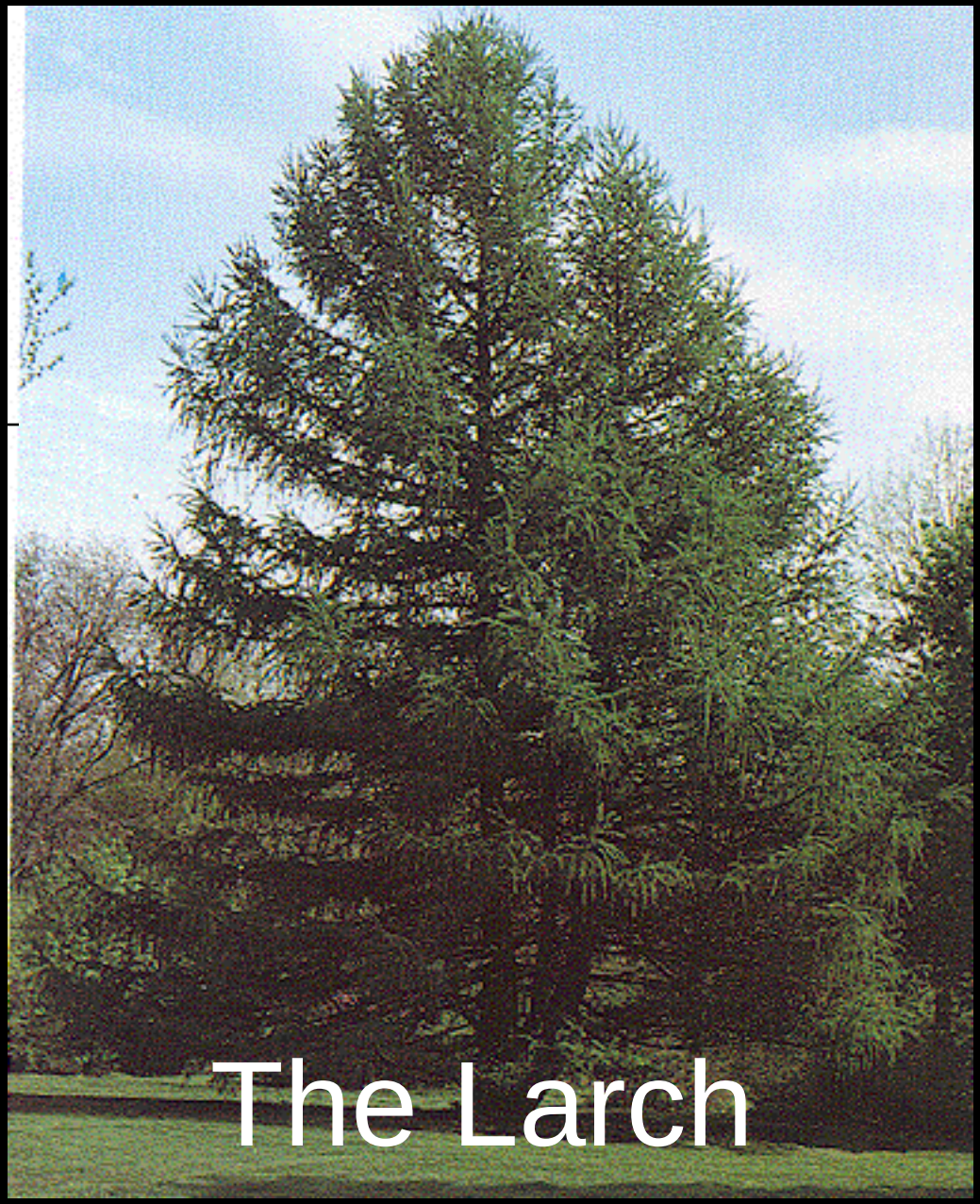
Publisher

- A Publisher is an editable section
- Writes its contents to its EditionContainer

Subscriber

- A subscriber is a section
- Caches a copy of its EditionContainer contents

1


The Larch

Data Formats

- Data is published in "clipboard format"
- Example: TEXT, PICT, private
- Same rules:
  - be able to read TEXT and PICT
  - write at least TEXT or PICT
- Some new special formats: prvw, fmts

I/O model

- Edition can contain multiple formats
- Each format has its own mark
- Can set mark to any position
- Can read/write any amount starting at mark

When does updating occur?

- By default
  - Publisher writes when its document is saved
  - Subscribers in open documents notified immediately
  - Subscribers in closed documents notified when next opened
- User can turn off updates

Section events

- Arrive as AppleEvents
- Four kinds:
  - Read
  - Write
  - Cancel
  - Scroll To

Creating a Publisher

- User selects: Create Publisher...
- Application calls:
  - GetLastEditionContainerUsed
  - NewPublisherDialog
  - CreateEditionContainerFile
  - NewSection
  - then writes first edition

Creating a Subscriber

- User selects: Subscribe to…
- Application calls:
  - GetLastEditionContainerUsed
  - NewSubscriberDialog
  - NewSection

How a Publisher writes

- User saves document:
- Application calls:
  - OpenNewEdition
  - SetEditionFormatMark
  - EditionWrite
  - CloseEdition

