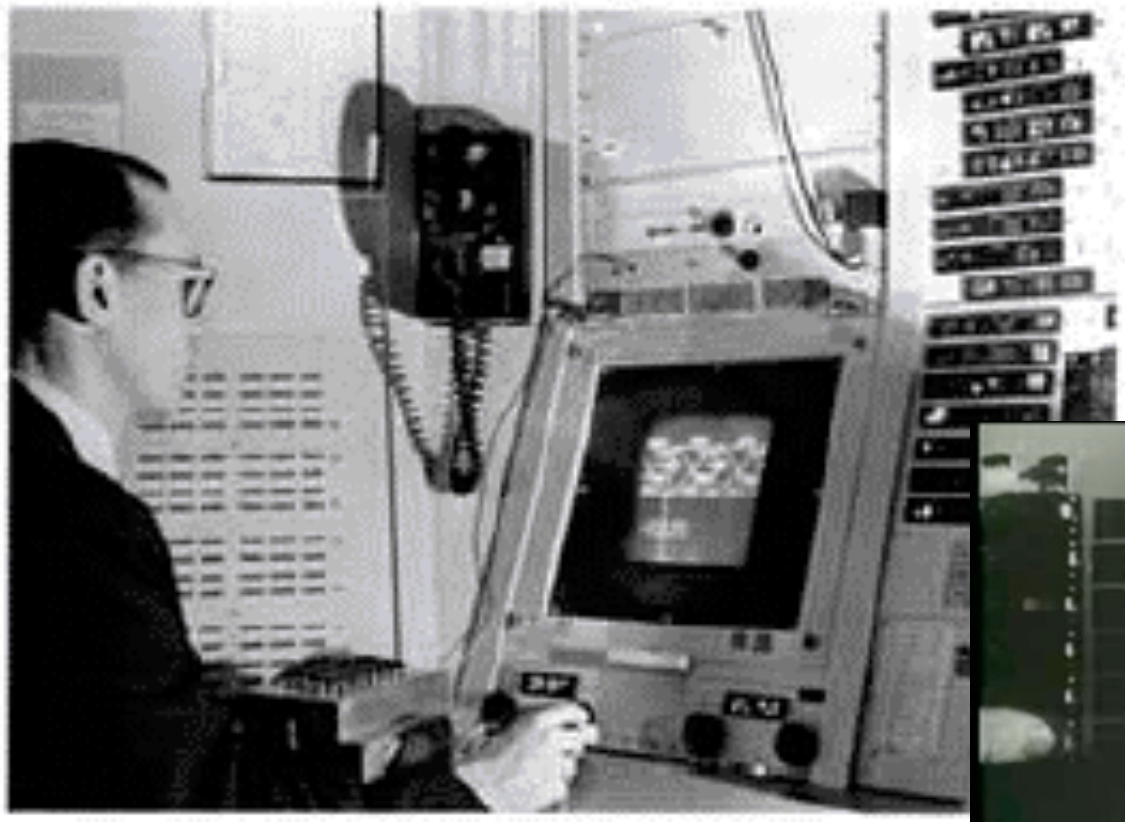
2 Challenges in Multimedia Programming

2.1 Historical Background

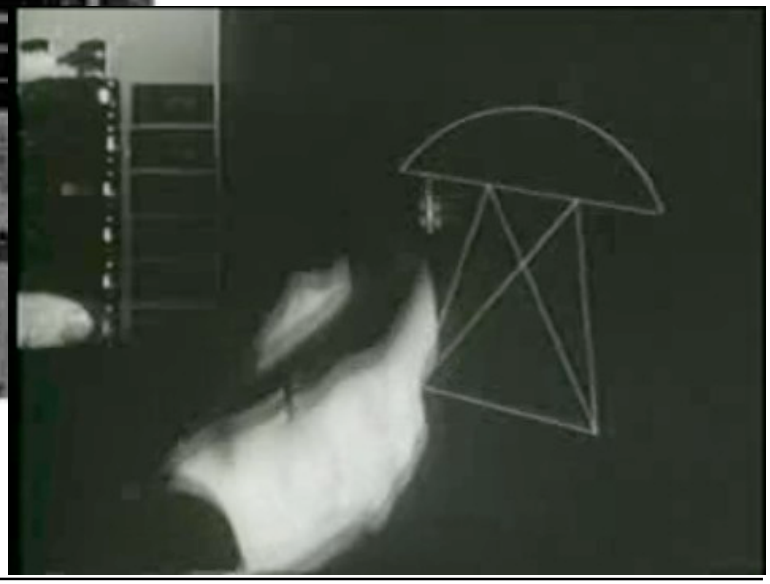
2.2 Application Areas, Terminology

2.3 Challenges & Features for Development Platforms

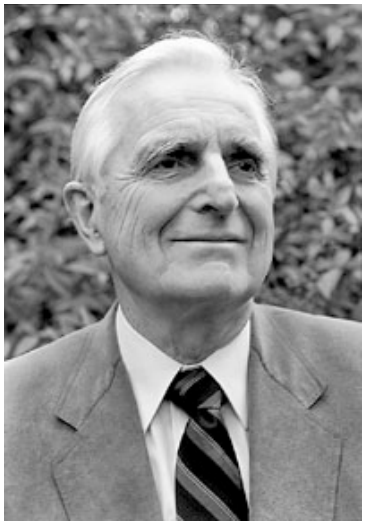
Ivan Sutherland's Sketchpad, 1963



- First object-oriented drawing program
- Master and instance drawings
- Rubber bands
- Simple animations



