Agenda -- continued

Detector		
• CDS -- Software configuration control		Rolf Bork
• Possible detector design trades for the future		Robbie Vogt
LUNCH	1245 - 1315	
LIGO Modeling Environment	1315 - 1400	
• Programming environment survey report & recommendations		Hiro Yamamoto
• Demonstration		Hiro Yamamoto
• Software tools organization - status		Andy Kuhnert
Specifications, Requirements, and Interfaces	1400 - 1500	
• Preliminary interface definitions: VE/BT/Facilities		Gerry Stapfer
• Science Requirements Document		Rai Weiss
• Operations		Fred Raab
• System Specification Status		Albert Lazzarini
Conclusion and Actions	1500- 1530	
• General Discussion		All

LIGO End-to-End Modeling

Feb. 9, 1995

Hiro Yamamoto

- Why do we need it ?
- What do we need ?
- How would it look like ?
- Candidates ?

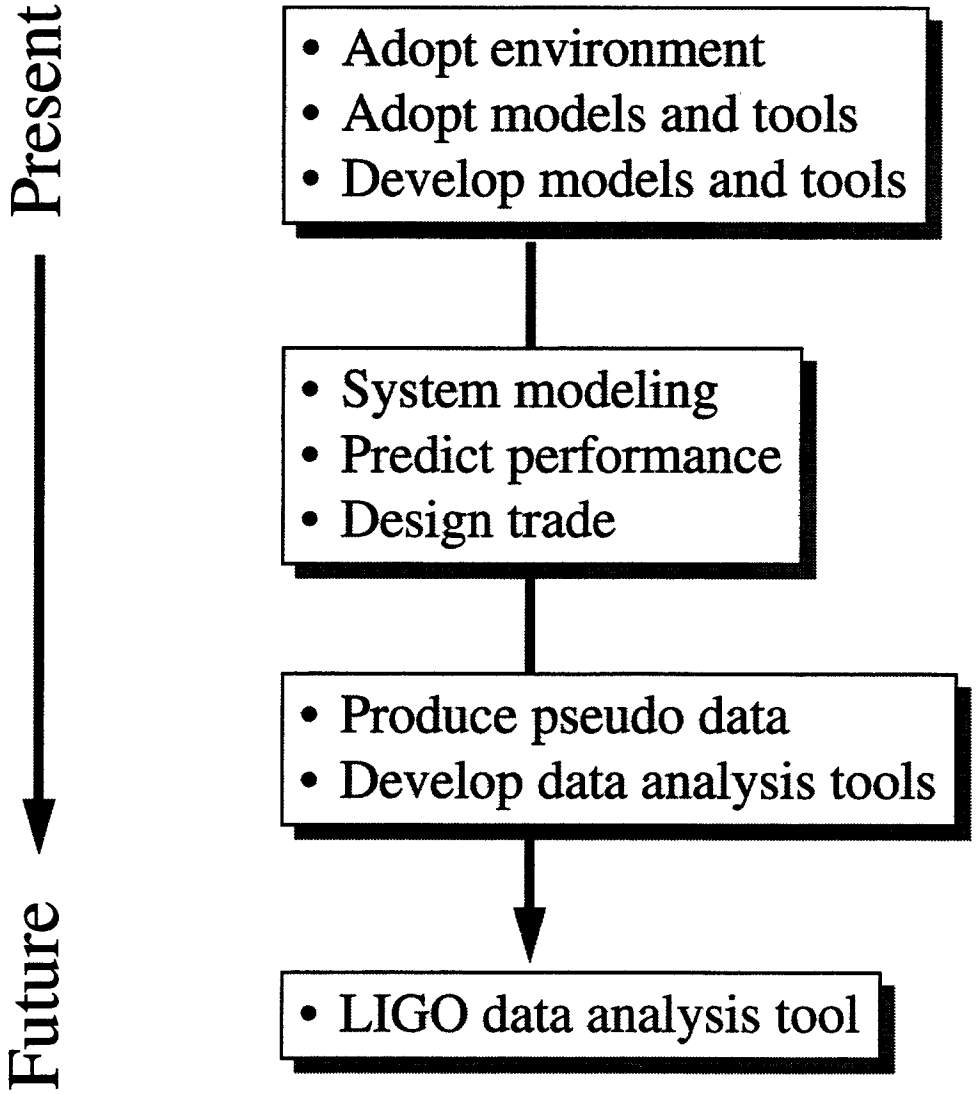
Why do we need - for LIGO

- Understand apparatus behavior
- Design trade study
  - Predict performance
  - Try different models
- Easy integration of work in LIGO
  - Work going on using different flavors - c / Fortran / MatLab / etc.
  - Use same environment to reuse work
- Data analysis
  - Compare models' predictions and data
  - on-line preselection algorithm
  - off-line search algorithm
- Make life easier
  - Friendly interface
  - Data visualization

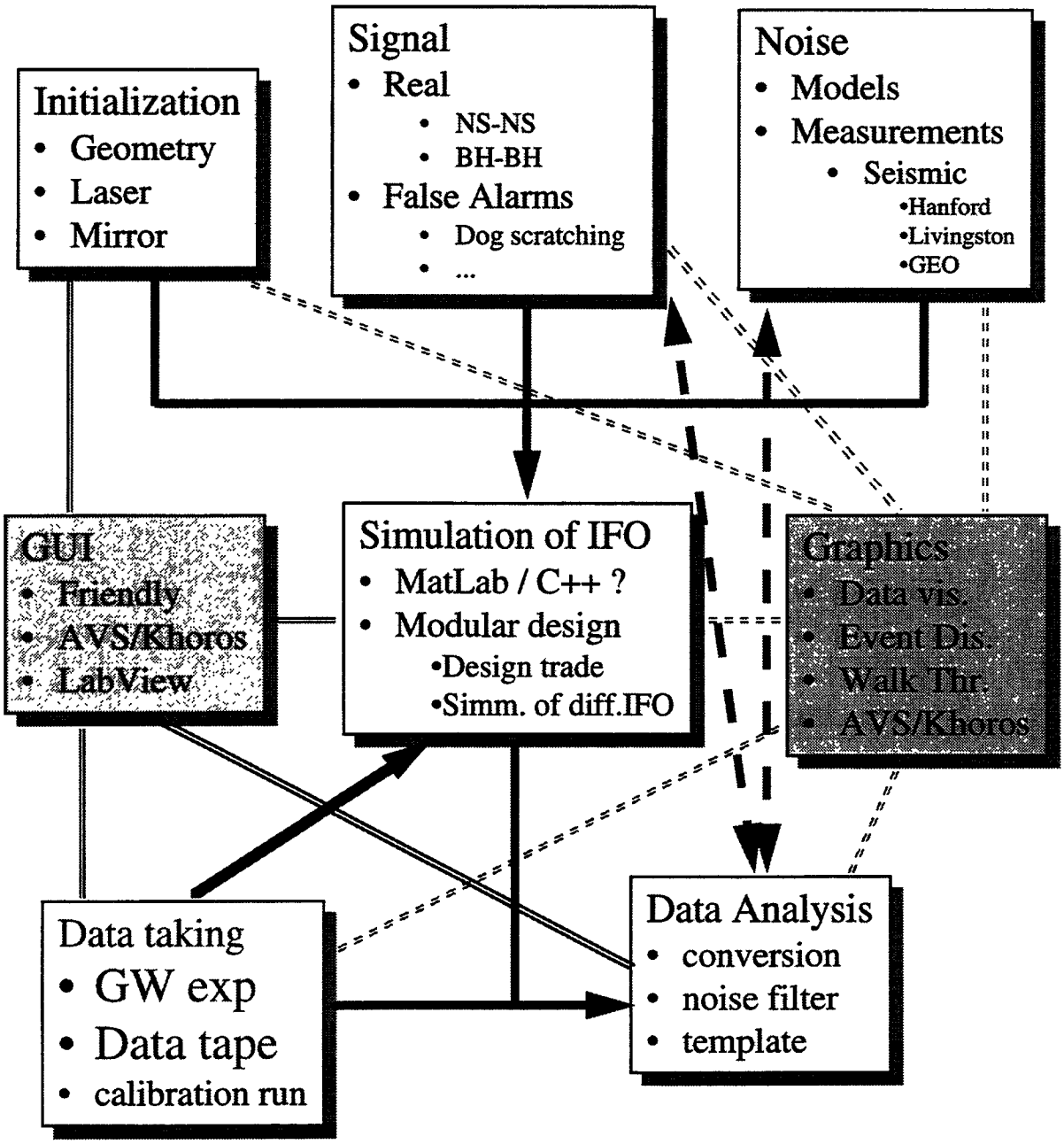
Why do we need - interexperiment collaboration