Publisher writing TEXT

```
OpenNewEdition(theSectionH,
      gSignature, thisDocPtr, theRef);

SetEditionFormatMark(theRef,
        'TEXT', 0);

WriteEdition(theRef, 'TEXT',
        texPtr, textSize);

CloseEdition(theRef, {success}TRUE);
```

How a Subscriber reads

- Application receives a section read event
- Application calls:
  - OpenEdition
  - EditionHasFormat
  - SetEditionFormatMark
  - EditionRead
  - CloseEdition

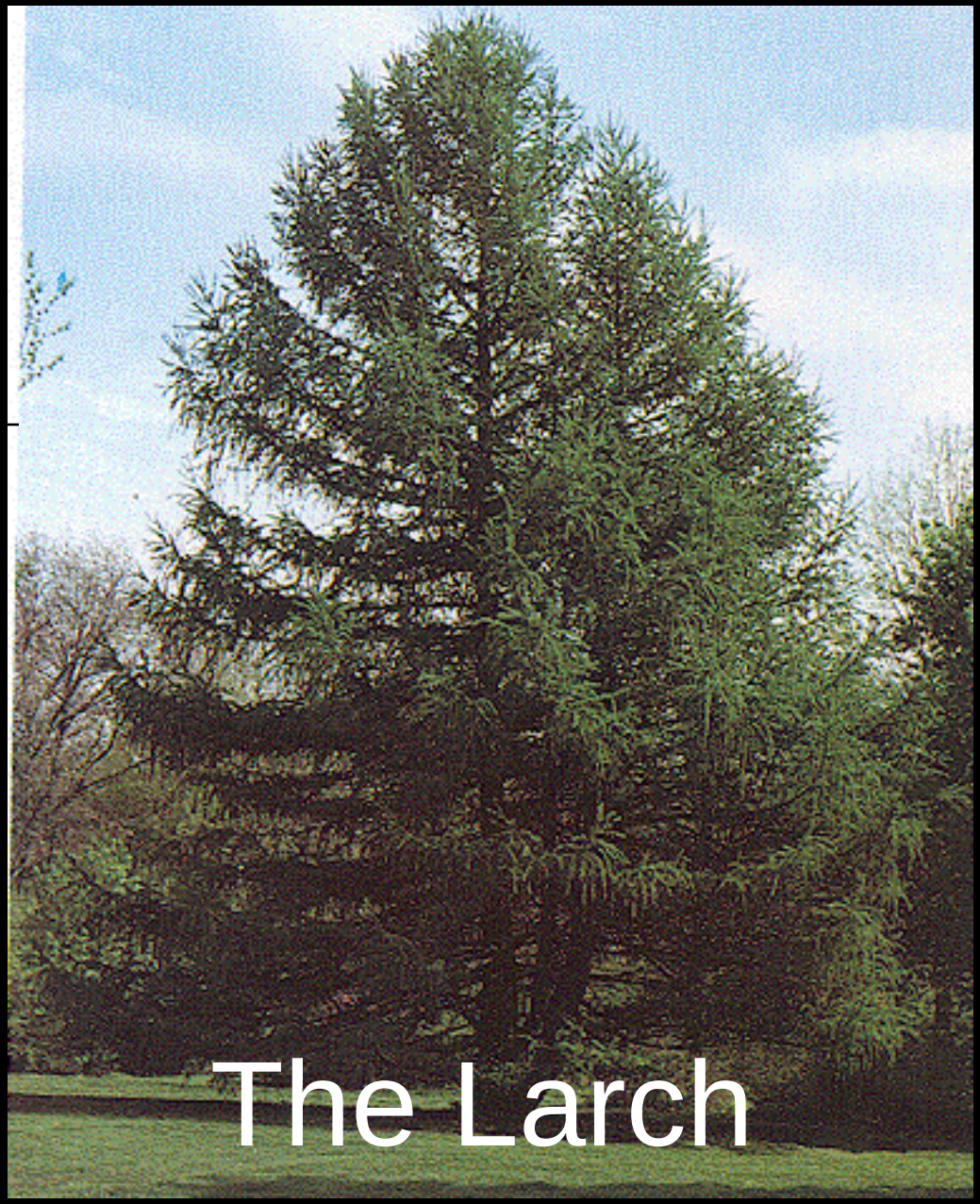
Subscriber reading TEXT

```
OpenEdition(theSectionH,theRef);
IF EditionHasFormat(theRef, 'TEXT',
                    textLen) = noErr THEN
BEGIN
   SetEditionFormatMark(theRef,
                                        'TE
   ReadEdition(theRef,
                        'TEXT', texPtr, textL
   CloseEdition(theRef,{success}TRUE);
END ELSE
   CloseEdition(theRef,{success}FALSE);
```

Saving a document with sections

- Writing modified publishers
- For each section:
  - Save SectionRecord as 'sect' resource
  - Save AliasRecord as 'alis' resource
- Save rest of document

3


The Larch

Bottlenecks

- All Edition I/O goes through bottlenecks
- Applications can intercept bottlenecks
- Interface is like a DEF proc
- Can allow your application to subscribe to:
  - whole paint documents
  - whole draw documents

Summary

- Every 7.0 application should Publish and Subscribe
- Don't go overboard with "feature creep"
- Consistant Macintosh applications succeed

Q & A