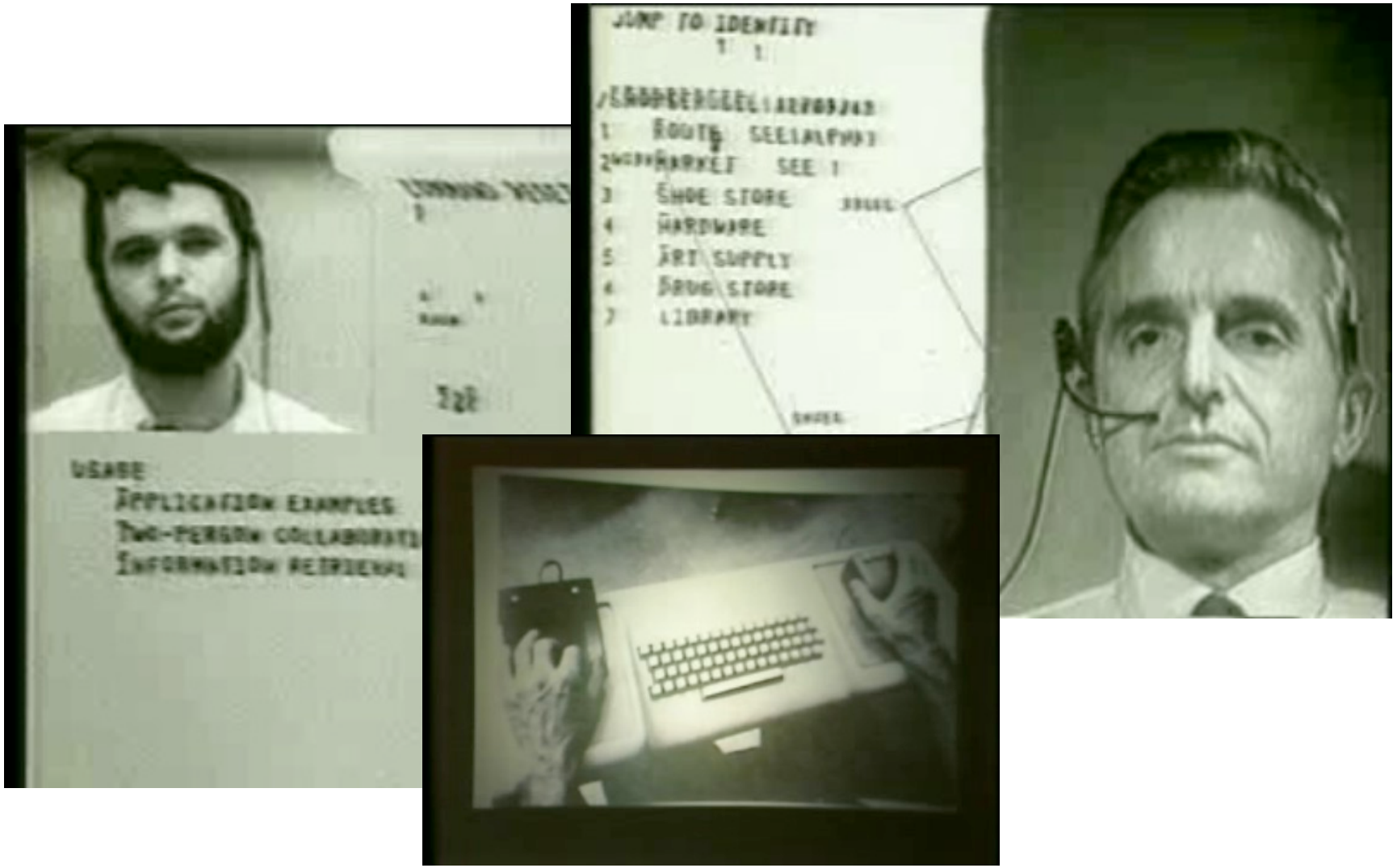
Douglas C. Engelbart 1962



- Born 1925, Ph.D. Berkeley 1955
- Influenced by Vennevar Bush's article "As We May Think" (1945)
- 1962: Research Project at SRI (Stanford Research Institute): "Augmenting Human Intellect: A Conceptual Framework"
  - Research support triggered by the "Sputnik shock" (1957)
- Basic ideas:
  - Computer supported learning
  - Computer supported collaboration
  - Seamless integration of computer interaction into workflows
- Development of the "NLS" (oNLine System)
  - Demonstrated 1968 in Brooks Hall, San Francisco
- 1970: Patent application for "X-Y pointing device" (mouse)

http://www.bootstrap.org/augdocs/friedewald030402/augmentinghumanintellect/ahi62index.html

NLS Demo 1968



Alan C. Kay

- U. Utah PhD student in 1966
  - Read Sketchpad, Ported Simula
  - "Flex: A Flexible Extendible Language"
- Saw “objects” as the future of computer science
- Dissertation (1969): "The Reactive Engine" propagates an object-oriented *personal* computer
  - A *personal* computer was a radical idea then!
  - How radical?

"There is no reason anyone would want a computer in their home."
(Ken Olsen, Digital Equipment Corp, **1977**)

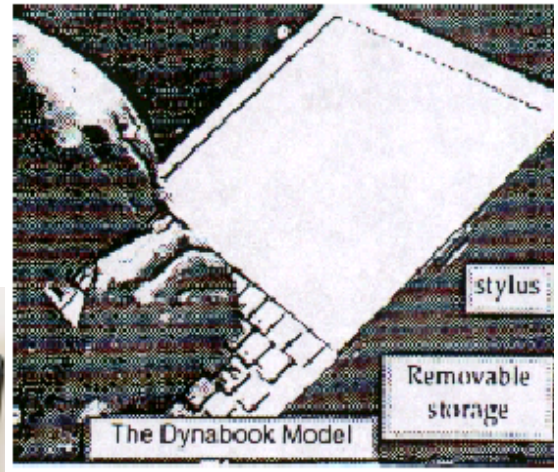
Further stations of Alan Kay’s life:
– Stanford Artificial Intelligence Laboratory
– **Xerox PARC**
– Atari
– Apple
– Disney Interactive
– Viewpoints Research Institute
– Hewlett-Packard

