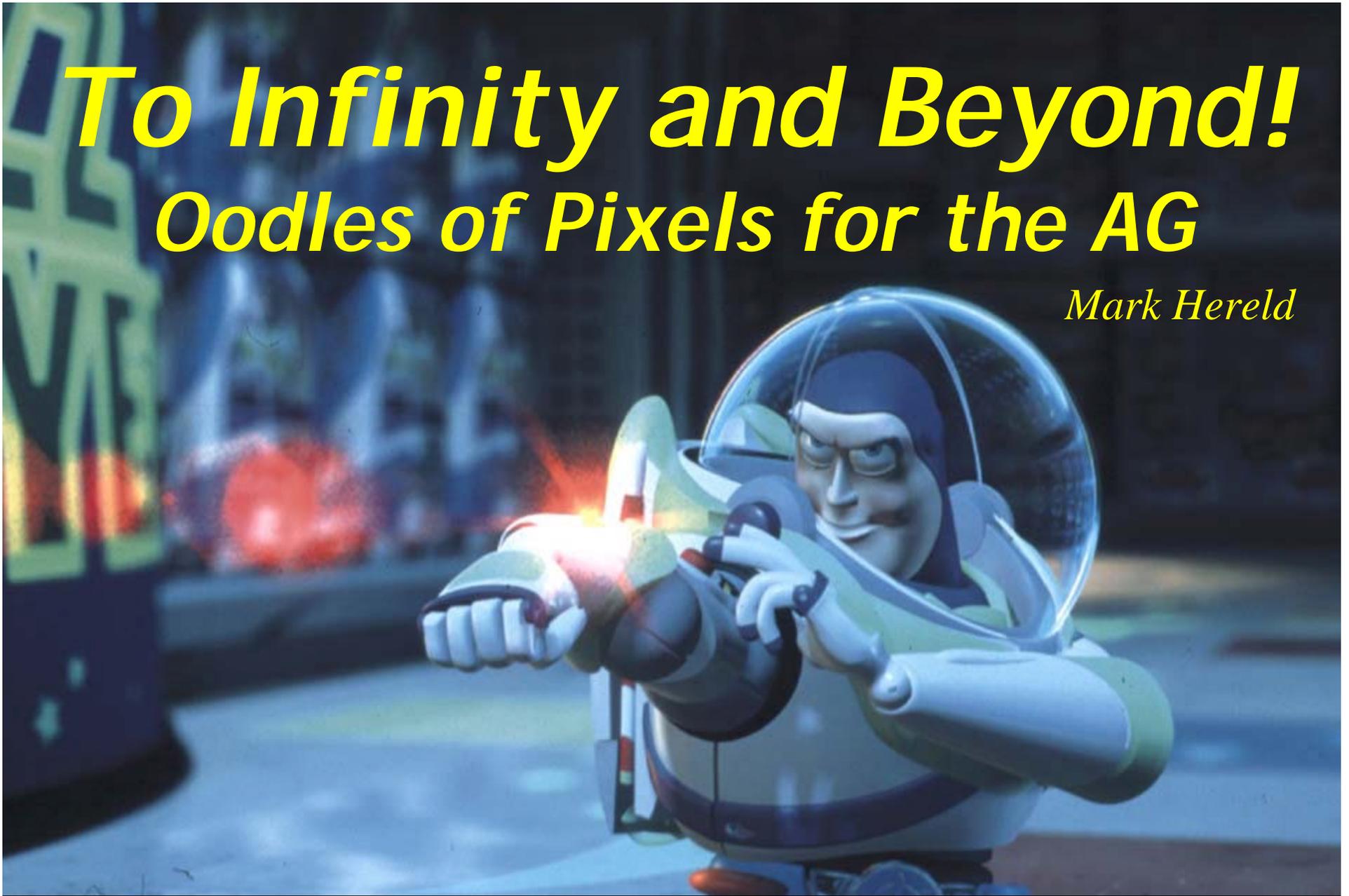
