RAPIDS – An Overview

9 February 2004

Andrew Cox

Technical Director

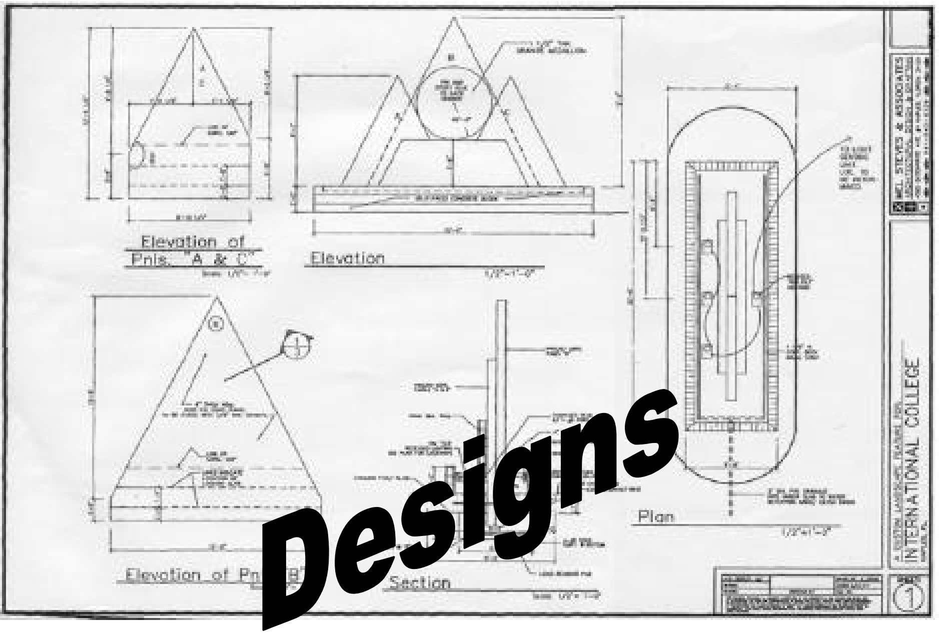
PEO C4I and Space

What is RAPIDS?

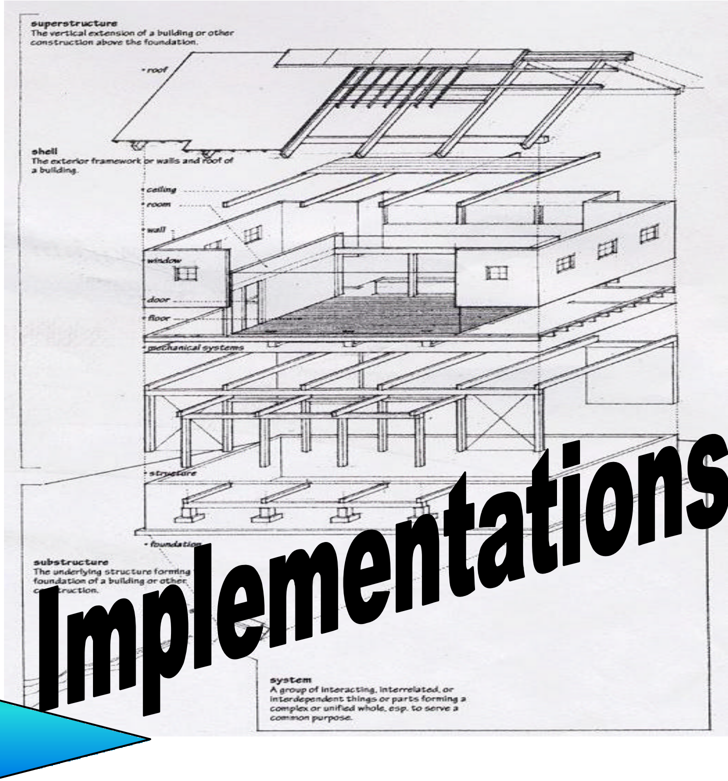
Reusable Application Integration and Development Standards

It’s not an architecture, it’s the building codes

RAPIDS Guidance



Designs


superstructure The vertical extension of a building or other construction above the foundation.

shell The exterior framework or walls and roof of a building.

substructure The underlying structure forming foundation of a building or other construction.

system A group of interacting, interrelated, or interdependent things or parts forming a complex or unified whole, esp. to serve a common purpose.

