MET++ Project Presentation

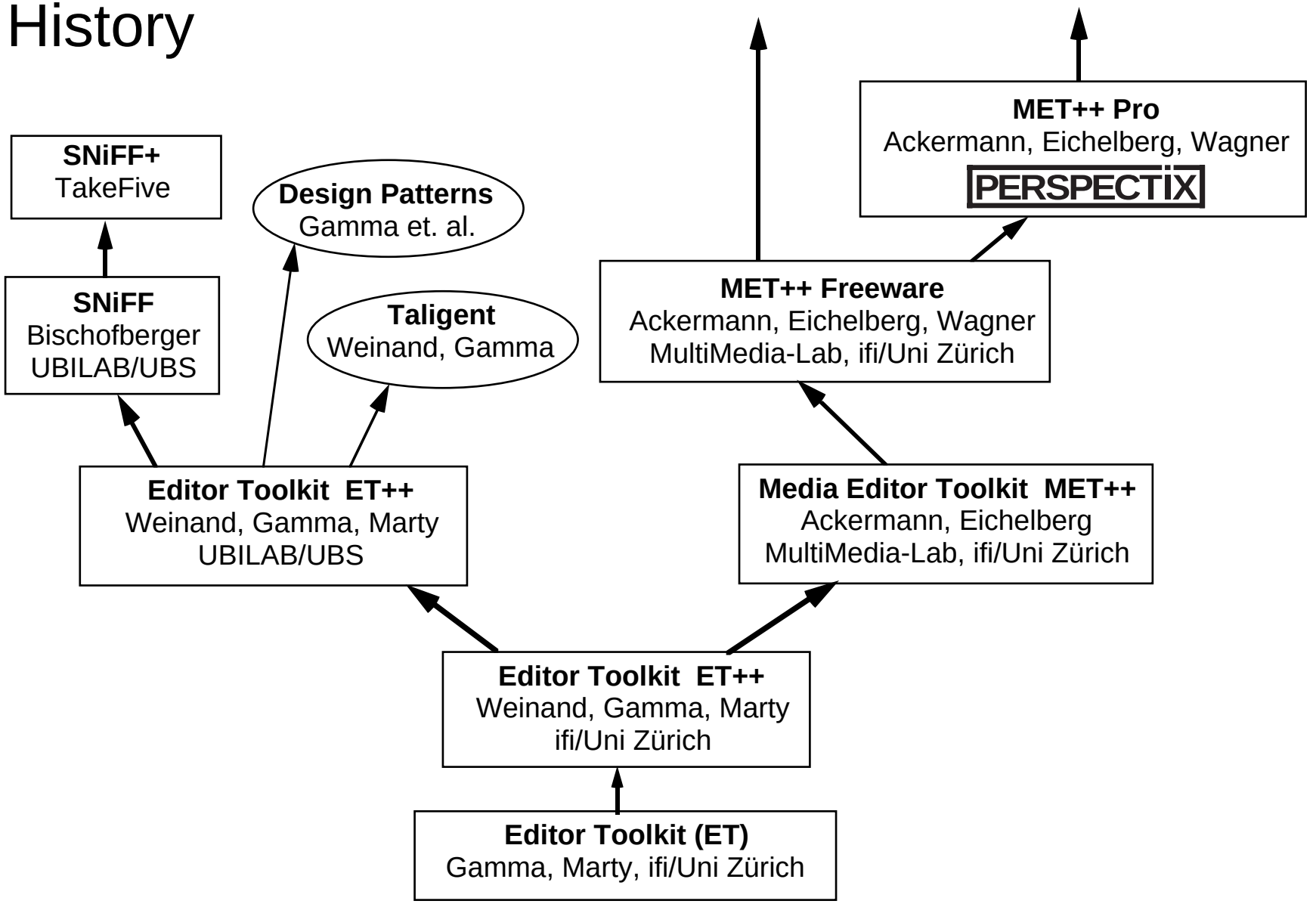
Introduction

- **Philipp Ackermann**
  - ⇨ PERSPECTIVE
  - ⇨ MultiMedia Laboratory (MML)
    University of Zurich

- **Overview**
  - ⇨ system architecture
  - ⇨ application domain
  - ⇨ visual object composition
  - ⇨ lessons learned