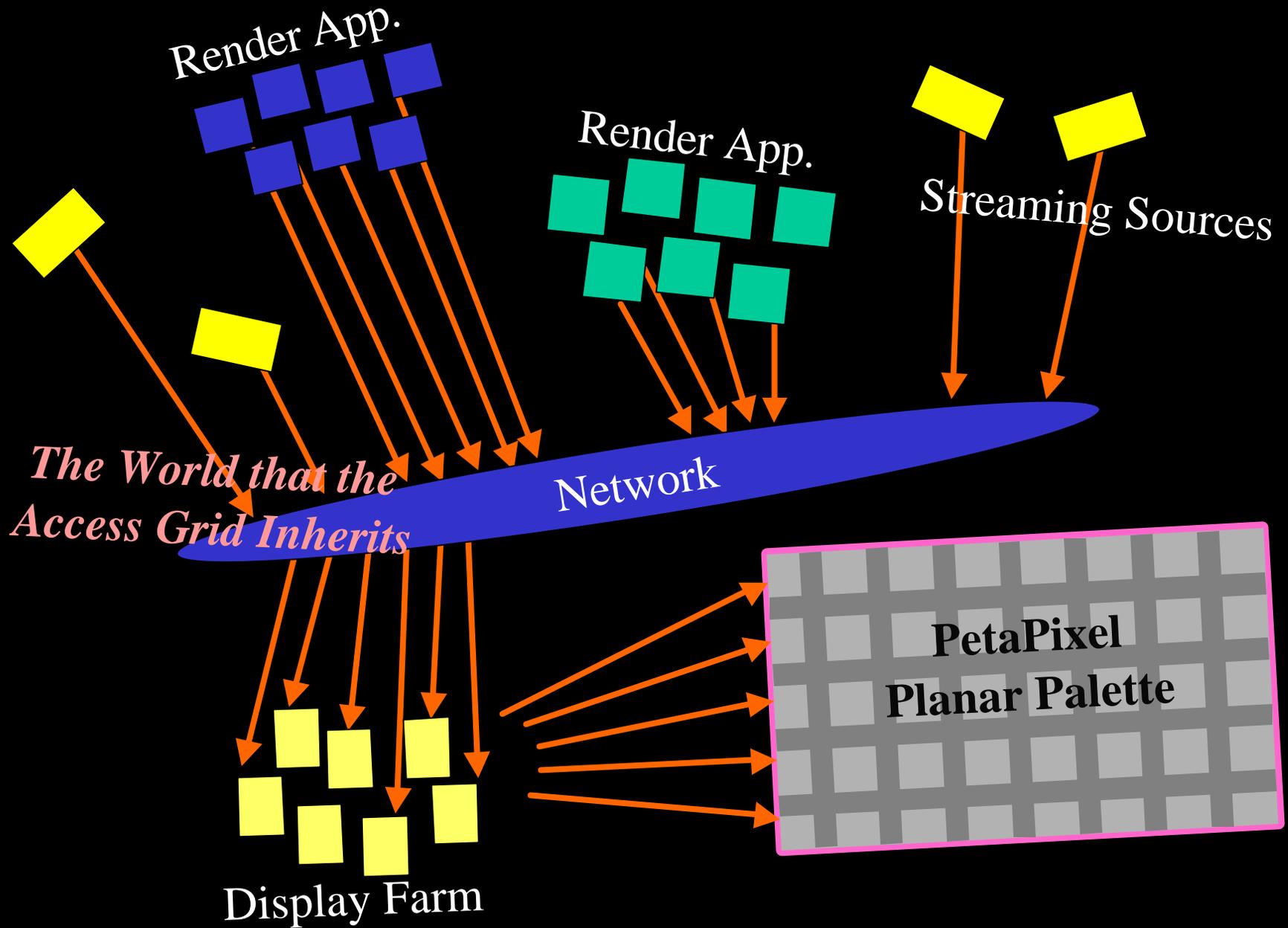