Implementations

Architectures, Frameworks, Patterns, Mechanisms

RAPIDS Is …..

- Technical
- Contractual/Legal/IP rights management
- Acquisition Processes
- Software Development process management
- Open Source
- Collaborative Software Development
- Compliance/Enforcement processes

What's the Problem?

- Our current software development process is not efficient enough to support the warfighter
  - Can't keep up with technology
  - Can't keep up with changing fleet needs (Sailor Alts)
- Current system designs make reacting to ever changing needs cost prohibitive
  - "Start Over", Kill, or Evolve?
  - Continuous code "face lifts" are stretching our financial skin
  - Platform-centric designs make network operation difficult

What's the Cause?

- ROEs change with each deployment
  - "Sailor-alts" are necessary to engage the enemy
- Installation timelines are excessive
  - D-30 will guarantee obsolescence
- Requirements and POM processes are slow and bureaucratic
  - "I wanted that *last* year"
- "Initiatives du Jour" are difficult to react to in a cost effective and timely manner
- Code re-use in the DoD is virtually non-existent
  - Each new initiative spawns a duplicate copy of similar functionality

RAPIDS Objectives

- **Maximize Flexibility**
  - **Develop code that can be customized by others**

"However One Wants"

- **Maximize Reuse**
  - **Structure code so that others can re-utilize components**

"By Whomever One Wants"

- **Maximize Portability**
  - **Implement standards carefully to promote re-use across multiple architectures**

"Wherever One Wants"

Maximize Flexibility

- Leverage Tools and initiatives unleashed by IT-21
  - Savvy fleet users trying to “fill the gaps” in fielded capability (a la Quiver, A3, CAS, etc.)
- Make applications support dynamic operations/rules of engagement
  - Warfighter customization
  - Risk reduction
- Enable 3rd party developers to extend capabilities without duplicating what’s available