from M. Guzdial

The Dynabook Vision

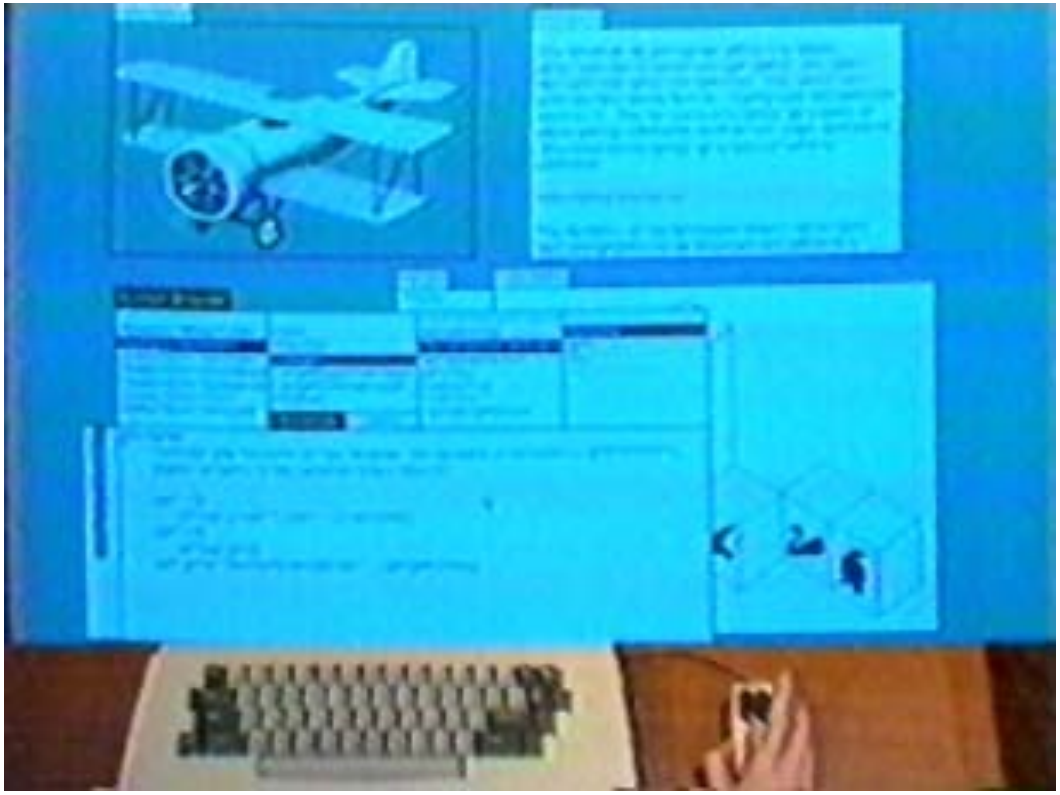
- Small, handheld, wireless(!) device – a new *medium*
- Can be used creatively by everybody, in particular children, for learning
- Xerox PARC Learning Research Group, early 70s

Tablet PC, 2004



iPad, 2010

Xerox PARC Learning Research Group: Smalltalk-72

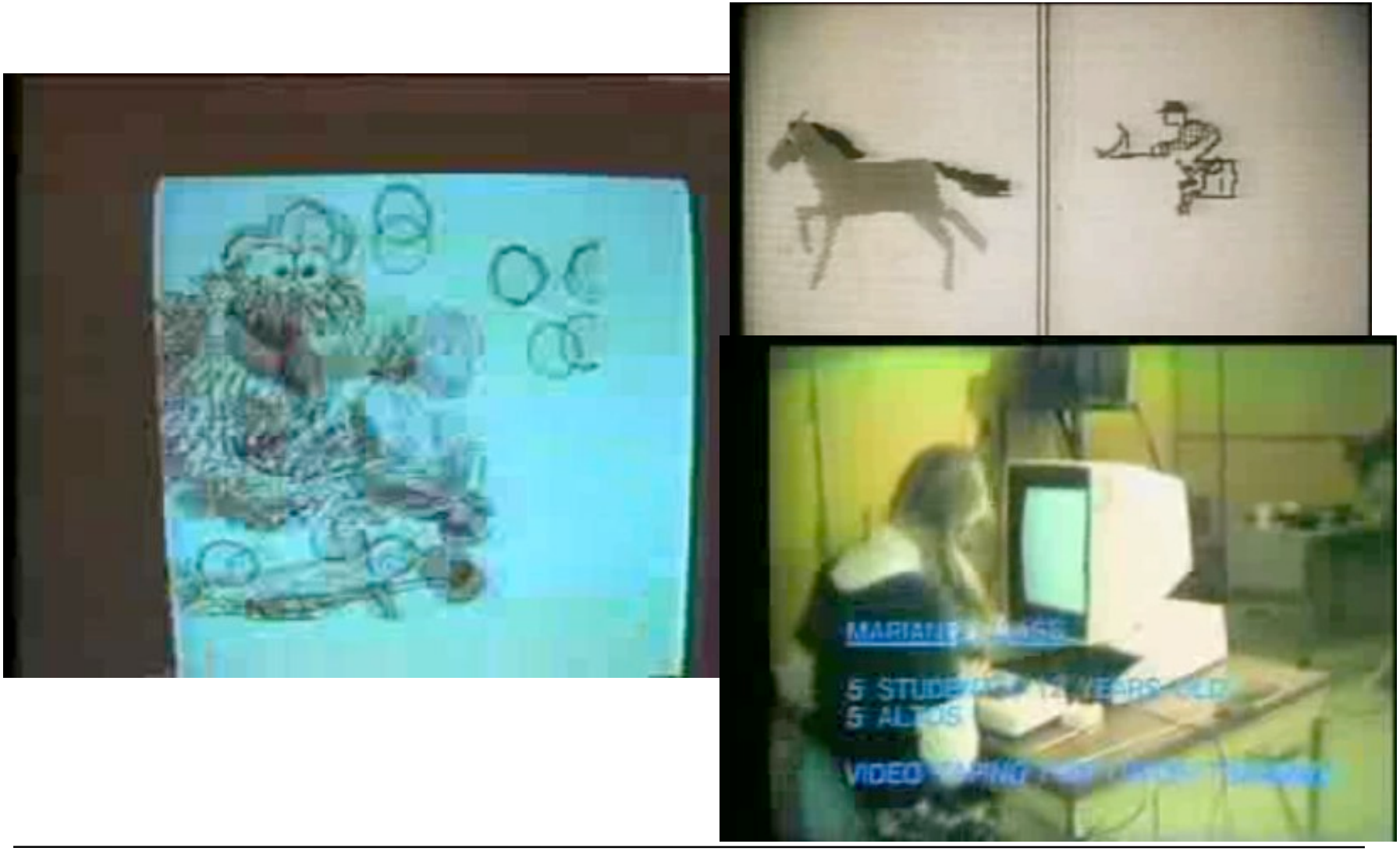


- Object-oriented programming system
  - Mouse
  - Windows
  - Icons
  - Pop-up menus
- Uses simple object-oriented language “Smalltalk”
- Idea of user interface: Make computers easy to use for everybody
- Idea of language: make programming both more simple and more powerful (e.g. include *multimedia: sound*)