# *To Infinity and Beyond! Oodles of Pixels for the AG*

*Mark Hereld*



# SCALABLE STUFF



# Why Tiled Displays?

- Scalable road to many many pixels, when fed by display farm
  - Windows pixel-based model doesn't scale
  - Shared memory models don't scale (and are expensive!)
- Hi-res video promotes presence
  - Detailed, life-size & compelling portrayals of collaborators, their environment, and their work.
- Deep and rich scientific visualizations
  - Driven to higher resolution by dataset size, simulation detail
    - State-of-Art: applications with 5 to 15 Mpix
    - Horizon: 100 Mpix displays
    - Compare: 3-projector AG node provides 2.3 Mpix
- Hi-res streaming movies
- Scaling AG to larger spaces
  - Floor-to-ceiling wall-to-wall active display space
  - Very large venues
- (Why not?)

# *Objects in Mirror **Appear Closer***

*Now*

*Then*

*That was the  
Best Birthday Ever!  
Thanks!!!*

# Oodles of Pixels (on the cheap)

- Tiled display systems scale better (in principle)
  - *Pixels*
    - aggregating display devices >> larger monolithic devices
  - *Communications*
    - networks >> busses
  - *Rendering power*
    - display farms >> shared memory driving graphics pipes

# Some Related Work

- parallel rendering - ICASE and UNC
- X
  - 3-D Graphics (GLX), Tiling (Xinerama), Direct Rendering Infrastructure (DRI), X Protocol Multiplexors (XMX, X<sup>\*</sup>-X, SDWS, SLS/d)
- WireGL (Stanford)
  - OpenGL-to-many (output parallel, tile-aware, efficient)
- Scalable Wall Project - Princeton
  - Sorting algorithms
- Hardware-based
  - DEXON, Panaram, Metabuffer (U.T. Austin)

# Major Challenges

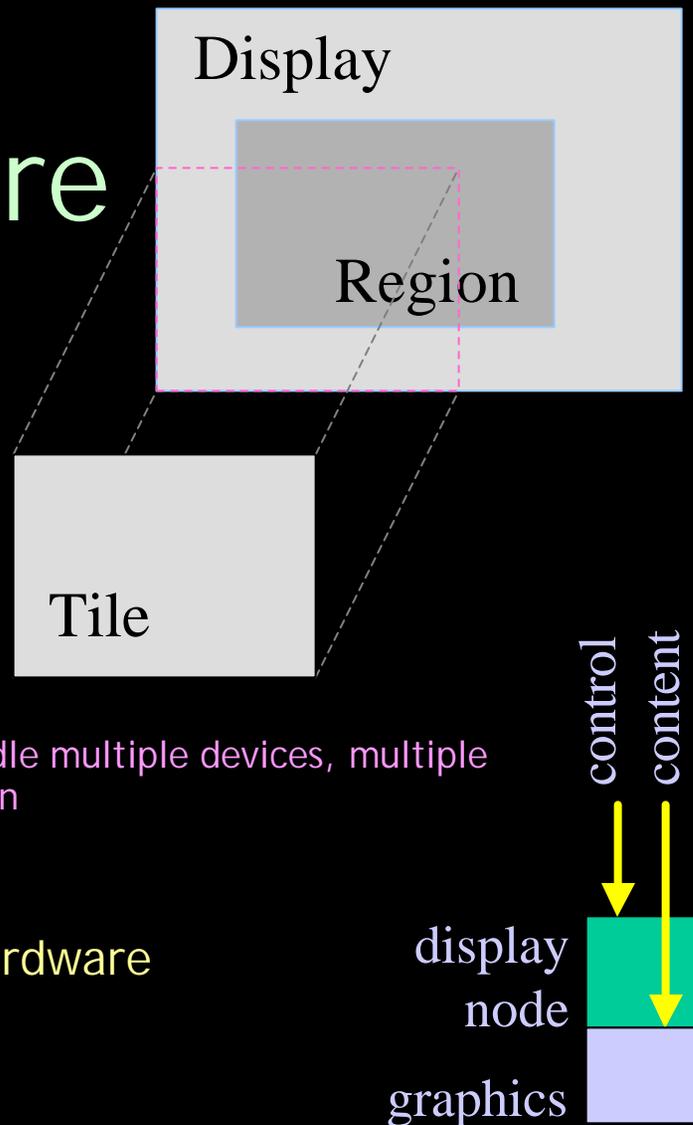


- Distributed management
  - data distribution
  - state synchronization
- Network transport
  - extracting acceleration from networked rendering hardware
  - minimizing protocol overhead
  - compression (efficiency and overhead)
  - efficient data representations
- Decomposing the rendering problem
  - image space, object space, anywhere in between (i.e. sort-first, sort-last, sort-middle)