- Share data
  - GEO
    - GEO600 for Advanced LIGO detector R&D
  - VIRGO
    - Combine data
  - TAMA
- Common simulation tools
- Interchangeable data format / conversion modules
- Possibility to provide software service when LIGO becomes public observatory

LIGO Model - Evolution



LIGO Model - Structure



What do we need ?

- Internal (LIGO) support/acceptance
- Powerful and flexible
- User friendliness
- Easy to maintain
- Easy to integrate works of different flavors (c/Fortran/MatLab/...)
- Easy to try different models
- Easy to integrate data and models

• Options considered

Khoros,AVS,Grid,SIESTA,LabView/CVI,EPICS

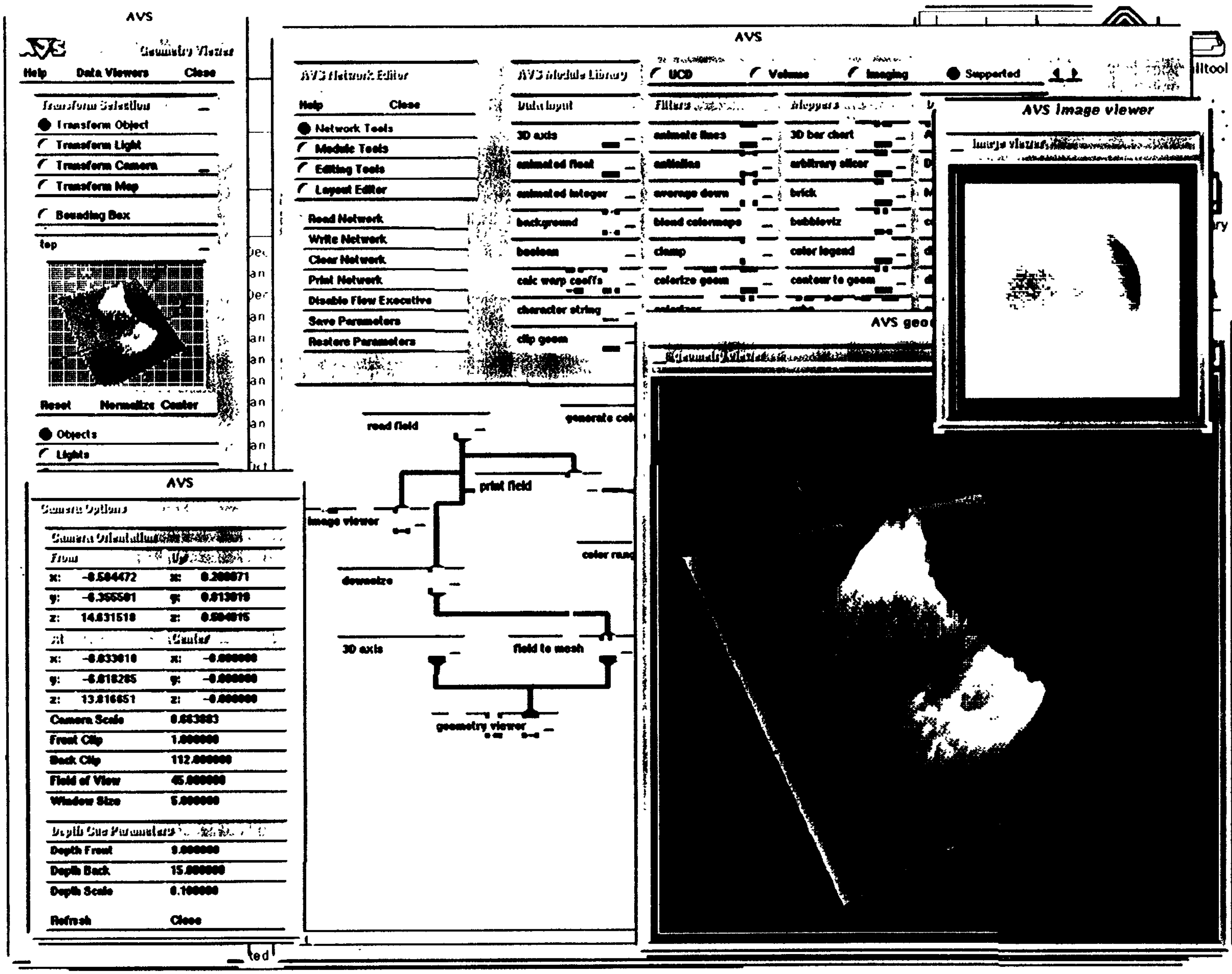
- Modular vs Monolithic
  - Part of SIESTA(VIRGO) and Grid(GEO) will be included as modules
- Powerful enough modules
- Module can be written by c/Fortran
  - Automated Make, GUI and Man
- Graphical Programming Interface
- Good graphics viewer
- Shared memory, Distributed processing
- User recommendation