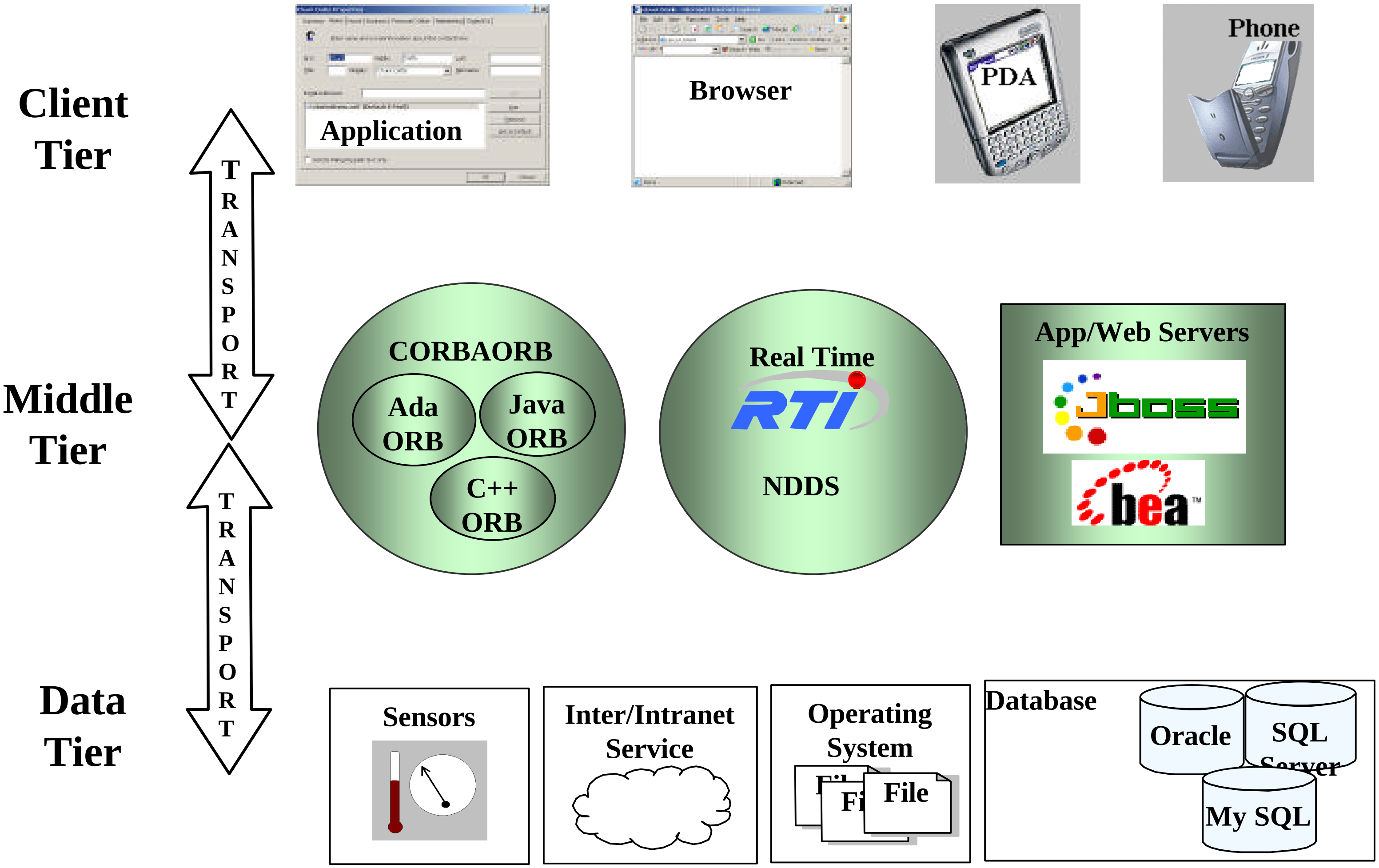
Maximize Reuse

- Create“parts store”-- components/functions are available for mixing and matching to create new capabilities
  - Incentives to contractors for re-use?
    - Different models to imitate (ARCI/APB)
- Write code that promotes re-use
  - E.g. separation of interface from implementation
  - Limited Open Source development environment
- Reducing porting costs and redundancy
  - Keeping up with “Initiatives du Jour”

Maximize Portability

- Avoid vendor-specific “extensions” to standards that decrease portability
  - J2EE and Deployment Descriptors
  - CORBA and IIOP
  - Database connectivity
- Portability across the DoD
  - Provide cross-service code re-use opportunities
  - Ability to communicate across the real-time/non real-time boundary (*working the real-time/shared-time issues)*
- Get “own house” in order first
  - Navy applications that plug-n-play with each other
  - Open source standards provide greater portability