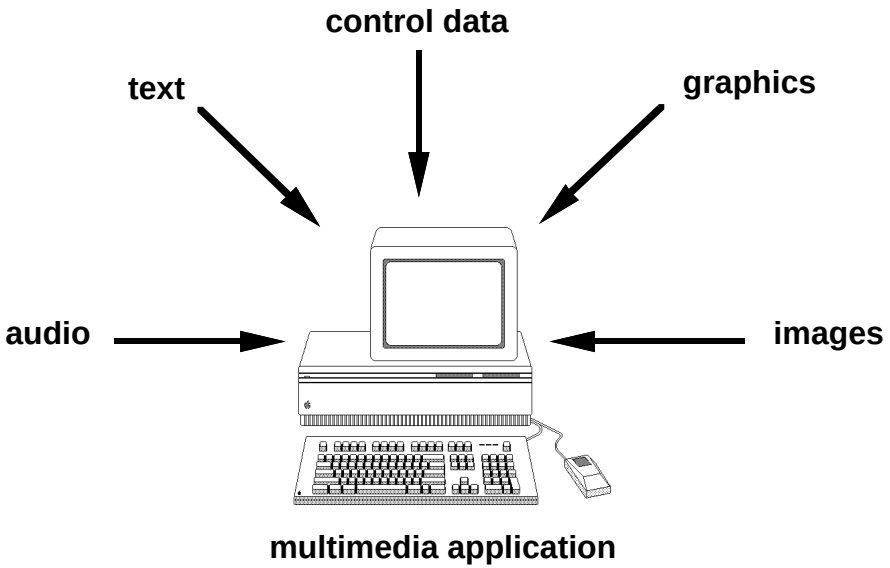
The MET++ Application Framework

History

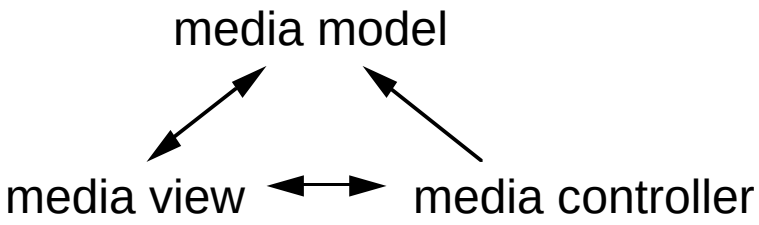


Multimedia Requirements

Seamless Integration



Media-MVC



Coexistence

Seamless Integration

MET++ Overview

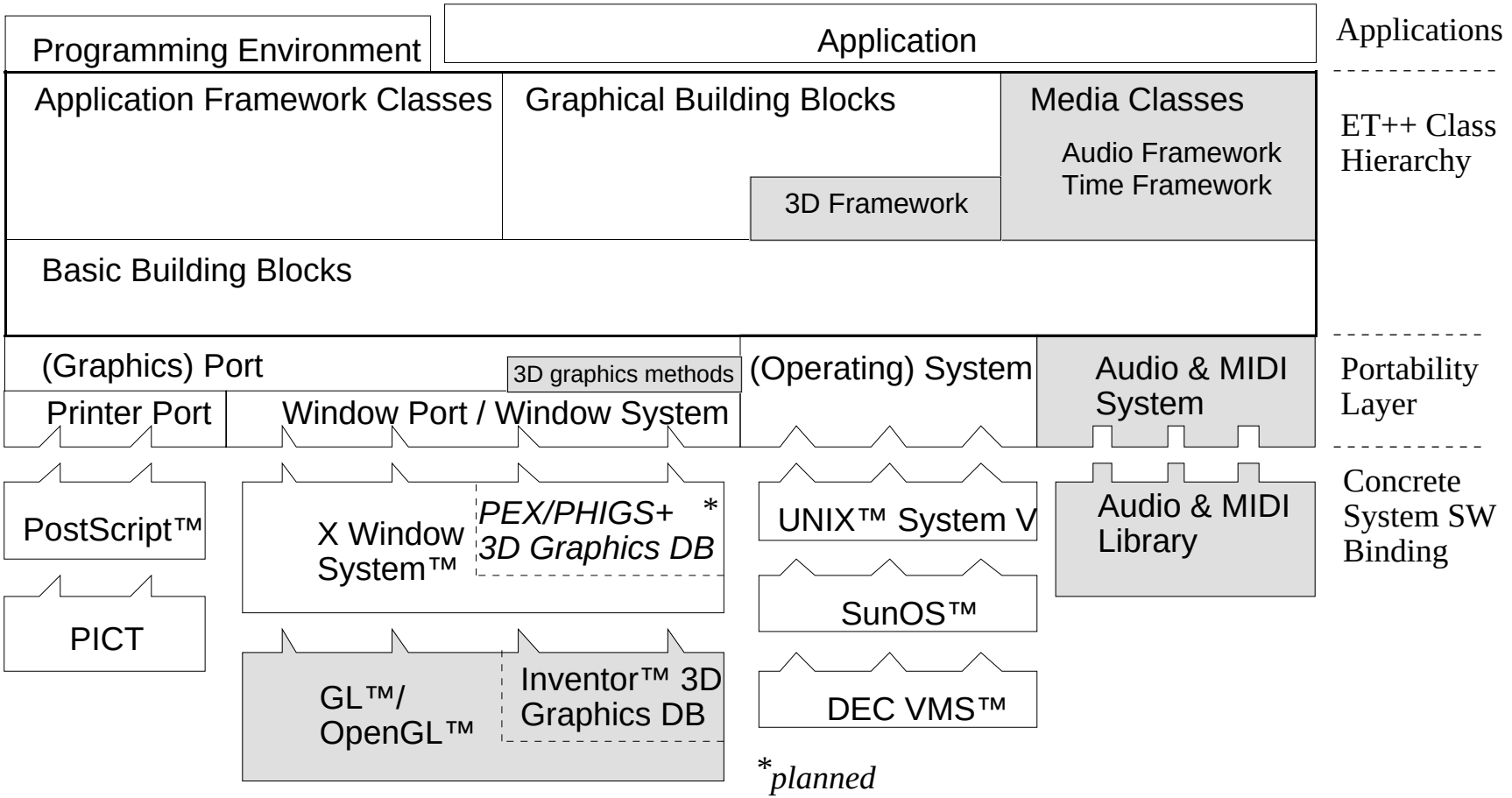
Features

- **object-oriented media application framework**
  - ⇨ abstract & concrete media classes + object collaboration
  - ⇨ not only coexistence but real integration
  - ⇨ reuse of code and of design
  - ⇨ seamless integration
- **flexibility through object composition**
  - ⇨ portability layer classes (system configuration)
  - ⇨ media composition structure
  - ⇨ composition of visual components
  - ⇨ visual programming environment
- **direct manipulation of graphical representations**
  - ⇨ different views of the same data model (MVC)
  - ⇨ interactive composition
  - ⇨ immersive interaction

System Design

Multimedia Application Framework MET++

- **system architecture**



Data Converters

I/O with automatic type checking

- **Text:**
  - ⇨ ASCII, RTF, HTML
- **Images:**
  - ⇨ TIFF, EPSF, PICT, PBM, XPM, JPEG, GIF, RGB, TGA
- **3D Models:**
  - ⇨ DXF, STL, VRML
- **Audio:**
  - ⇨ Apple/SGI AIFF, Sun/DEC/NeXT SND, MS WAVE/RIFF
- **Music:**
  - ⇨ Standard MIDI File (SMF), CSound Score File
- **Video:**
  - ⇨ only on SGI: MPEG, Motion-JPEG