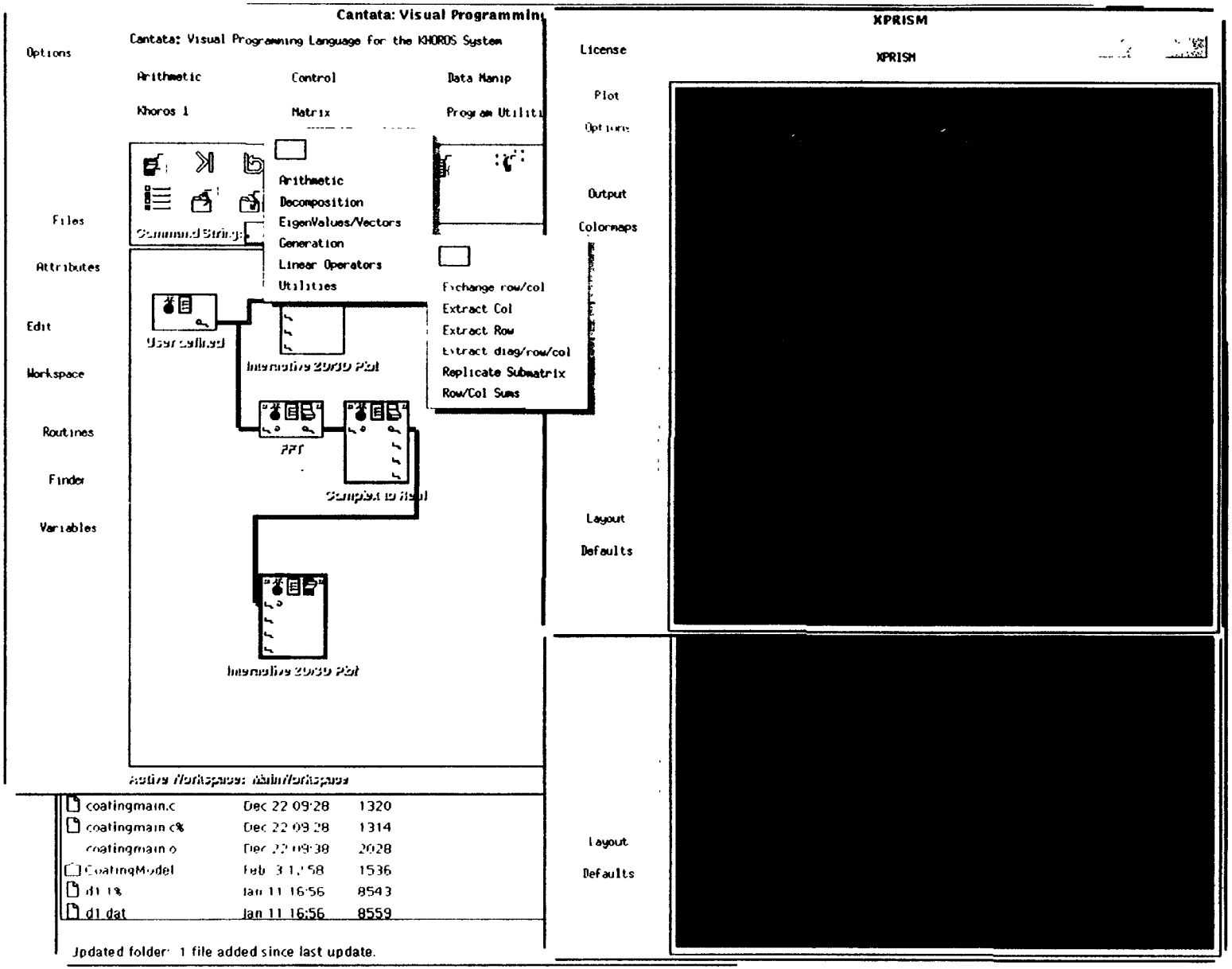
• Not included in considerations

- All possible case study
- Free vs Commercial
- Now vs future
  - OLE/COM and OpenDoc/SOM

People Interviewed

- R. Frost (SDSC) - Khoros advocate
  - Said that the architecture we wanted to build was ideal for Khoros.
- M. Lee (JPL) - old Khoros user
  - Khoros v1 is too slow (v2 will address those), AVS is good but expensive. JPL built custom in house system, not very flexible - script based.
- JPL Space Project Development Group
  - They used MatLab for optics modeling, and believe using a coarse network environment like Khoros / AVS would compromise speed and limit functions.
- J. Fox (Syracuse Univ) - AVS advocate
  - NPAC did a trade study and opted for AVS.
  - Technology of AVS is first class - module networks, support, reliability...
  - Future of Khoros / AVS like systems
    - Current architecture used in both Khoros and AVS will be obsolete in 5 years. To have our effort useful in the future, we must write good modular code.
- B. F. Schutz (Cardiff) - Grid originator
  - The developer of Grid became nervous breakdown.
  - Grid is similar in many respects to Khoros/AVS