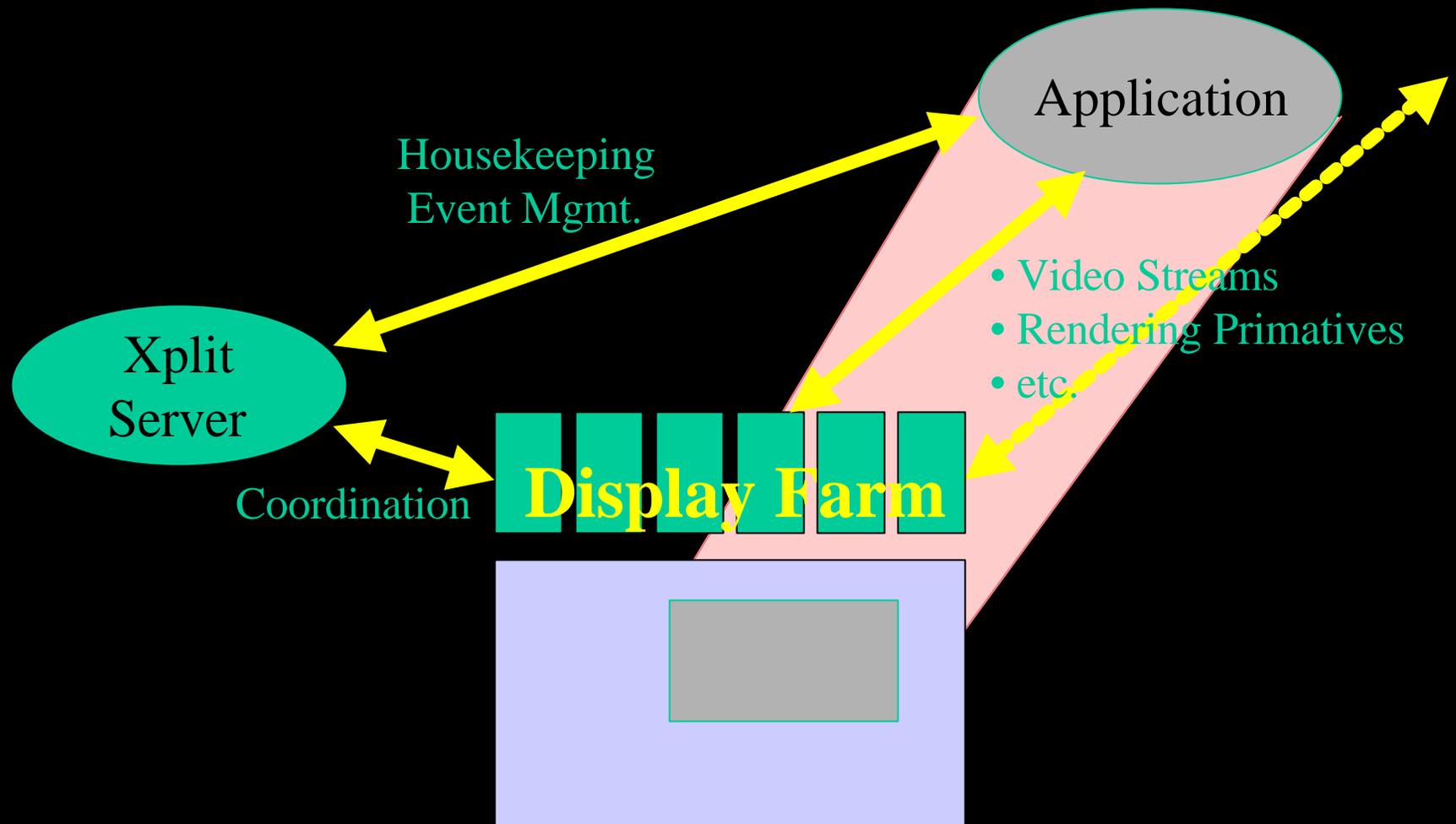
100 MB/sec raw to a Tile  
1024 x 768 pix, RGBA bytes, 30 fps