Sample Applications

Custom-specific Apps in Several Domains

- **Multimedia**
- **Hypermedia**
- **Visualization**
- **Augmented User Interfaces**
- **Image/Video Processing**

Object Composition

Black Box Reuse

Design Patterns	**2D Graphics**	**3D Graphics**	**Temporal Structures**	**Audio/Video**
Composites	Groups/Layers - *x-y-move, stretch* Box - *automatic layout* OneOfBox - *selection*	Groups - *x-y-z-move, stretch, rotate* - *property inheritance* LOD - *selection*	Groups - *temporal transform.* Sequence, Synchro - *automatic layout* Time Functions - key frame interpolat. Time-Value-Pairs	Source-Filter-Sink Data-Flow Mixer
Decorators	Inlines/PasteLinks - *load on demand* Border Scroller Hyperlink - *action*	Inlines - *load on demand* Hyperlink - *action*	Repeat Loop	Filter Analyzer
Wrappers/ Adaptors	GUI Elements Embedded Docus - text views - 3D views - media views		Media Wrappers - animations - audio/music - video	Synthesizer Display Frame Grapper

generic parameter control ?

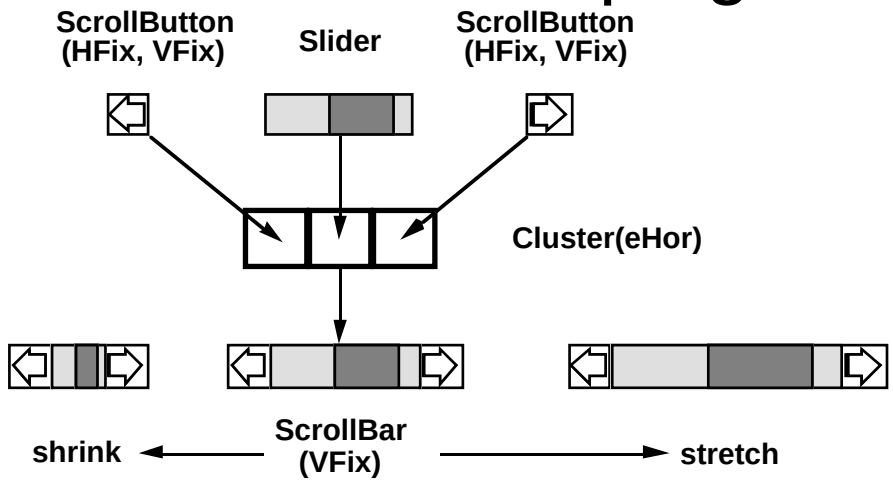
Components Run-time Environments

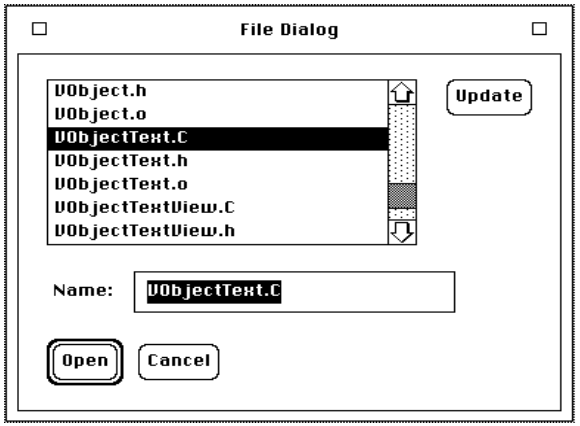
Black Box Reuse

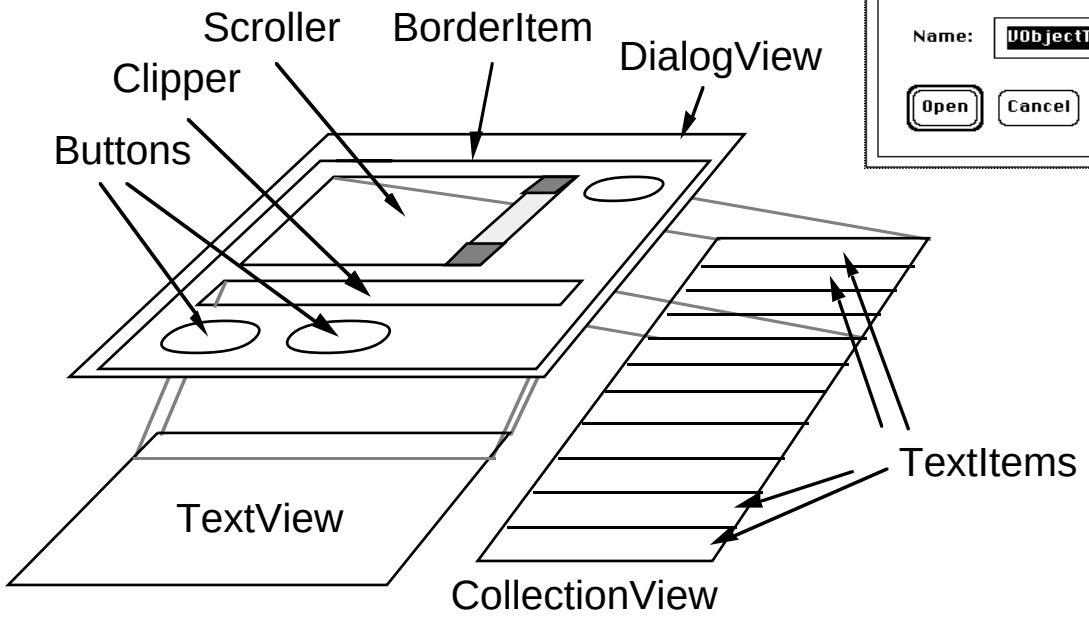
- **Scripting (not favored and not yet supported)**
- **Visual Composition**
  - ⇨ Trees, Graphs
  - ⇨ Graphical elements
    - → Box, Shapes, User interface elements, ...
  - ⇨ Temporal elements
    - → Sequence, Synchro, Loop, Repeat, Time Functions, Wrappers
  - ⇨ Media (Visuals, Conductor) <----- MediaView
- **Hyperlinking**
  - ⇨ Hypertext, hypermedia, information web
  - ⇨ Hot words, hot images, buttons
- **Visual Data Flow Programming**
  - ⇨ Audio Signal Flow
  - ⇨ Video Signal Flow
  - ⇨ 'Data' Flow of simple types (not objects)

