Implementing a CFD steering system for immersive environments

Kai-Mikael Jää-Aro

Department of Numerical Analysis and
Computer Science
Royal Institute of Technology

Contents

- Background – the VIRTUALFIRES project
- Current implementation
- Conclusions for the future

VIRTUALFIRES partners

- http://www.virtualfires.org/
  - SiTu, TU Graz, Austria
  - CD, Uni Leoben, Austria
  - PDC, KTH, Sweden
  - FIGD, FhG, Germany
  - EUVE, Spain
  - FDDo, Germany
  - LTF, France
  - CETU, France

VIRTUALFIRES – the aims

- Real-time numerical simulation of tunnel fires
- Real-time steering of simulation parameters
- Immersive visualisation
- Safety studies of future tunnels
- Scenario training for fire fighters
- (Support partners' pet projects)

Desired interface

- Users should be able to set boundary conditions interactively in the displayed tunnel geometry
- Users should be able to interactively place fire loads, fire-fighting equipment and other items in the tunnel
- Users should be able to indicate events happening at future times in a scenario (ventilation turned on/off, fire extinguishers used, etc)

Further desirables

- Multiplatform – PC to supercomputer
- Ideally use identical user interfaces in CAVE, in HMD and on desktop
- Support non-expert users
- Easily extensible interface

Choices made

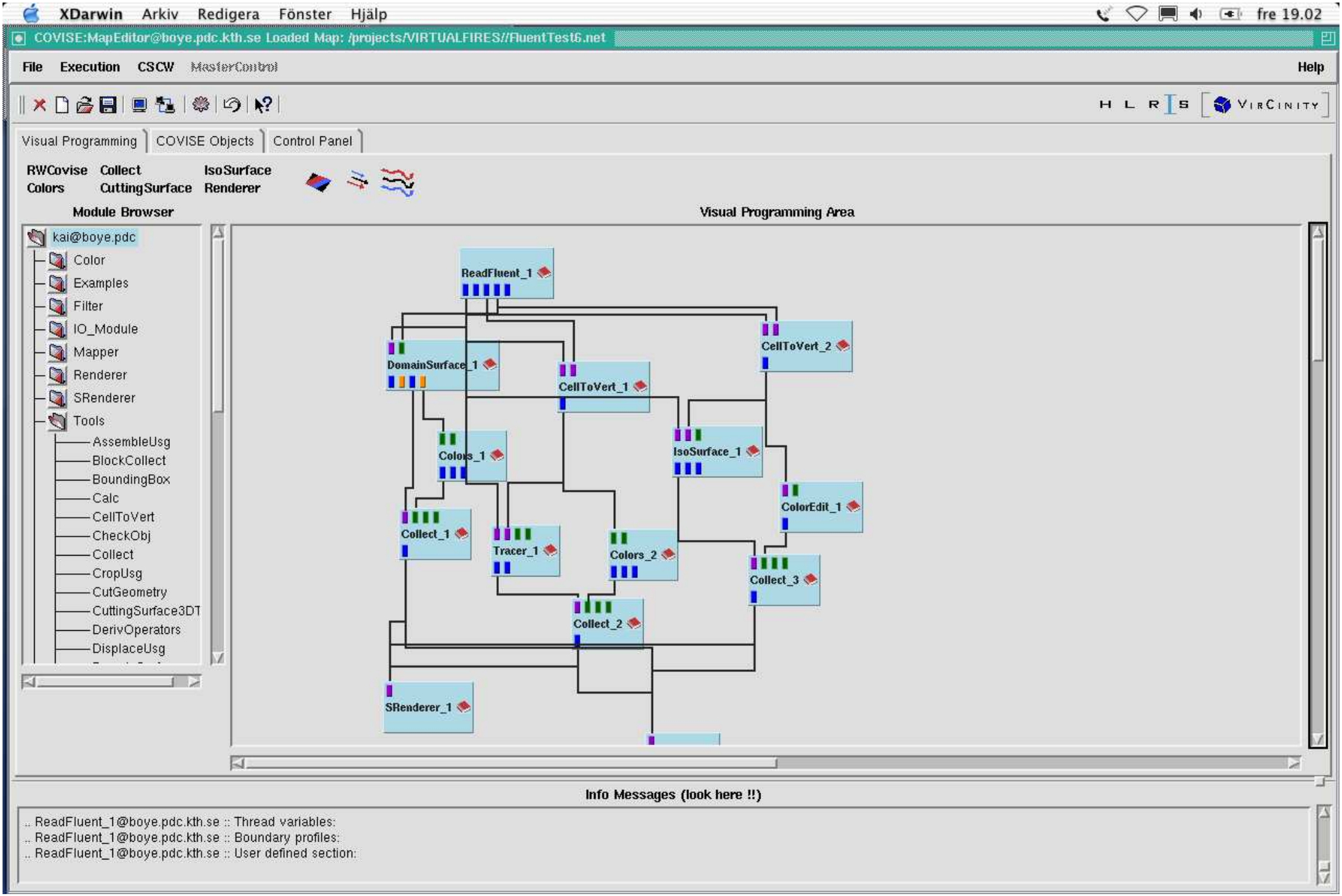
- Small-size GUI
  - Displayable on PDA for CAVE version
  - Fits in HMD view
  - Can be used on desktop screen
- Use COVISE as visualisation platform