The Alto

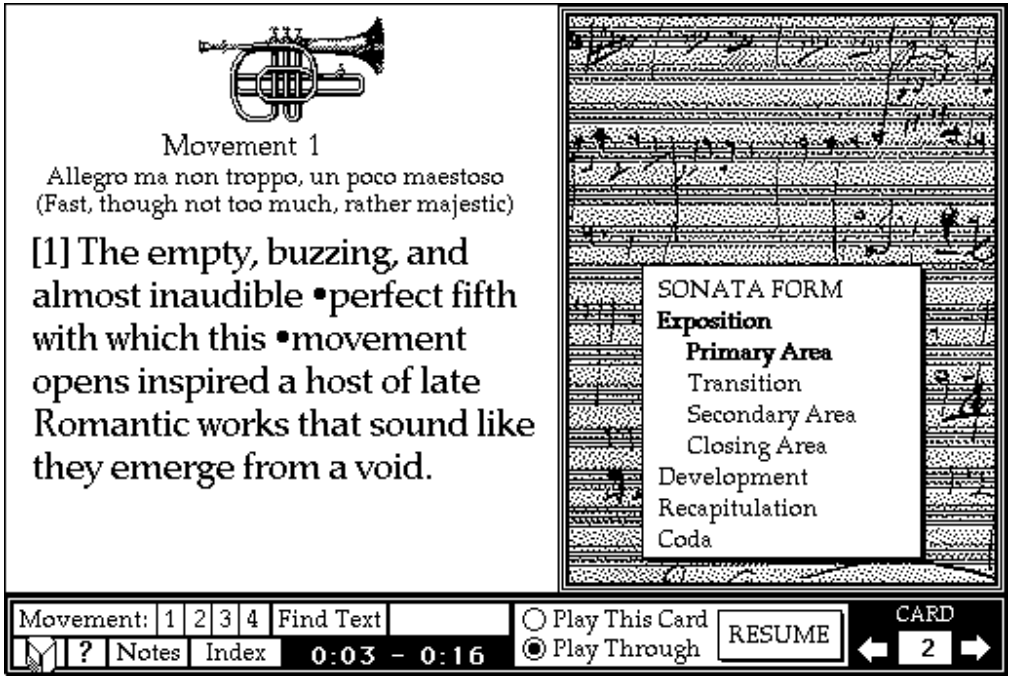
- The machine the prototype of which impressed Steve Jobs so much that he decided to produce the Lisa/Macintosh kind of computers for the mass market (1979)
  - Graphical user interface
  - Networked via Ethernet
  - Programming language Smalltalk
- Hardware:
  - 800 x 600 display
  - Data General 16 Bit processor
  - 400.000 instructions/second
  - 256 kByte – 512 kByte RAM
  - 2 x 2,5 MByte Festplatte

Animation Software on the Alto



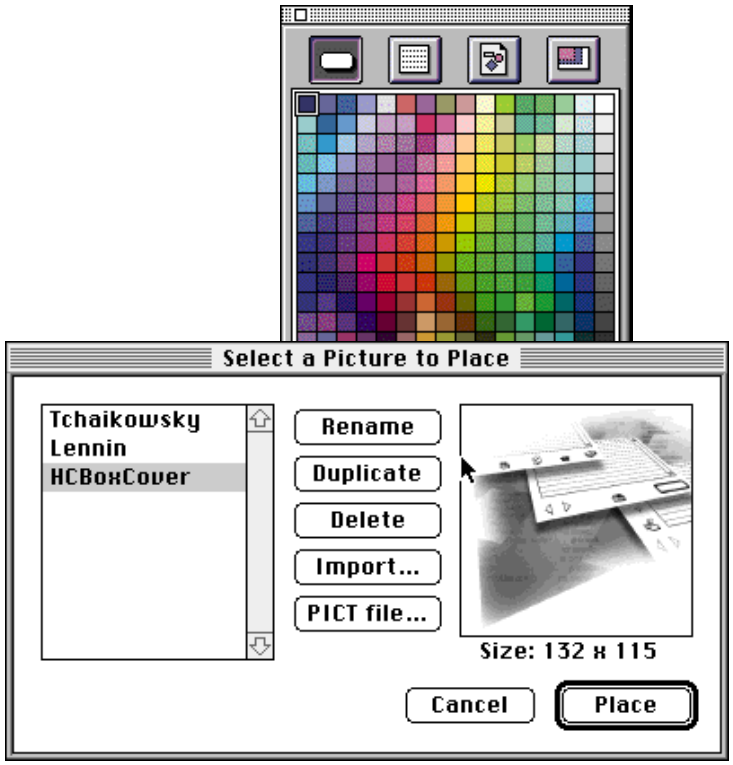
Hypertext Authoring Tools

- Visual design of user interface, integration of media (images, sound):
  - 1982, Peter Brown (Kent): Guide authoring system
  - 1987, Bill Atkinson (Apple): HyperCard authoring system (*HyperTalk* scripting)



MultimediaHyperCard stack (Voyager 1989)