Graphics Composition

2D and 3D Grouping



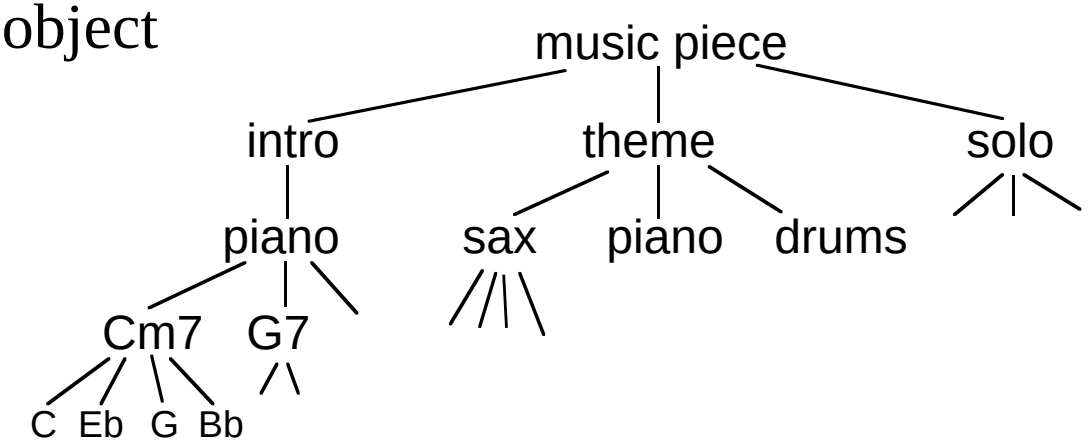




Time Framework

Synchronization

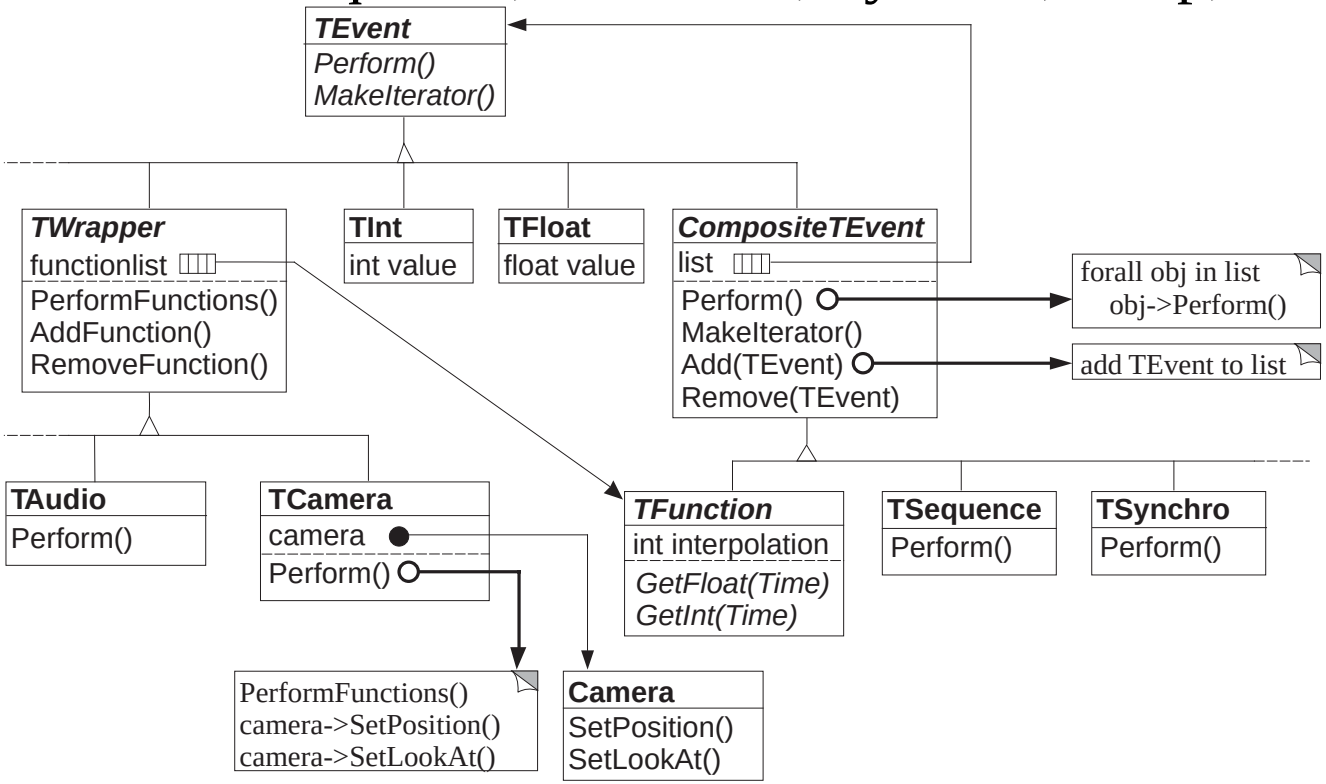
- **Temporal Composition**
  - ⇨ hierarchical structure
- **Generic Grouping and Object-specific Wrappers**
  - ⇨ generic visualization: Event Graph and Time Line
  - ⇨ automatic temporal layout
- **Temporal Transformation**
  - ⇨ move, expand/reduce, stretch/shrink
- **Time Functions**
  - ⇨ to control wrapped object