Three Tier Architecture



Client Tier

Application

Browser

PDA

Phone

TRANSPORT

Middle Tier

CORBAORB

Ada ORB

Java ORB

C++ ORB

Real Time

RTI

NDDS

App/Web Servers

JBoss

bea

TRANSPORT

Data Tier

Sensors

Inter/Intranet Service

Operating System

File

Database

Oracle

SQL Server

My SQL

RAPIDS End State

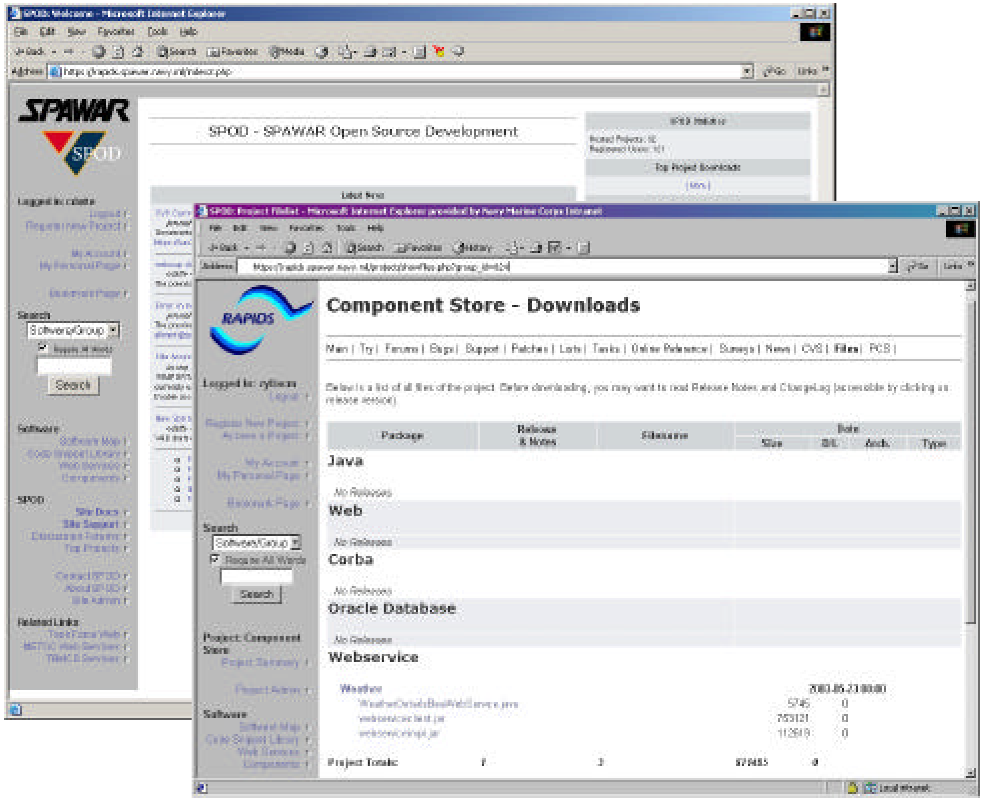
- Short term
  - Mitigation strategy to reduce risk and cost in trying to meet all the orthogonal mandated initiatives

- Long term
  - Makes development and integration of capabilities the shortest cycle in the “speed to capability” process

RAPIDS Media

- Collaborative Software Development Site
  - Source Forge Model
  - Web site for submitting and sharing source and object level code
    - 2 NIPR & 1 SIPR Sites: over 40 projects and growing
  - Products for assessing project complexity/costs
  - Project management tools
  - Parts/Services store
- Developer's Guidance documents
  - Detailed implementation guidance
    - Main document (Version 1.4 Released)
    - Recipes – 1 page quick tips
    - Code samples validated/tested in multiple enterprise domains/architectures ("Java Pet Store" approach)
  - Considers NAVAIR/NAVSEA input relevant to OA
- Technical Support