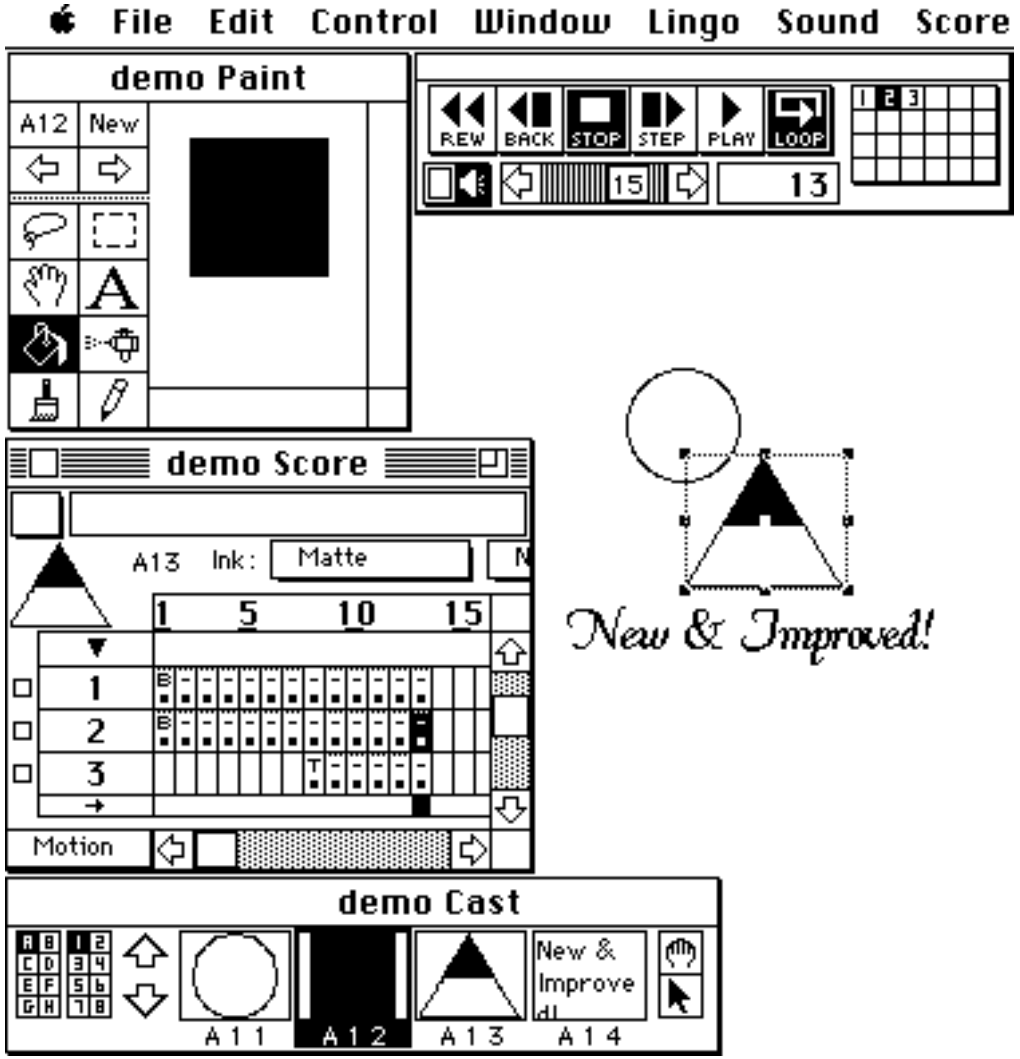
(Source for image: wapedia.mobi)



(Source for images: mactech.com)

Animation Authoring: VideoWorks

- Joe Sparks
- Macromind, 1985-88
- Later renamed to *Director*
- Used (for example) for multimedia tutorials on Apple MacOS
- Specialized scripting language *Lingo*



Timeline of Multimedia Programming History

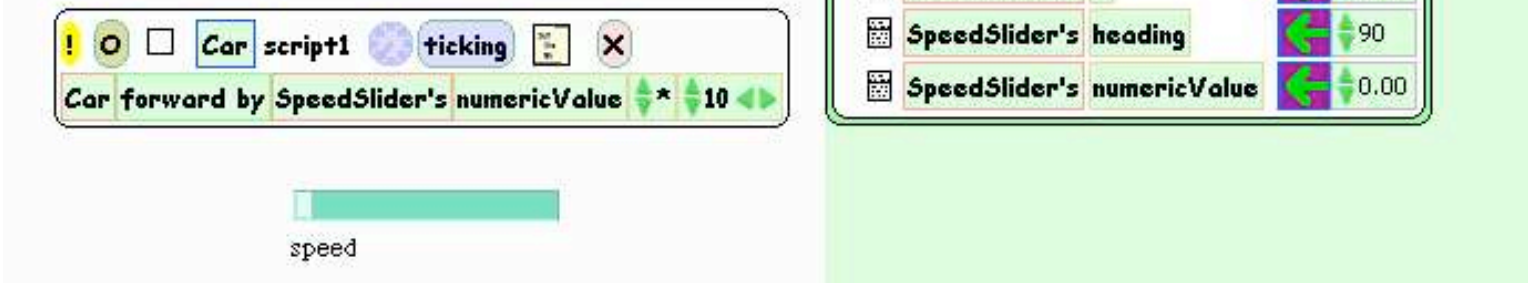
- 1963 – Sutherland: Sketchpad
- 1968 – Engelbart: NLS
- 1972 – Kay: Dynabook, Smalltalk
- 1979 – Xerox PARC: Alto
- 1982 – Brown: Guide authoring system
- 1985 – Sparks: VideoWorks
- 1987 – Atkinson: Apple HyperCard
- 1988 – Macromind Director
- 1989 – Kretz: Start of work on MHEG
- 1990s – Various multimedia education and gaming applications (CD-ROM)
- 1995 – Kay/Ingals/Kaehler: Squeak
- 1996 – Ackermann: MET++ Framework
- 1998 – W3C: SMIL
- 1997 – Macromedia Flash (ex *FutureSplash Animator* ex *SmartSketch*, by J. Gay)
- 2001 – Reas/Fry: Processing
- 2004 – ISO: MHEG-5
- 2004 – Adobe Flex
- 2004 – Bederson/Grosjean/Meyer: Piccolo framework
- 2005 – Oliver: F3 (later called JavaFX)
- 2007 – Microsoft Silverlight

Visual Multimedia Programming in Squeak

- 1995: Alan Kay, Dan Ingalls, Ted Kaehler at Apple
- Reintroducing multimedia features into Smalltalk
- Programming environment targeted at children (primary school level)