Time Synchronization

Time Events

- **interleaving time and media objects**
- **flexible composition mechanism**
  - ⇨ hierarchical object composition (grouping)
  - ⇨ classes for Sequence, TimeLine, Synchro, Loop, Shift



Time Synchronization

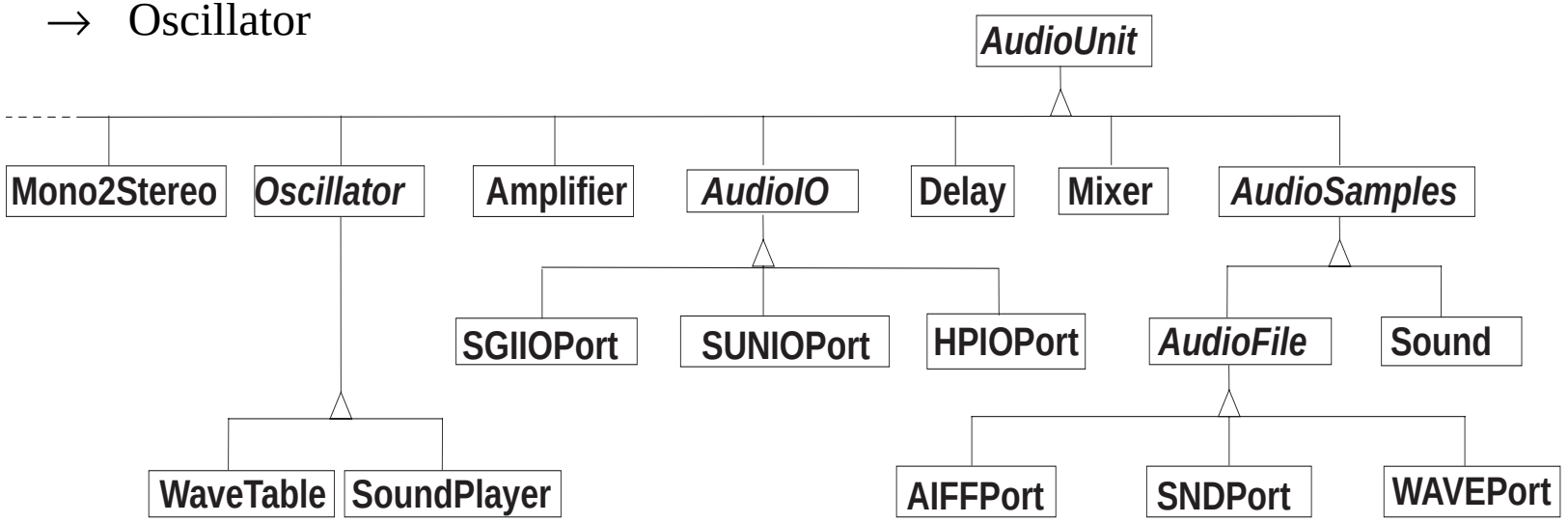
Timed Media Objects

- **timed media objects**
  - ⇨ subclasses of TEvent/TWrapper
  - ⇨ defining dynamic behaviour with time functions (no scripts)
  - ⇨ time functions for boolean, integers, floats, text, time, color
- **predefined classes for**
  - ⇨ visual objects (TVObject)
  - ⇨ text (TTextItem)
  - ⇨ 3D objects (TThreeD)
  - ⇨ camera (TCamera)
  - ⇨ light (TLight)
  - ⇨ audio units (TAudio)
  - ⇨ audio files (TSamples)
  - ⇨ music (MusicPlayer)
  - ⇨ MIDI controlled audio mixing console (TDMR8)
  - ⇨ movies (TMovie)

