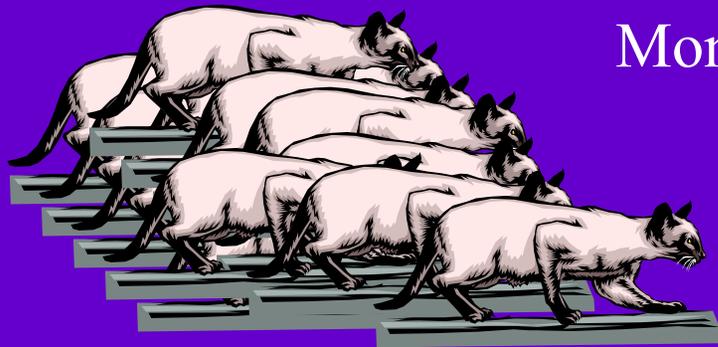


Components and Common Interfaces for Remote and Distributed Visualization

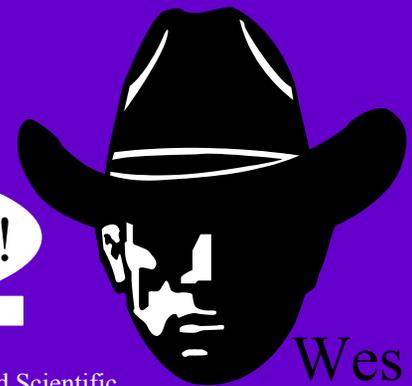
Jim Kohl

Oak Ridge National Laboratory

Panel: Visualization Frameworks, Toolsets, Interoperability
DOE Computer Graphics Forum
Monday, April 28, 2003



Heeee-hawwwwww...!



Research supported by the Mathematics, Information and Computational Sciences Office, Office of Advanced Scientific Computing Research, U.S. Department of Energy, under contract No. DE-AC05-00OR22725 with UT-Battelle, LLC.

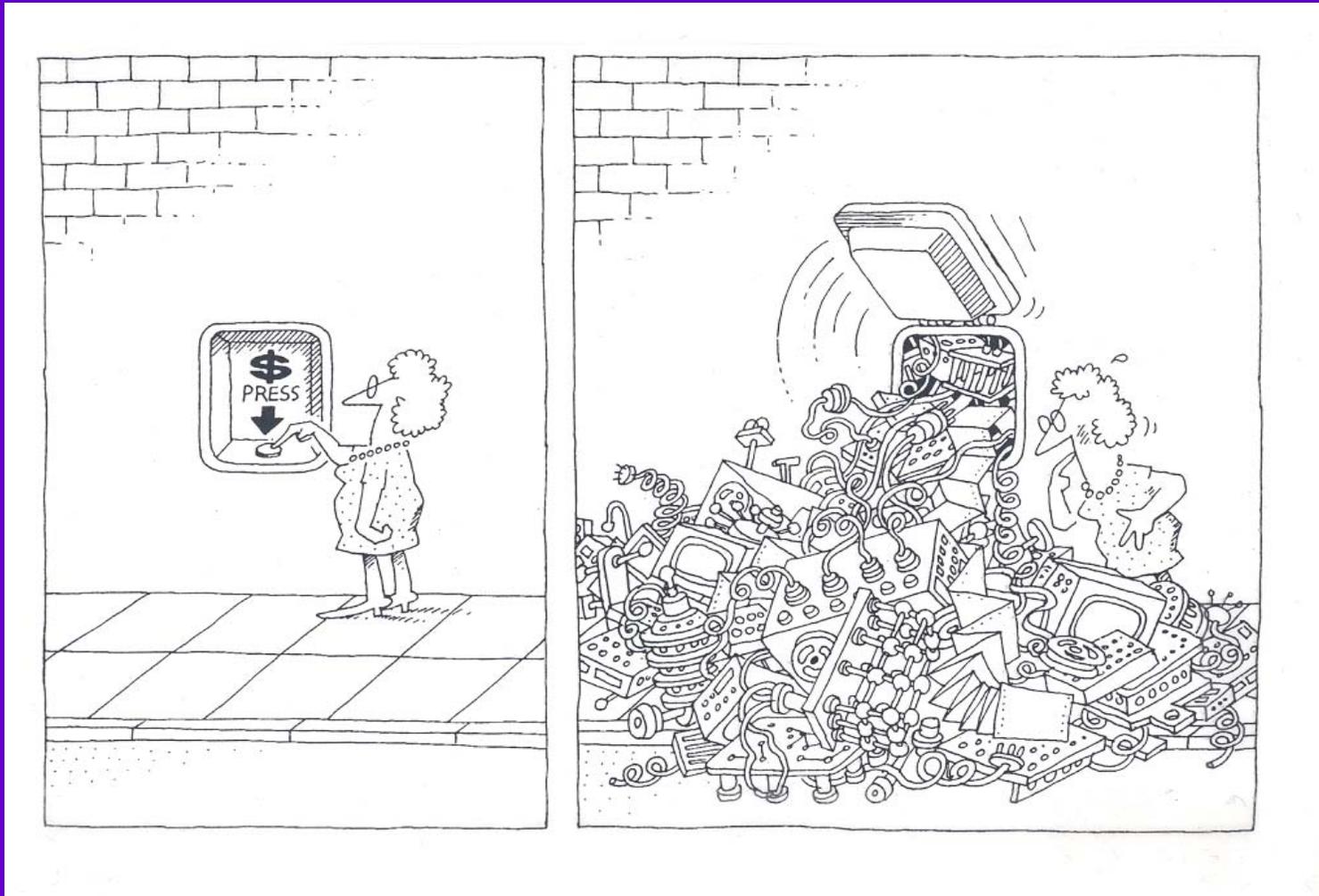
Remote & Distributed Viz (RDV)