  - Cardiff uses Glid only for analysis end of modeling.


AVS
Geometry Viewer
Help
Data Viewers
Close
Transform Selection
Transform Object
Transform Light
Transform Camera
Transform Map
Bounding Box
top
Reset
Normalize
Center
Objects
Lights
AVS
AVS Network Editor
Help
Close
Network Tools
Module Tools
Editing Tools
Layout Editor
Read Network
Write Network
Clear Network
Print Network
Disable Flow Executive
Save Parameters
Restore Parameters
AVS Module Library
UCD
Volume
Imaging
Supported
Data Input
Filters
Mappers
3D axis
animated float
animated integer
background
boolean
calc warp coeffs
character string
clip geom
animate lines
antialias
average down
blend colormaps
clamp
colorize geom
3D bar chart
arbitrary slicer
brick
bubbleviz
color legend
contour to geom
AVS Image viewer
read field
print field
image viewer
downsize
3D axis
field to mesh
geometry viewer
AVS
Camera Options
Camera Orientation
From
x: -8.584472
y: -6.356581
z: 14.631518
Camera Scale 0.683883
Front Clip 1.000000
Back Clip 112.000000
Field of View 45.000000
Window Size 5.000000
Depth Cue Parameters
Depth Front 9.000000
Depth Back 15.000000
Depth Scale 0.100000
Refresh
Close