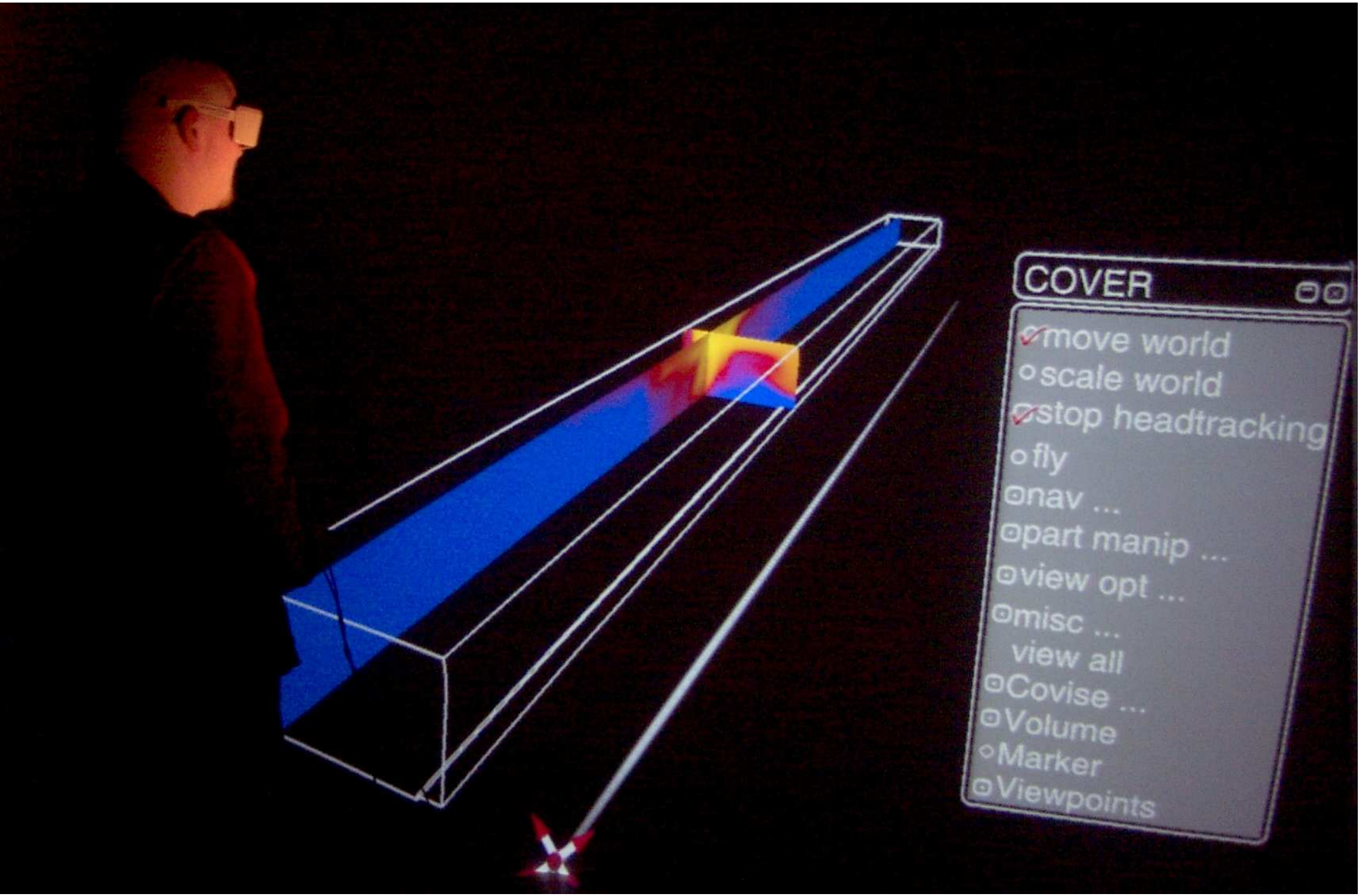
COVISE

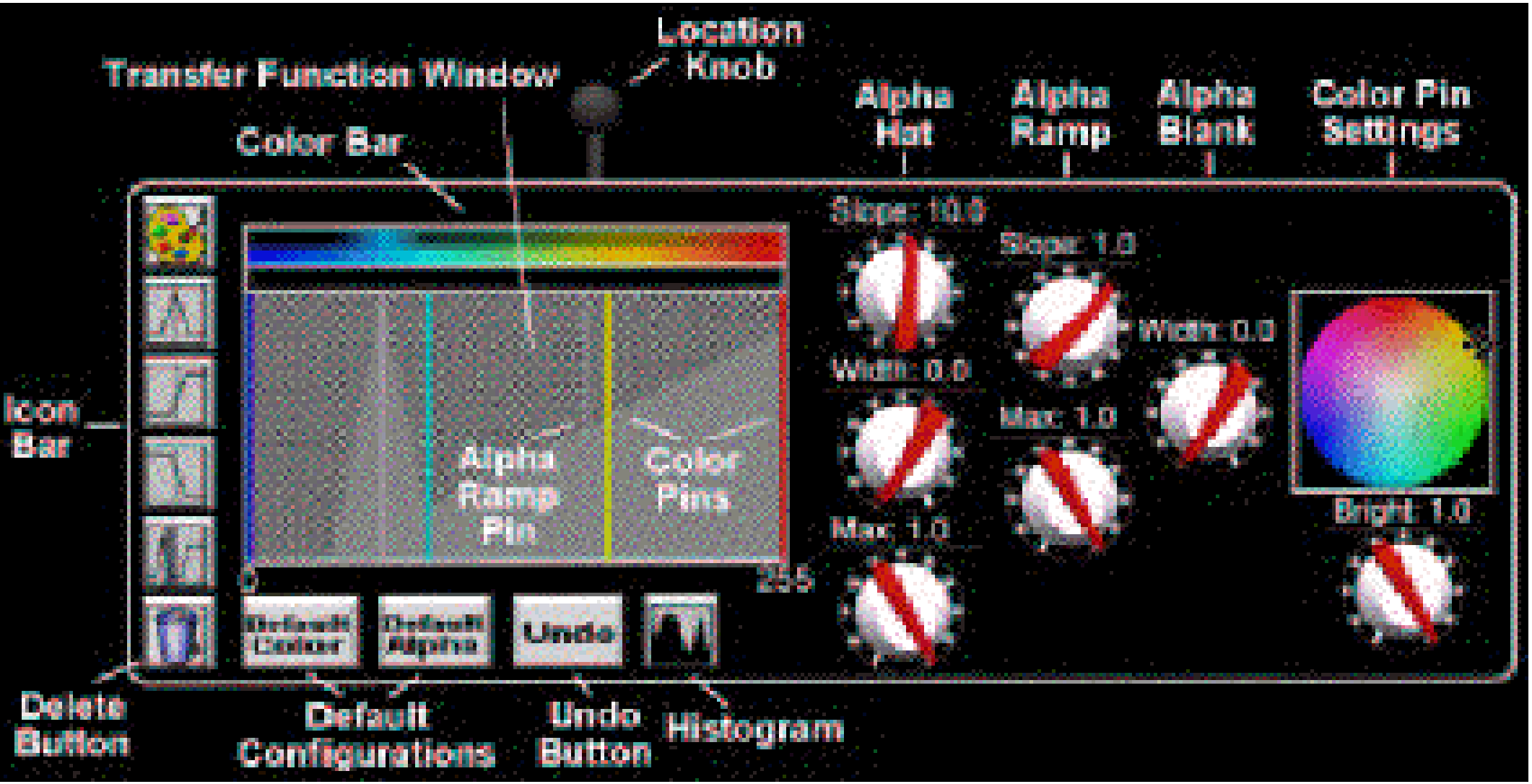
- `http://www.vircinity.de/`
- Modular visualisation system
- Graphical programming language
- Distributed system
  - Modules on different machines
  - Support for remote collaboration
- Immersive rendering module – COVER
- User-extensible