Audio

Audio Unit

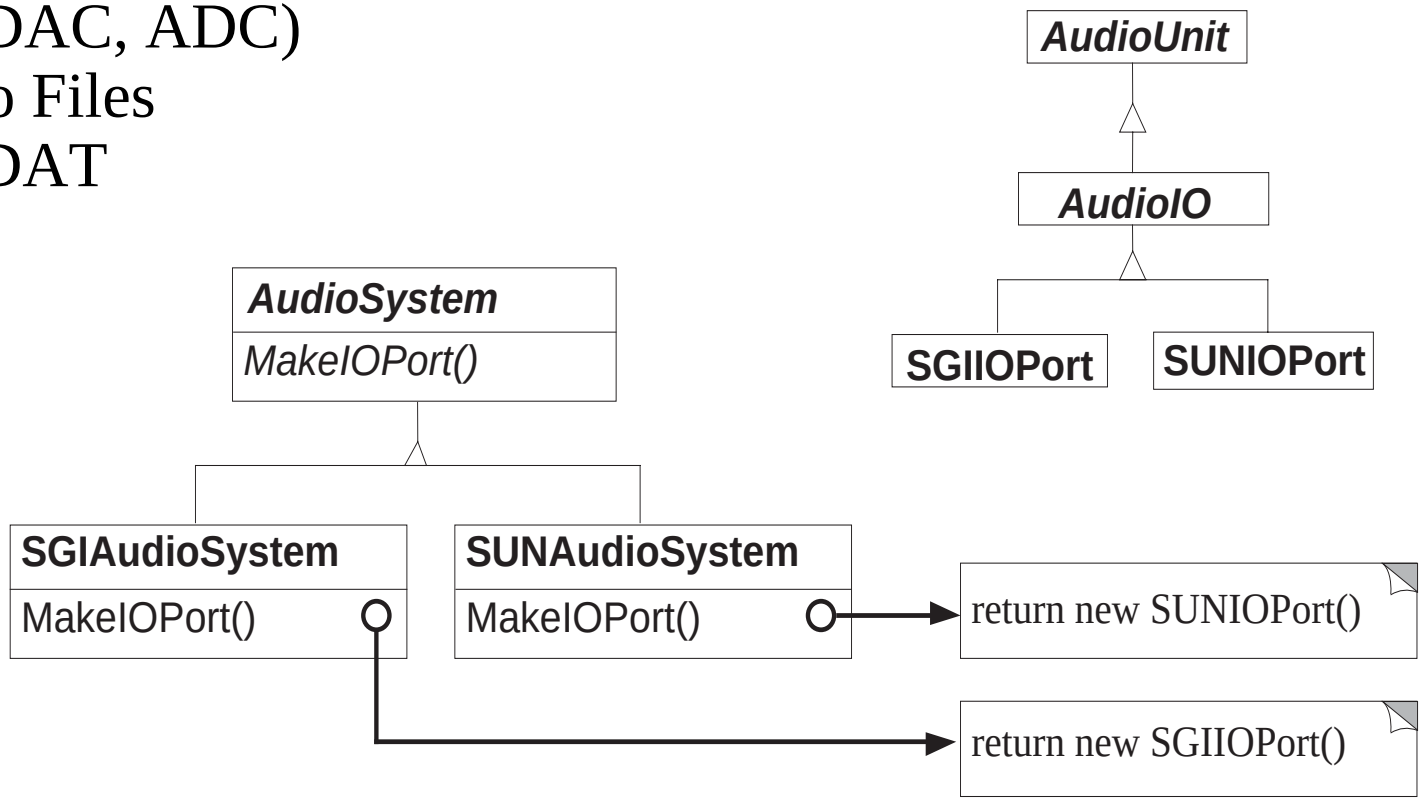
- **source-filter-sink architecture**
  - ⇨ audio signal flow as directed acyclic graph
  - ⇨ simple sample processing
  - ⇨ format conversion (sample rate, bits per sample, channels)
  - ⇨ system independent units
    - → Amplifier, Mixer
    - → Delay
    - → AudioAnalyzer
    - → Oscillator



Audio

AudioSystem

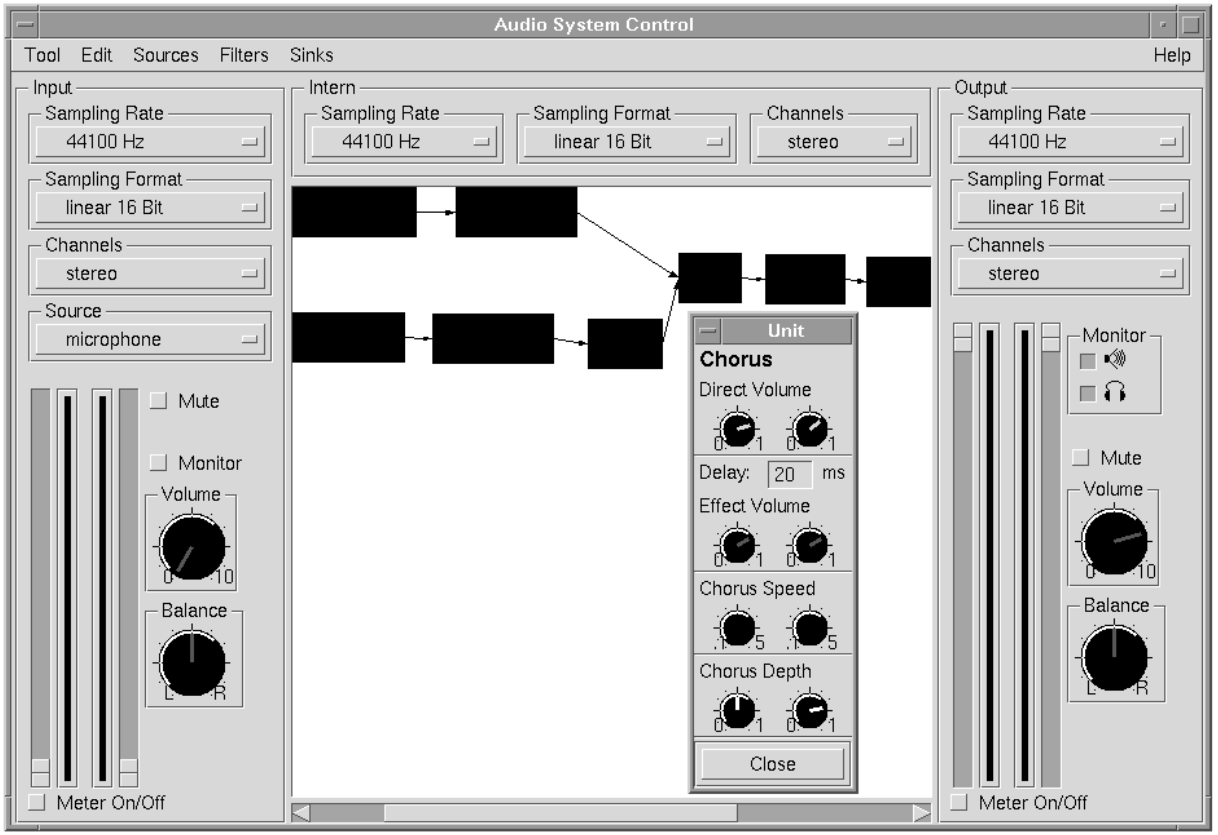
- **system configuration**
- **manages audio units**
- **abstract factory**
  - ⇨ I/O (DAC, ADC)
  - ⇨ Audio Files
  - ⇨ CD, DAT