Cantata: Visual Programming Language for the KHOROS System
Options
Arithmetic
Control
Data Manip
Khoros 1
Matrix
Program Utiliti
Files
Attributes
Edit
Workspace
Routines
Finder
Variables
Arithmetic
Decomposition
EigenValues/Vectors
Generation
Linear Operators
Utilities
Exchange row/col
Extract Col
Extract Row
Extract diag/row/col
Replicate Submatrix
Row/Col Sums
XPRISM
License
XPRISM
Plot
Output
Colormaps
Layout
Defaults
coatingmain.c Dec 22 09:28 1320
coatingmain.c% Dec 22 09:28 1314
coatingmain.o Dec 22 09:38 2028
CoatingModel Feb 3 1:58 1536
d1.1% Jan 11 16:56 8543
d1.dat Jan 11 16:56 8559
Updated folder: 1 file added since last update.

AVS vs Khoros

	AVS 5	Khoros 2.01
Product General		
Publisher	Advanced Visual Systems, Inc	Khoral Research, Inc Univ. of New Mexico initiated the project.
Price	$10,000 ~ $25,000 / user	free when used internally and the distribution is non-profit purpose. Funded by licencing fee for value-added commerical distribution and by the Consortium membership.
Stability	good Now	unstable, Now, may take 0.5-1 year It is very difficult to support many platforms by distributing soruce code and by limited resources
Support method	* Direct e-mail and phone to AVS, Inc * International AVS Center (IAC), a part of the North Carolina Supercomputing Program of MCNC * ask for help in newsgroup	* e-mail to KRI * ask for help in newsgroup
Time to get answer	1-2 days	1-? days, not guaranteed
Time to get bug fix /workaround	engineer will start asap (at least they say so)	not predictable (easy fix may be done immediately, hard ones ?)
Users	Wide range, Seismic, Finite Element, Quantum Mechanics... , professional graphics image handling, tend to be rich users	Much wider than AVS, tend to be poor users
Free modules by end users	Available (ftp to IAC)	Available (ftp to various sites, inc unm)
Distribution method	compiled program	source code (nice interactive make included)
Support platform	major UNIX workstations, VMS, Cray	major UNIX workstations (inc. 486), Cray

	AVS 5	Khoros 2.01
Program Architecture		
Program design	Extendable toolbox in low level and iconic network structure in high level	Same as AVS
Module / Glyph	Run under AVS environment	Gryph can run as a stand alone executable.
Control structure in network editor	doable using special modules	Supported
Modules provided	Rich, mature no good xvgr-like ? (gnuplot/gnufit based module)	Provides essential minimum for our use, but some are premature and/or limited
3D viewer and modeling	Good 3D viewer and building tools of 3D object	Primitive 3D viwewer
Conv. c/Fortran code to module	easy make utility available	easy make utility available
man / help	easy to prepare	easy to prepare
Data transfer	via disk file or memory	via disk file or memory
Data structure	standard + Unstructured Cell Data Data conversion => modules, simple header	standard + Data Service Mgr - flexible but slow
Distributed Computing	supported Now	planned in 2.1 (2nd Qtr,95), but not yet in 2.0.1.
GUI	Pretty.	Essentially providing the same as AVS, but aethetically not as pretty as AVS
Command Line Interpreter	Extensive control of AVS	To drive each modules
Platform dependence	Utilize the best library for each system and support 3rd party hardware	Platform dependence is minimied
Language	C++	C/Fortran, because of variations of C++ compiler. Files are organized so that the development can mimic (to some extent) object-oriented way.
Stand alone application	With the developer option, one can make a stand alone application with unique GUI by linking AVS Library.	One can make a stand alone application with unique GUI, which runs under an interpreter.