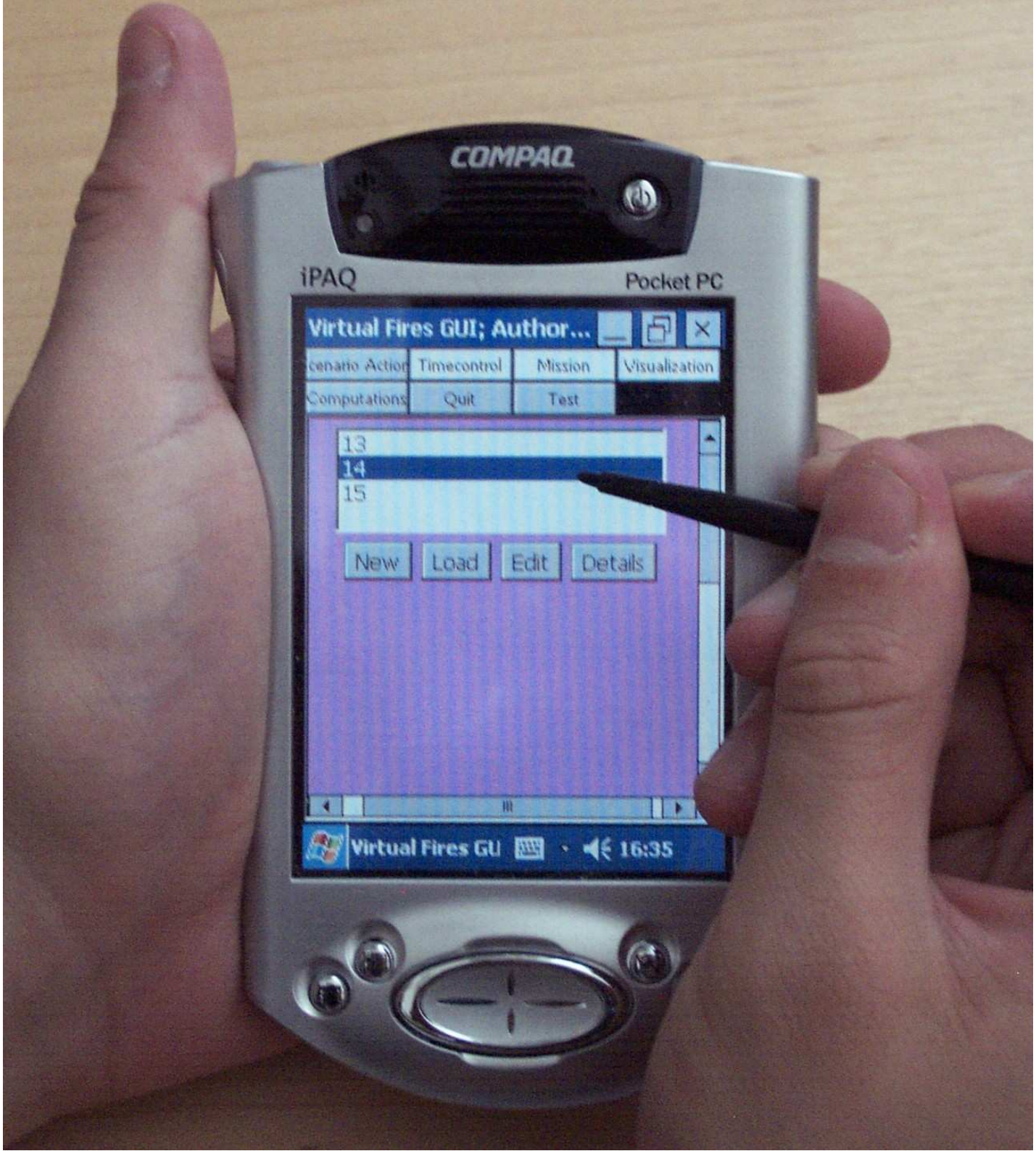
COVISE programming model

- Fairly strict dataflow, but modules can attach messages to data
- Plugins in COVER can intercept these messages and send responses to the originator
- Plugins can send messages to each other
- Plugins can access the scene graph