"Halo" menu

Visual scripts



2 Challenges in Multimedia Programming

2.1 Historical Background

2.2 Application Areas, Terminology

2.3 Challenges & Features for Development Platforms

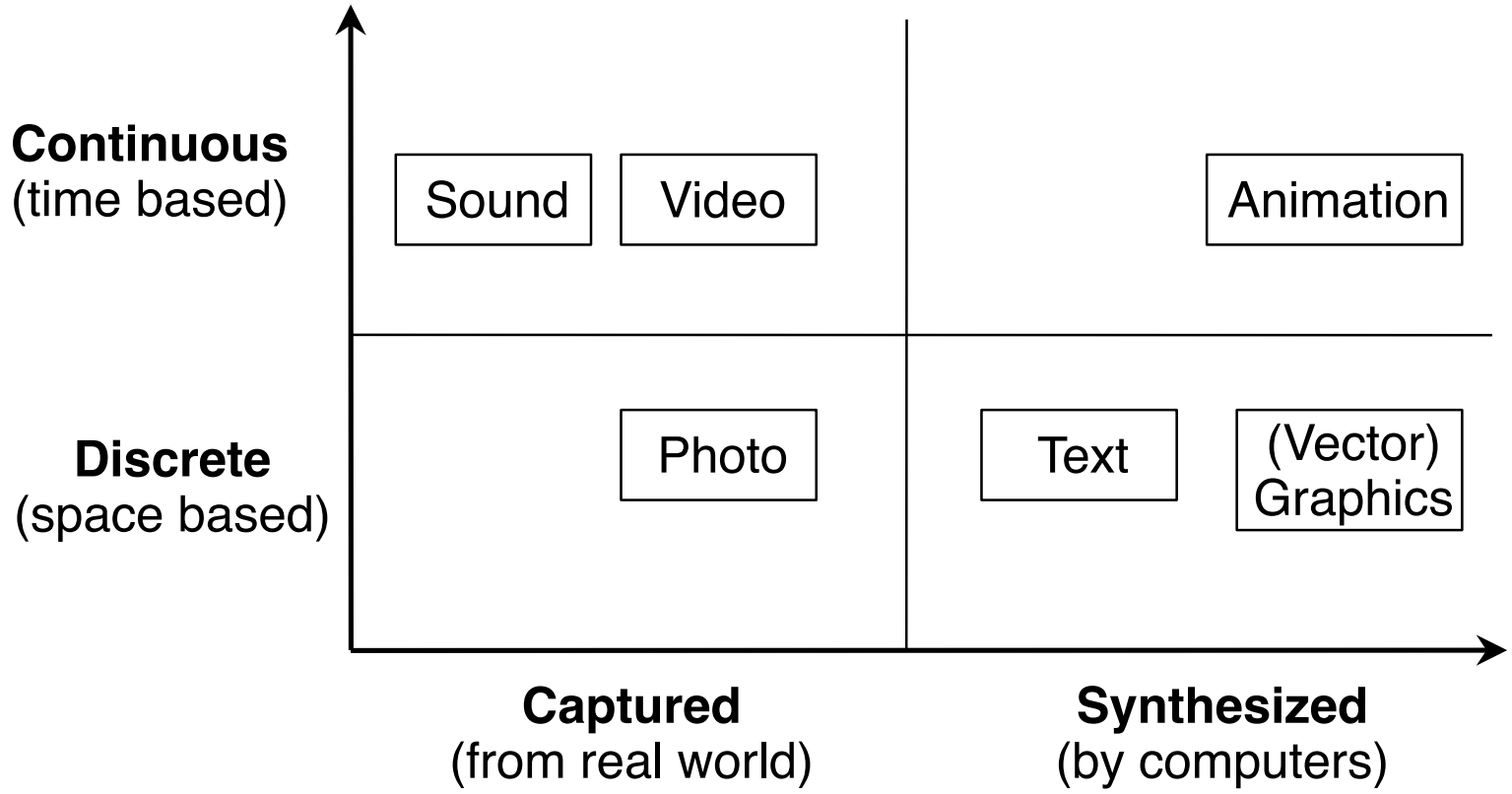
Multimedia Development

- Development of mostly interactive, possibly distributed, multimedia applications
- Typically carried out by “multimedia agencies” (Multimedia-Agenturen) or specialized software development companies
- Distribution channels: CD/DVD, Web, Broadcasting
- Position in the value chain:

Content Production	Application Production	Distribution Platform Provision
• Media industry • Traditional industry (e-commerce)	• Multimedia agencies • System integrators	• Telcos • ISPs (Internet Service Provider)
• Dev. Tools: Photoshop, Illustrator, 3DS, ...	• Dev. Tools: Flash, Flex, Director, Python, Java, …	• Dev. Tools: C, C++, …

Classification of Media Types

- Fetterman/Gupta 1993:



Multimedia vs. Multimodality

- *Modality*:
  - Channel used for communication with human user (human sense)
    - » See literature on Human-Computer Interaction
  - *Multimodal:* Using several different modalities
- Multimedia:
  - May combine media elements which belong to a single channel (i.e. visual channel: text, graphics, animation)
  - Does not require multiple channels (multimodality)
  - Combines media elements without analysing their inner structure
- Multimodality:
  - Often used for techniques which deeply *analyse* input information (e.g. speech, gestures)

Interactivity

- Degrees of interactivity (based on T.A. Aleem 1998):
  - Passive, Reactive, Proactive, Directive
- Application to multimedia (Heller et al. 2001) - Examples:

Media type ↓	*Passive*	*Reactive*	*Proactive*	*Directive*
Text	Sequential presentation	Page turner Linear spacing	Browsing, Hypertext	Word processing
Graphics	Sequential presentation	Predefined changes (choice between graphics)	Change of colors, sizes, shapes, …	Drawing graphics
Sound	Sequential presentation	Predefined changes (sound clip, volume)	Selection of track, fast forward, loop	Creation of sounds
Motion	Sequential presentation	Predefined changes (path, target of motion)	Start, stop, pause, forwd, reverse	Creation of animations

Application Areas and Examples

Domain → Interactivity ↓	*Business*	*Information*	*Communi-cation*	*Edutain-ment*	*Education*
Reactive	Online shop	Encyclopedia	News channel	Media player	Medical course
Proactive	Car configurator	Navigation system	Video conference	Car racing game	Flight simulator
Directive	Authoring tool	Wiki	CSCW System	City building game	Electronic circuit simulator

Based on Dissertation A. Pleuß (2009) and Projektarbeit S. Kraiker (2007)

Origin of Media Content

- *Received:*
  - Media content is produced elsewhere
  - Multimedia application only integrates existing content
- *Designed:*
  - Media content is created specifically for the developed application
- *Generated:*
  - Media content is automatically constructed from application data (e.g. maps, data visualizations, thumbnail overviews)

- In any case, copyright rules have to be taken care of!