Audio

Audio System Editor

- **Direct Manipulation**
  - ⇨ input/output selection
  - ⇨ input/output volume
  - ⇨ peak meter
  - ⇨ monitoring
  - ⇨ sampling rate
- **Audio Signal Graph**
  - ⇨ visualization of signal flow



Hyperlinking

Hyperstructure and Web Connectivity

- **VObjectText**
  - ⇨ Annotations
  - ⇨ Links, PasteLinks
  - ⇨ Embedded Documents
- **URLData**
  - ⇨ stream abstraction for
    - → HTTP
    - → FTP
  - ⇨ collaborates automatically with Converters
- **LinkManager**
  - ⇨ Name Resolving and File Searching
  - ⇨ File Caching
- **Navigator**
  - ⇨ Link History
  - ⇨ Notify over Control()

```
gLinkManager->GetData(char*)
```

Visual Programming

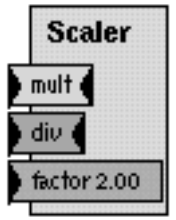
Reuse in Application Frameworks

- **Software Reuse**
  - ⇨ White-box Reuse
    - → Reuse by Subclassing
  - ⇨ Black-box Reuse
    - → Object Composition
- **Visual Programming**
  - ⇨ Visual Object Composition
    - → Interactive connecting of pluggable components.
    - → Advantages of black-box reuse combined with comfortable handling, i.e. no coding, compiling, or linking, easy specification.
- **Component Run-time Environment**
  - ⇨ Bidirectional data-flow
  - ⇨ high granularity (simple data types, not objects)
  - ⇨ separation of data flow and GUI elements

Visual Programming

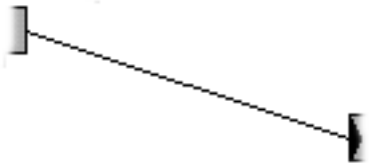
Building blocks

- **Data Units**



  - ⇨ Data Repository
    - → Read/write (Single Data Item, Data Array, NetCDF, etc.)
    - → Read only (Constants, Timer, Clock)
    - → User interface component (menu, slider, checkbox, radio button, etc.)
  - ⇨ Data Filter
    - → Basic arithmetic functions (Scaler, Shifter)
    - → Mathematical functions (trigonometric, logarithmic, etc. and their inversions)
    - → Conditional functions (if-then-else, threshold)
  - ⇨ Data Mapper
    - → 2D, 3D, Audio, MIDI, TextTable
- **Data Ports**
  - ⇨ Dependent
  - ⇨ Independent

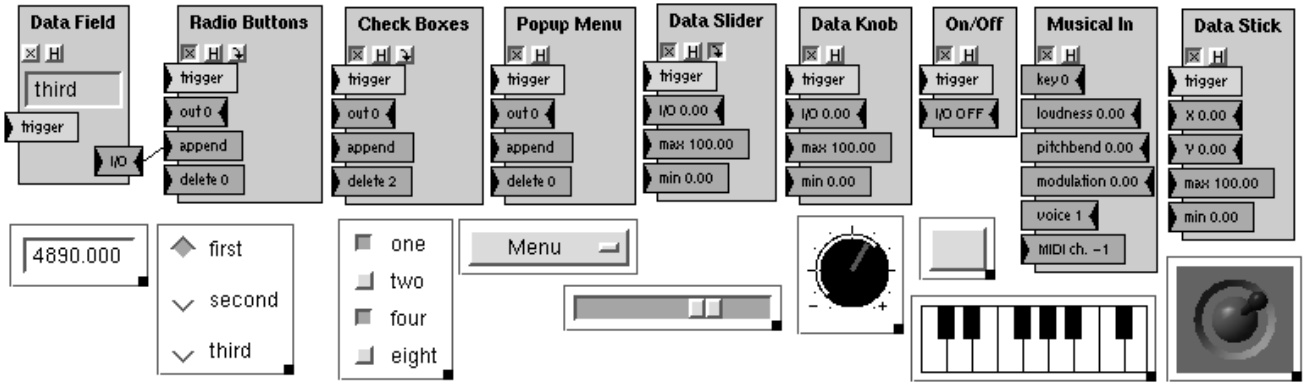
- **Connectors**
  - ⇨ Bi-directional



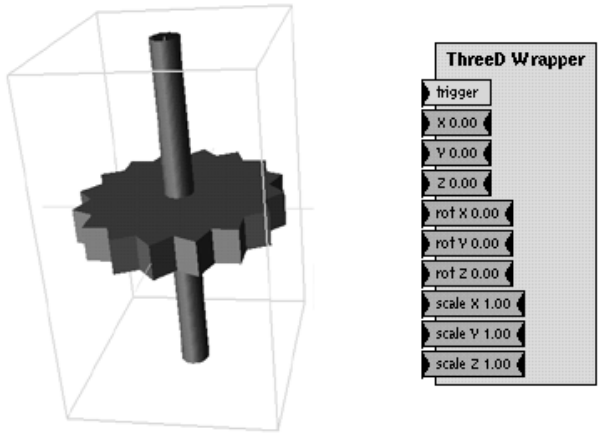
Integration with MET++

Integration on two levels

- **User Interface Builder**
  - ⇨ Drag & Drop of GUI-components
  - ⇨ Separation of GUI components and visual program