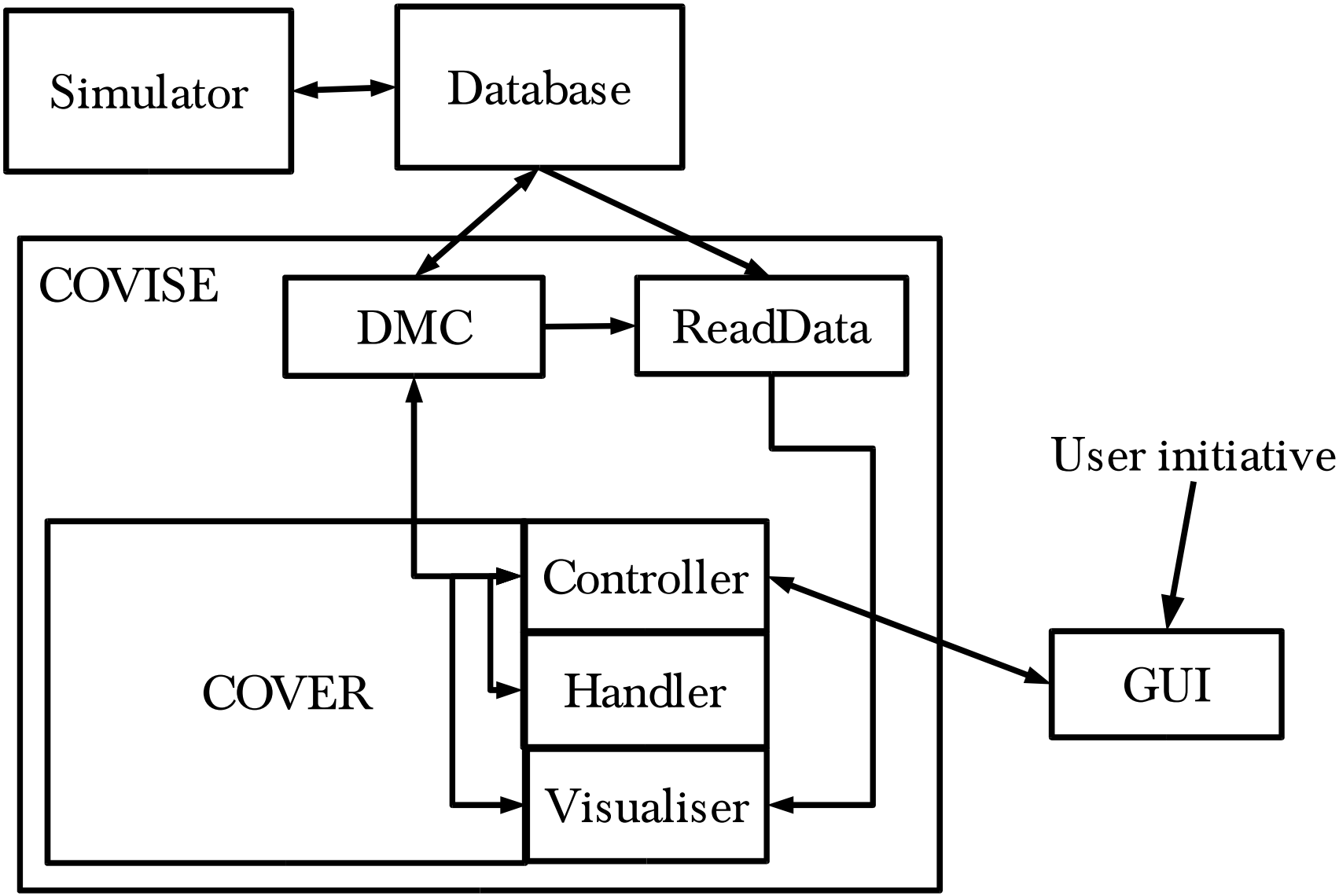

XDarwin Arkiv Redigera Fönster Hjälp
fre 19.02
COVISE:MapEditor@boye.pdc.kth.se Loaded Map: /projects/VIRTUALFIRES//FluentTest6.net
File Execution CSCW MasterControl Help
Visual Programming
COVISE Objects
Control Panel
RWCovise Collect IsoSurface
Colors CuttingSurface Renderer
Module Browser
Visual Programming Area
kai@boye.pdc
Color
Examples
Filter
IO_Module
Mapper
Renderer
SRenderer
Tools
AssembleUsg
BlockCollect
BoundingBox
Calc
CellToVert
CheckObj
Collect
CropUsg
CutGeometry
CuttingSurface3DT
DerivOperators
DisplaceUsg
ReadFluent_1
DomainSurface_1
CellToVert_1
CellToVert_2
Colors_1
IsoSurface_1
ColorEdit_1
Collect_1
Tracer_1
Colors_2
Collect_3
Collect_2
SRenderer_1
Info Messages (look here !!)
.. ReadFluent_1@boye.pdc.kth.se :: Thread variables:
.. ReadFluent_1@boye.pdc.kth.se :: Boundary profiles:
.. ReadFluent_1@boye.pdc.kth.se :: User defined section:


COVER
move world
scale world
stop headtracking
fly
nav ...
part manip ...
view opt ...
misc ...
view all
Covise ...
Volume
Marker
Viewpoints


Transfer Function Window
Location Knob
Color Bar
Alpha Hat
Alpha Ramp
Alpha Blank
Color Pin Settings
Slope: 10.0
Slope: 1.0
Width: 0.0
Width: 0.0
Max: 1.0
Max: 1.0
Bright: 1.0
Icon Bar
Alpha Ramp Pin
Color Pins
0
255
Undo
Delete Button
Default Configurations
Undo Button
Histogram


COMPAQ
iPAQ
Pocket PC
Virtual Fires GUI; Author...
Timecontrol
Mission
Visualization
Computations
Quit
Test
13
14
15
New
Load
Edit
Details
Virtual Fires GU
16:35

Current system design



Experiences

- Using COVISE did not work out very well
  - Most of the programming model had to be dropped
  - Much functionality had to be reimplemented
- Yet, COVISE is not a bad system
  - What happened?

Analysis

- We were locked in by an application framework. COVISE expects data to come in at the top, be processed and sent to a renderer.
  We want to use graphics as *input* to the system as well as output from it.
  - COVISE *did* allow us to implement extensions
  - But, very little of the functionality of COVISE is left

COVISE-specific problems

- The behaviour of COVISE modules is set through module *parameters*