- Big Data Over Wide Area Network...
 - ⇒ Performance, Bandwidth, Performance...
 - Data Reduction & Filtering (SDM)
 - ⇒ Where to Break the Pipeline?!
 - ⇒ Money Talks...
- Need a Common Viz Framework!
 - ⇒ Collaborate, Generalize, Collaborate...
 - ⇒ Social & Political Obstacles
 - ⇒ Need Flexible & Efficient Infrastructure!

RDV Framework Criteria

- High-Performance / High-Bandwidth
 - ⇒ Cannot Sacrifice Performance for “Glue”
- Commonality
 - ⇒ Everyone’s Stuff Has To Fit!
- Ease of Integration, Simplicity
 - ⇒ Minimal Wrappers & Cruft, Appls & Viz
- Generalized Interfaces!
 - ⇒ Interoperability ~ Alternate Solutions
 - ⇒ Code Re-Use; Share Expertise...

Why Components?



The task of the software development team is to engineer the illusion of simplicity [Booch].

Common Component Architecture (CCA) Components & Frameworks

- Strict Component Boundaries
 - ⇒ Enforces Interfaces Better than Plain OO
- High-Performance Solutions
 - ⇒ “Direct-Connect” Short-Circuiting
 - Virtual Function Call Overhead Within Process
 - ⇒ “MxN” Parallel Data Redistribution
 - High-Level Parallel Data Exchanges
 - Substrate for Inter-Framework Interactions
- A Good Foundation for RDV...? (Yup. 😊)

No “CCA Sunshine”...

- CCA is Good Component Infrastructure, But...
- It's Not Easy to Generalize & Collaborate
 - ⇒ Hasn't really been fully done before for Viz...
 - No Single Authoritative Flow-Based Framework!
 - ⇒ Requires *EXTRA* Effort
 - Not Just the Coolest, Strongest, Fastest Toys...
 - Interface Development Takes Careful Thought
 - Must Work TOGETHER (else useless...)
- Big Payoff ~ Cooperate, Not Compete...

Where to Start?

- Basic Viz Functions
 - ⇒ Data Analysis (SDM), Transmission, Rendering...
 - ⇒ Initial Interfaces Should Cover Existing Work
 - Informal “Standards” are a Good Start!
- Common Distributed Data Model
 - ⇒ Describe Existing Data Organizations
 - ⇒ Work Already In Progress ~ SciDAC TSTT & CCA
 - ⇒ *Viz Community* Should Join the Fray...

Then What?

- Build Full Parallel/Distributed Viz Pipeline
 - ⇒ Use Emerging “MxN” Technology
 - ⇒ Generalize Functional Blocks
 - Introduce WAN as Adjustable Stage in Pipeline?
- Viz Cache Architecture ~ ORNL, UTK, OSU
 - ⇒ Multiple Parallel Clusters, Daisy-Chained...
 - ⇒ Not a Specific Solution → General Framework