2 Challenges in Multimedia Programming

2.1 Historical Background

2.2 Application Areas, Terminology

2.3 Challenges & Features for Development Platforms

Challenges for Multimedia Development (1)

- ***Main challenge 1: Multimedia programs are complex***
  - Challenge 1.1: *Multimedia data types are complex in themselves*

    Example: (Bitmap) pictures
    - » Many different external storage formats
    - » Complex encoding/decoding algorithms (e.g. JPEG)
    - » Graphics processing entails additional data complexity (e.g. usage of double buffering)
    - » Many standard operations are complex algorithms (e.g. convolution filters, affine transformations)

    Example: Video objects
    - » Rendering requires special playback components
    - » Rendering involves time-critical operations

Basic idea for solution: Re-use prefabricated code

Challenges for Multimedia Development (2)

- ***Main challenge 1: Multimedia programs are complex***
  - Challenge 1.2: *Multimedia objects have high data volume*

Example: Video data (similar challenges for pictures and audio)

» Pre-loading all video data is not realistic

» Loading executed in parallel to other activities of the program

» Techniques for processing partial data (e.g. streaming)

» Consumer-producer synchronization

» Conceptually unlimited data volume is possible
(e.g. live audio, live camera, steerable camera)

Basic idea for solution: Use multi-threaded runtime execution platform

+ Re-use prefabricated code

Challenges for Multimedia Development (3)

- ***Main challenge 1: Multimedia programs are complex***
  - Challenge 1.3: *Multimedia requires synchronization of parallel activities*

    Example: Audio/image synchronization
    - » Slideshow style: Audio in parallel to animation
    - » Game style: Flexible mixing of audio output according to situation
    - » Harmonization of natural durations for sounds, video, animations
    - » Long-running parallel activities need fine-grained synchronization (e.g. per video frame)

Basic idea for solution: Use specialized concepts for time control

+ Use multi-threaded runtime execution platform

Challenges for Multimedia Development (4)

- ***Main challenge 1: Multimedia programs are complex***
  - Challenge 1.4: *Interactive multimedia requires flexible programming schemes*

Example: Proactive and directive interactivity
e.g. in an interactive kitchen furniture planner

» Great variety of user actions
(adding an element, placing, modifying, specifying view, ...)

» Program has to react to an unforeseeable sequence of actions

» Program should be always responsive and provide clear feedback

Basic idea for solution: Use event-driven programming

+ Re-use prefabricated code

Challenges for Multimedia Development (5)

- ***Main challenge 2:***
  Multimedia content is often created by non-technical people
  - Challenge 2.1:
    Interface with established media production workflows/tools

    Example: Import from drawing, sketching, video editing programs
  - Challenge 2.2:
    Enable software development involving non-technical team members

    Examples: Professional animators doing computer animation,
    Graphic designers doing interface design

Basic idea for solution: Hide technicalities, provide adequate tools

Categories of Development Environments

Engels and Sauer 2002:

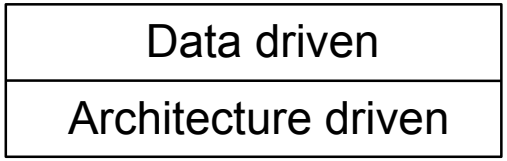
- Frameworks and APIs (Application Programmming Interfaces)
  - Based on traditional programming and scripting languages
  - Description of control flow (execution order)
  - Development of text with editors and IDEs
- Declarative Languages
  - Specialized languages for multimedia
  - Description of multimedia documents
  - Development of text with editors and IDEs (often XML-based)
- Authoring Tools
  - Specialized development environments
  - Visual metaphors for document/application features
  - Drag & drop, property editors, drawing tools etc.

Frameworks

- **Definition** (Taligent): “A ***framework*** is a set of prefabricated software building blocks that programmers can use, extend, or customize for specific computing solutions.”
- **Definition** (nach Pomberger/Blaschek): “A ***framework*** (Rahmenwerk, Anwendungsgerüst) is a collection of classes which provides an abstract design for a family of problems”
- Goals:
  - Reuse of code, architecture and design principles
  - Reuse of schematic behaviour for a group of classec
  - Homogeneity among different application systems for a problem family (e.g. similar usability concept)

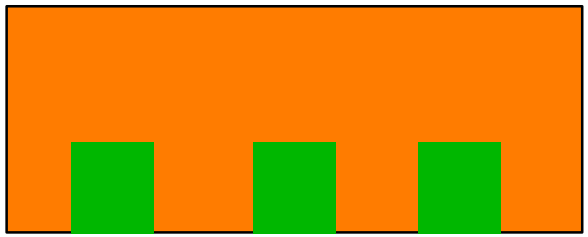
Classification of Frameworks

- Architecture driven framework:
  - Adaption by inheritance and method override
  - Complex class hierarchies and patterns
  - Adaption requires excellent programming skills and steep learning curve
  - Examples: Java Media Framework (JMF), MET++
- Data driven framework:
  - Adaption by object creation and setting of object properties
  - Delegation mechanisms (chaining of objects, events as objects)
  - Easier to learn but less flexible
  - Example: Pygame
- Compromise: Two-Level architecture:

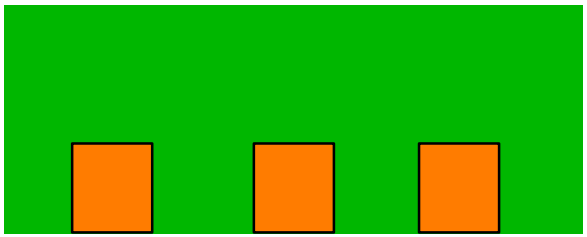


Class Library vs. Framework

Class library



Framework



Application specific parts

Prefabricated parts

"Don't call us, we call you"