  - Parameters can be interactively set by the user, by editing values in a popup window and/or by adding parameters to a separate *Control panel* window
  - A module can, through an explicit message, allow a COVER plugin to modify its parameter values immersively, but only a few modules send this message, and the rest cannot be modified, only reimplemented.
- System in principle allows customisation, but does not fully support it in practice.

- In our application many operations have to communicate with each other to allow interaction

  $\Rightarrow$ use plugins instead of modules

  But, plugins are invisible in the programming interface

  $\Rightarrow$ the logic of the program is hidden in parallel code

  $\Rightarrow$ programs are not interpreted but have to be compiled

COVISE-specific remedies

- Input ports instead of “hidden” parameters give more flexibility when programming
- Sub-classable modules diminishes need for reimplementation
- Input ports made available for plugins

- Plugins should not be hidden in the visual program, they need to be made public, showing their connections to other program elements

Better yet

- We should not be constrained by a framework
- Yet we want to use as much existing functionality as possible from a visualisation system
  - We should be able to link in visualisation functions in our program without prejudice
  - Caveat: This is (currently) difficult to do in a purely graphical programming system

The case for open source

- There are very few general visualisation tools for immersive environments:
  AVS Express/MPE, COVISE, vGeo
  - It's non-trivial to test commercial systems and get them to work
  - They are difficult to extend beyond their framework
- We need an open source platform for immersive visualisation

Alternatives

- Currently there are two major open visualisation packages: VTK and Open DX
- Both can be linked into external programs
- VTK has already been used for immersive visualisation through VtkActorToPF module, but is not as easy to program
- Open DX has graphical user interface, but no immersive renderer/interaction module

My suggestion

- I would prefer an immersive extension to Open DX, as it is easier for non-experts to program and has more complete functionality