# The Xplit Architecture

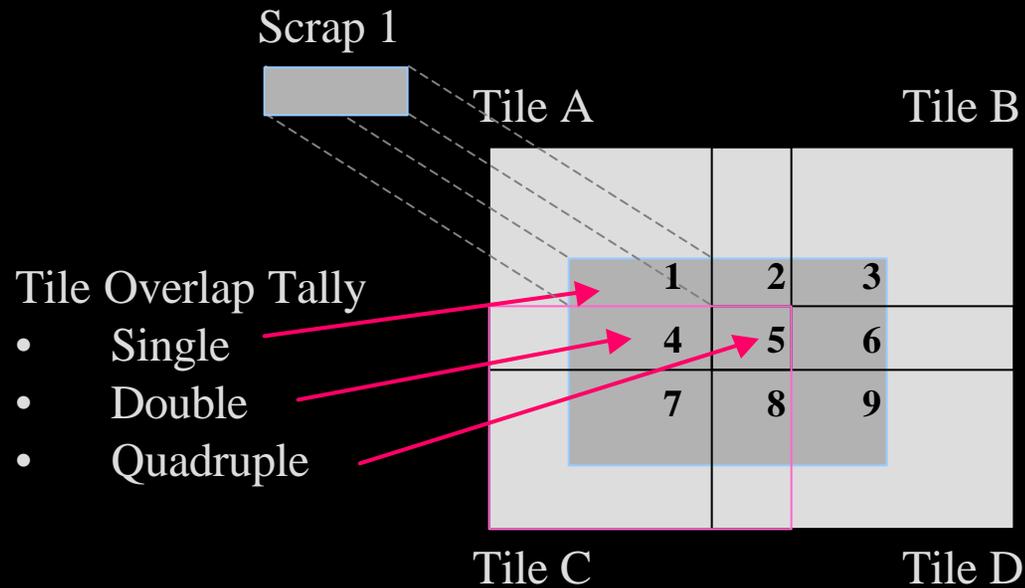
- Region as First Class Object
  - behaves as a virtual display
  - windowlike properties (layerlike?)
- Virtual display server
  - display region allocation/coordination
    - Account for overlap regions
  - key/mouse event handling
    - Opportunity to extend current models to handle multiple devices, multiple users, brokered floor control and collaboration
  - desktop manager is just another client
    - besides, is it really a desktop anyway???
- ★ Binding over-the-wire content to display hardware
  - hardware acceleration in the display farm
  - servelets in display farm nodes
  - extensible
    - new content formats supported as drivers + servelets become available