- **Media Wrappers**
  - ⇨ Visual Objects
  - ⇨ Temporal Events



Bi-direction

Bidirectional data flow

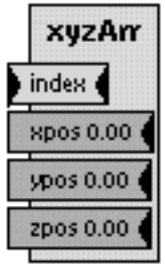
- **Repositories**
  - ⇨ Reading
  - ⇨ Writing
- **User Interface Components**
  - ⇨ Manipulation by user
  - ⇨ Setting by visual program
- **Filters**
  - ⇨ Function, e.g. e^x
  - ⇨ Inverse, e.g. ln(x)
- **Mappers**
  - ⇨ Setting by visual program
  - ⇨ Manipulation by user

Applications

Visual Programming for Multimedia

- **Visualization/Sonification of Data**
  - ⇨ self-describing input file format (netCDF)

```
⇨ netcdf xyzArr {
dimensions:
     index= 6;
variabes:
     double xpos (index);
     double ypos (index);
     double zpos (index);
data:
     xpos= 1.0, 3.0, -2.0, 4.0, -5.0, 4.0;
     ypos= 1.0, 3.0,  1.0,-3.2,  2.0, 4.0;
     zpos= 0.6, 3.1, -4.0,-3.2,  3.0, 1.0;
}
```




```
netcdf plane6x4 {
dimensions:
     x= 6;
     y= 4;
variables:
     long zpos (x,y);
data:
     zpos=0,1,1,2,1,1,1,1,1,0,4,1,
          1,4,0,1,1,1,1,1,2,1,1,3;
}
```

Applications

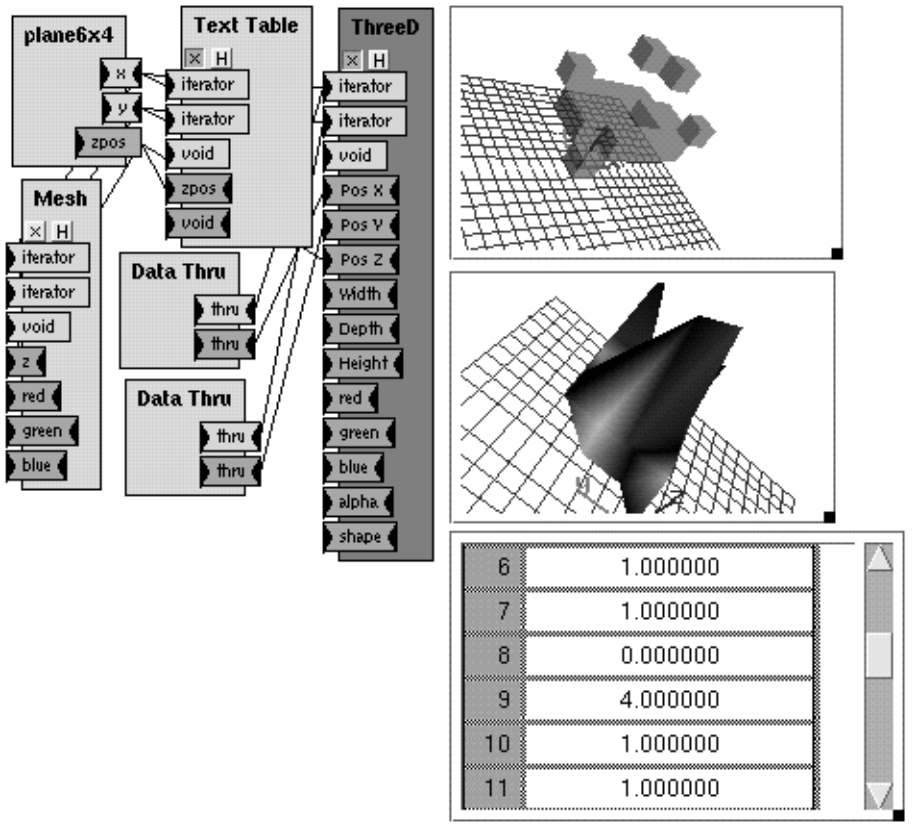
Visual Programming for Multimedia

- **Visualization/Sonification of Data (contd.)**
  - ⇨ generic mappers

```
netcdf plane6x4 {

dimensions:
        x = 6;
        y = 4;
variables:
        float zpos (x,y);
data:

    zpos=
          0,1,1,2,
          1,1,1,1,
          1,0,4,1,
          1,4,0,1,
          1,1,1,1,
          2,1,1,3;
}
```



Applications

Visual Programming for Multimedia

- **CAD**
  - ⇨ Parametric Construction
    - → Constraining of interactive manipulation
    - → Maintaining relationships between 3D objects
- **Redundancy-Free Animation Specification**
  - ⇨ Maintaining relationships between 3D objects
  - ⇨ Animation of Keyplayers only
- **Generic Media Patcher**
- **Executable Documents**
  - ⇨ Embedded Functionality in Documents
  - ⇨ Distributable Object Clusters (WWW)
  - ⇨ *better than scripting*
    (a JAVA killer?)

Future Plans

Open Issues, Future Work

- **Ongoing Work**
  - ⇨ Image Processing
  - ⇨ Portability
    - → LINUX, PEX, MESA, SoundBlaster Audio, MIDI
    - → MacOS, QuickDraw3D
  - ⇨ Visual Programming
    - → grouping (ICs)
    - → executable documents
- **Future Work**
  - ⇨ Video on X11
  - ⇨ Windows NT
  - ⇨ Documentation
  - ⇨ Applications

Official Release 1.0 Summer '96

Book with CD-ROM

dpunkt Verlag

Conclusion

Lessons learned

- **Software Crisis**
  - ⇨ Machine vs. Model Abstractions
- **Frameworks as pre-integrated architectures**
  - ⇨ solutions for a specific problem domain
- **Iterative Development Process**
  - ⇨ changing perspectives
  - ⇨ shift from analysis to implementation
  - ⇨ lean SW production
- **From formal/abstract to empirical/concrete**
  - ⇨ a new paradigm of computer science?