RAPIDS REPOSITORY



SPOD - SPAWAR Open Source Development

Component Store - Downloads

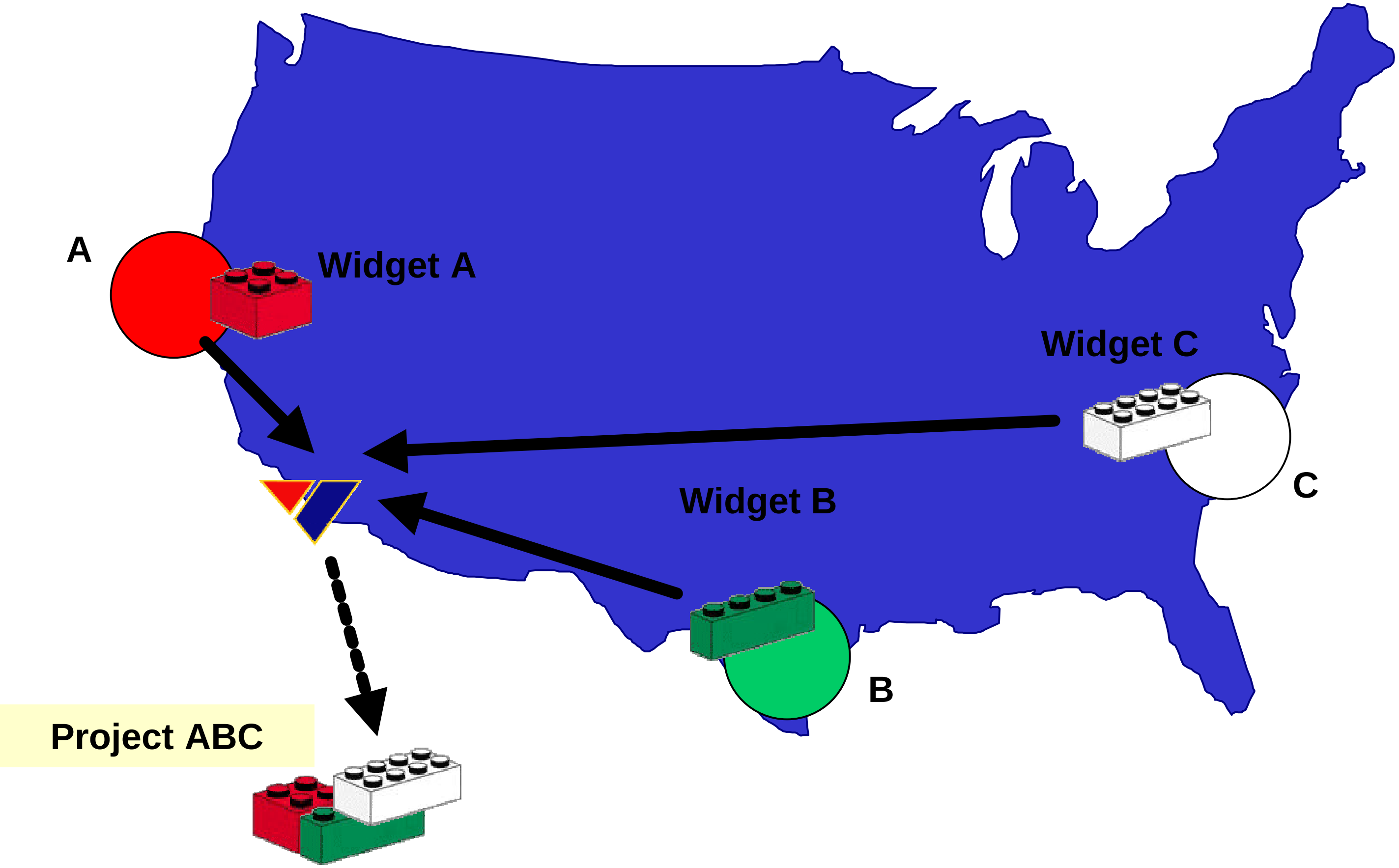
Main | Try | Forums | Bugs | Support | Patches | Lists | Tasks | Online Reference | Surveys | News | CVS | **Files** | PCS |

Below is a list of all files of the project. Before downloading, you may want to read Release Notes and ChangeLog (accessible by clicking on release version).