# The Application's View

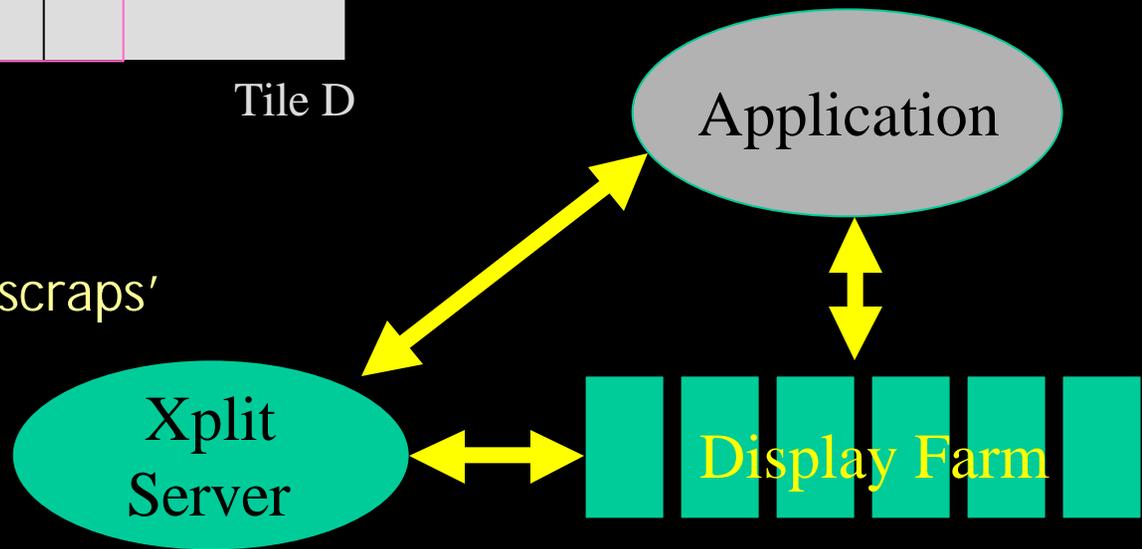


# Region Baggage



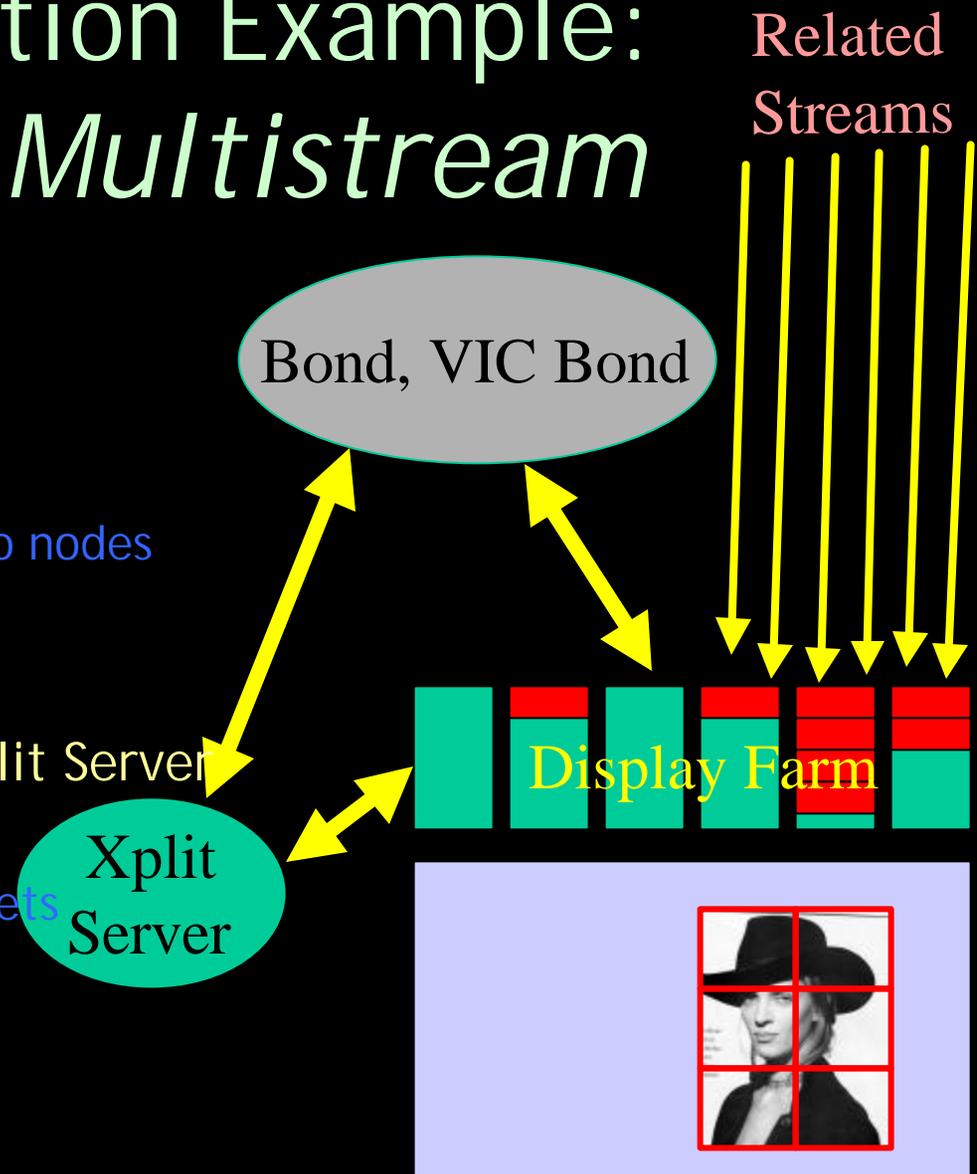
My Region	
Scrap 1	Geom
Tile A	ID
Scrap 2	Geom
Tile A	ID
Tile B	ID
Scrap 2	...