(“Hollywood Principle”)

Adaptation by instantiation mainly

Control flow not pre-defined

Adaptation includes specialization

Predefined control flow

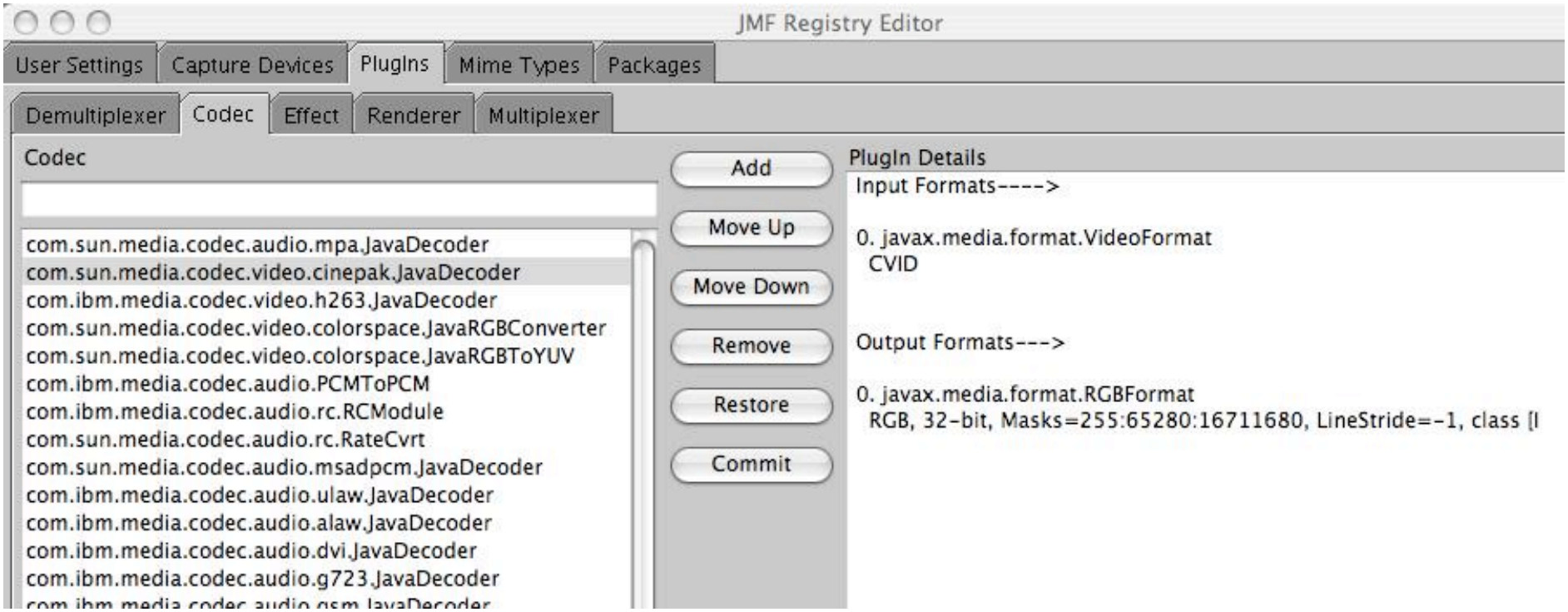
Examples for Development Environments

- Frameworks and APIs
  - DirectX
  - Java Media Framework
  - MET++ (for C++)
  - Piccolo (for C++ and Java)
  - **Pygame (for Python)**
- Declarative languages:
  - **SMIL**
  - **OpenLaszlo**
  - MHEG-5
- Authoring tools:
  - Adobe Director
  - **Adobe Flash**
- Hybrid approaches:
  - Declarative language integrated with framework: **JavaFX**
  - Specialization of programming language & framework: **Processing**

Example for a Universal Abstraction: Stage

- Multimedia application as visual interface
  - Integration into interface/window framework
  - Root for time and space containment hierarchy
- Examples:
  - *Display* in Pygame
  - *Layout* in SMIL
  - *Canvas* in OpenLaszlo
  - *Stage* in JavaFX, Flash/AS
- Functions:
  - Define size of display area
  - Define general properties of display area (color space etc.)
  - Set window caption

Example for Extensibility: Codec Plugin Architecture in JMF



Specification Paradigms for Timing

- Formal language
  - Programming language:
    - » Control flow defines timing
    - » Expressiveness achieved through constructs for concurrency: Threads, active or passive waiting (Example: Python/Pygame)
  - Declarative specification language:
    - » E.g. temporal logic expression (“X is repeated until Y” etc.)
- Time functions (time line)
  - Basic principle: Function from time value to parameter value
  - Parallel *tracks* to express concurrency (Example: Flash)
- Event composition
  - Implicit ordering given by event processing
  - May include temporal relations for events (like *before, meets, overlaps, during, after, …*)

Variations of Time Functions

- *Time function*:
  - Maps a time value onto a parameter determining the audio/visual presentation (Concept from MET++)
  - Various *interpolation strategies* are used to compute intermediate values
    - » May affect performance of individual media elements (e.g. *local time warping* in MET++)
- Time line in JavaFX:
  - General mechanism to compute parameter values
  - Playable sub-presentation (time container)
- Time line in Flash:
  - Using parallel tracks (from visual authoring metaphor)
  - Time lines may be nested (objects having their own time line)

References

- G. Engels, S. Sauer, “Object-oriented Modeling of Multimedia Applications”, in *Handbook of Software Engineering and Knowledge Engineering*, S.K. Chang (ed.), Singapore: World Scientific 2002, pp. 21-53.
- R.L. Fetterman, S, K. Gupta, *Mainstream Multimedia: Applying Multimedia in Business.* New York: Van Nostrand Reinhold 1993
- R.S. Heller, C.D. Martin, N. Haneef, S. Gievska-Krliu, “Using a theoretical multimedia taxonomy framework”, *J. Educ. Resour. Comput.*, vol. 1, p. 6, 2001
- R. Steinmetz, K. Nahrstedt, *Multimedia Applications*, 1st ed. Berlin: Springer 2004