- Express minimum decomposition via 'scraps'



# A VICish Application Example: *Bonded H.261 Multistream*

- VIC Bond knows the streams
- Region from Xplit Server
  - includes scrap-list bindings to nodes
- Spawn H.261 servelets (■)
  - stream-to-node(s)
- VIC Bond gets events from Xplit Server
  - gets adjusted scrap-list
  - makes adjustments to servelets



# Xplit Feature List

- Graphics acceleration
- Unbiased support of transports
- Scalable
- Commodity networks, servers, rendering hardware
- Supports parallel applications
- Immediate-mode rendering
- Windowing (Layers?)
- Open source
- Supports remote clients (X)
- Transparent API (yeah, right!)

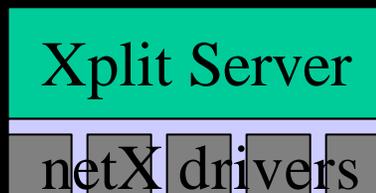


Retreat 2001

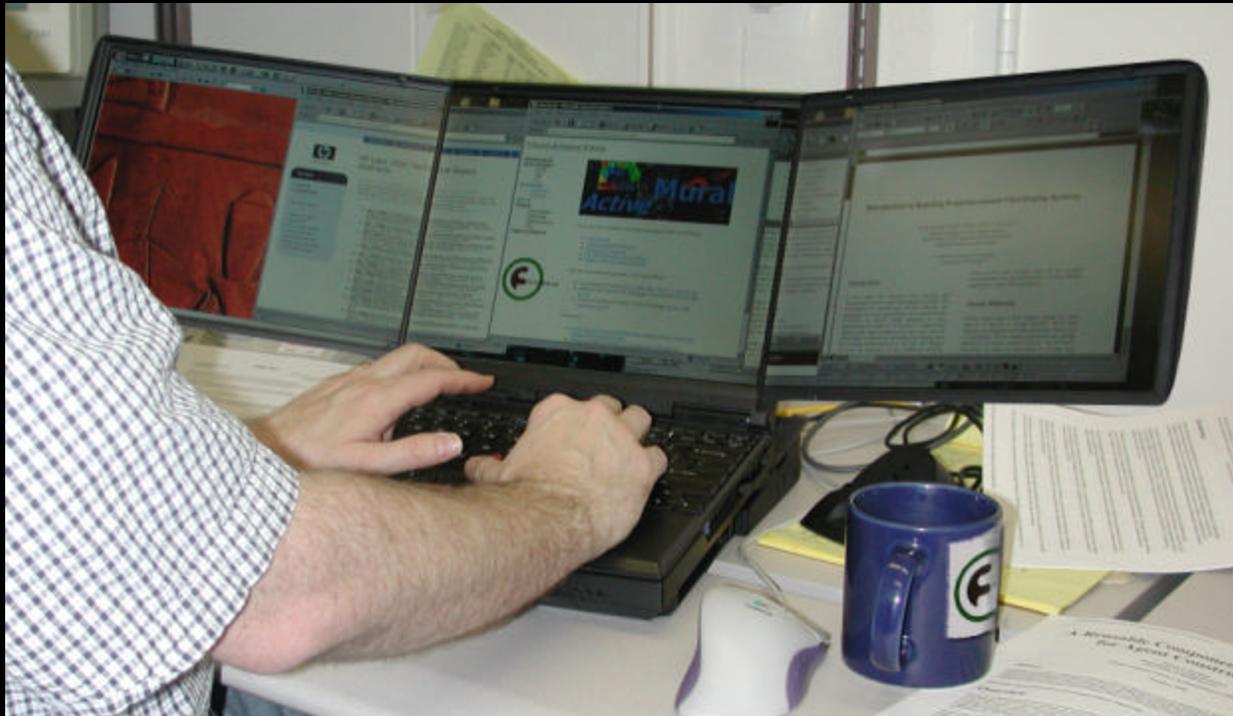
*Futures Laboratory, Argonne National Laboratory*

# Constructing Xplit for the AG

- Virtual server
  - netX (net display device)
  - kbd/mouse
  - window manager
- Application client interface
  - Region handling
  - Update/event handling
- Boot up w/central functions
  - H.261 servelet
- Boot up some example apps
  - VIC Bond
  - OpenGL (via WireGL?)
  - Hi-res vis movie
- Release Xplit v1.0
  - "the Access Grid cut"
- Extend to parallel apps
  - Client interface
    - Synchronization
    - Sorting
    - Load balancing



# ThreeTop *AG for the rest of us? :)*



- **Portable**  
less than 1" thick  
6 mighty mighty pounds
- **Configurable**  
3x1 (shown), 2x1 and 1x  
tablet w/touch  
3x2 versions coming soon!
- **Inexpensive**  
\$22,500 Limited Edition  
\$6,000 by 2004

# References

- "A Survey of X Protocol Multiplexors", John Eric Baldeschwieler, Thomas Gutekunst, Bernhard Plattner, *ACM Computer Communication Review*, Vol. 23, No. 2, pp. 13-22, 1993.
- "Design Considerations for Parallel Graphics Libraries", Thomas W. Crockett, *Proceedings of the Intel Supercomputer Users Group 1994*, San Diego, June 1994.
- "A Sorting Classification of Parallel Rendering", Steven Molnar, Michael Cox, David Ellsworth and Henry Fuchs, *IEEE Computer Graphics and Algorithms*, pp. 23-32, July 1994.
- "The Design of a Parallel Graphics Interface" Homan Igehy, Gordon Stoll and Pat Hanrahan, *Proceedings of SIGGRAPH 98*, pp. 141-150, 1998.
- "Load Balancing for Multi-Projector Rendering Systems", Rudrajit Samanta, Jiannan Zheng, Thomas Funkhouser, Kai Li and Jaswinder Pal Singh, *SIGGRAPH/Eurographics Workshop on Graphics Hardware*, August 1999.
- "Hibrid Sort-First and Sort-Last Parallel Rendering with a Cluster of PCs", Rudrajit Samanta, Thomas Funkhouser, Kai Li and Jaswinder Pal Singh, *SIGGRAPH/Eurographics Workshop on Graphics Hardware*, August 2000.
- "Distributed Rendering for Scalable Displays", Greg Humphreys, Ian Buck, Matthew Eldridge and Pat Hanrahan, *Proceedings of Supercomputing 2000*.
- "The Metabuffer: A Scalable Multiresolution Multidisplay 3-D Graphics System Using Commodity Rendering Engines", William Blanke, Chandrajit Bajaj, Donald Fussell and Xiaoyu Zhang, *Technical Report, Center for Computational Visualization*, University of Texas at Austin.



# <http://www.ReferenceNouveau.bib>

- OpenGL <http://www.opengl.org>
- GLX <http://www.sgi.com/software/opensource/glx>
- Utah-GLX <http://utah-glx.sourceforge.net>
- X <http://www.x.org>
- XMX <http://www.cs.brown.edu/software/xmx>
- WireGL <http://graphics.stanford.edu/software/wiregl>
- PGL <http://www.icas.edu/~tom/PGL>
- Linux <http://www.linux.org>
- Xinerama <http://www.linux.org/HOWTO/Xinerama-HOWTO.html>
- DRI <http://dri.sourceforge.net>
- DEXON <http://www.dexonsystems.com>
- Panoram <http://www.panoramtech.com>



Retreat 2001

*Futures Laboratory, Argonne National Laboratory*



**ACCESS GRID**  
Retreat 2001

*Futures Laboratory, Argonne National Laboratory*