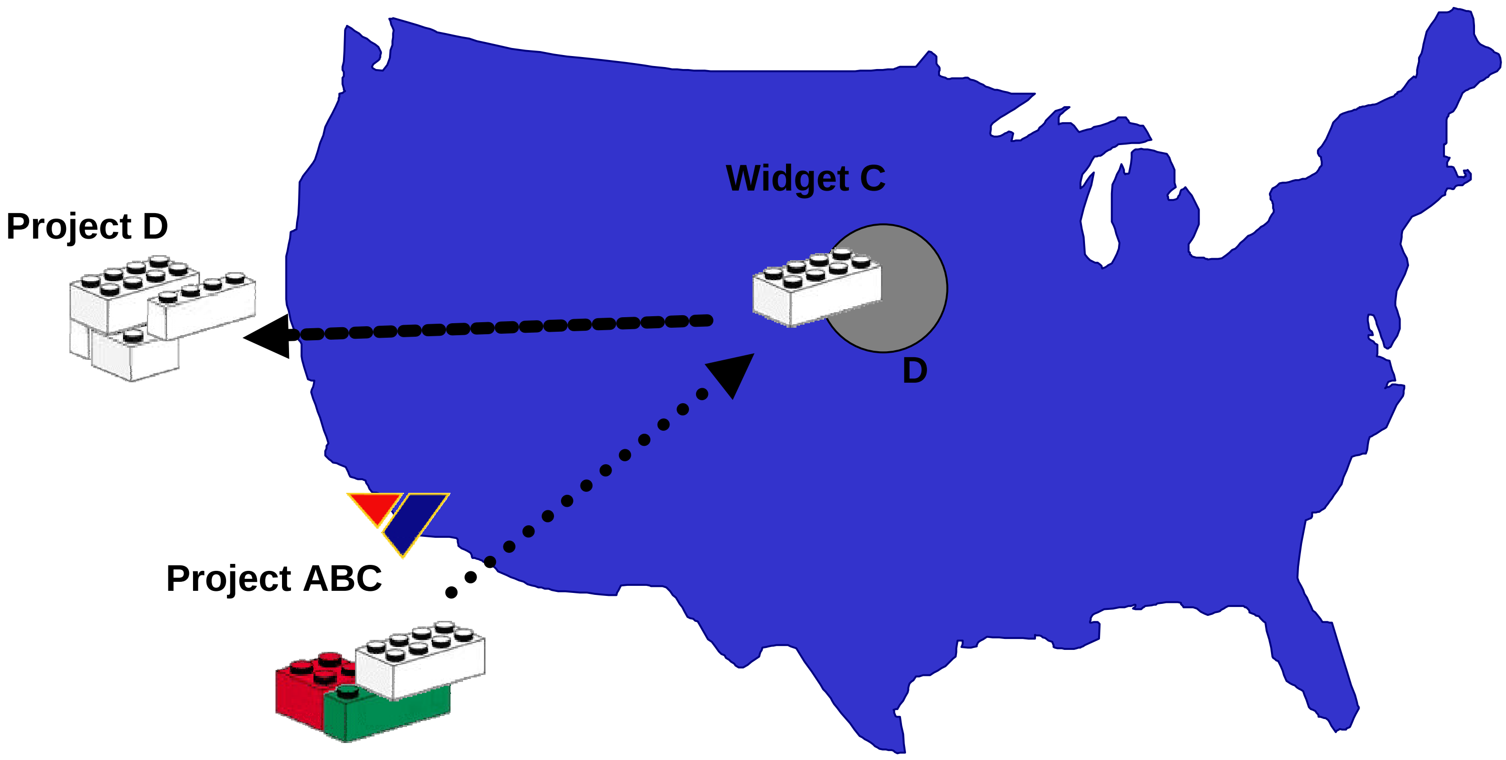
Package	Release & Notes	Filename	Date			
			Size	D/L	Arch.	Type
Java						
No Releases						
Web						
No Releases						
Corba						
No Releases						
Oracle Database						
No Releases						
Webservice						
	Weather		2003-05-23 00:00			
		WeatherDetailsBeanWebService.jws	5745	0		
		webservicetest.jar	753121	0		
		webservicexmpl.jar	112619	0		
Project Totals:	7	3	871485	0		

- Collaborative Web-site
  - “Parts Store”
  - Guidance materials
  - Code samples
  - Based on Open Standards
- Many hardware- and middleware- independent services available
- Speed to Capability
- Joint: Software developed without ties to unique service implementations

Collaborative Development



Collaborative Development



RAPIDS Legal

- Developed RAPIDS, *Managed Open Source* Boilerplate contract language for use in TDL contract efforts
  - Contracts
  - Legal (issue with existing/old code base)
  - Finance
- Language is structured to be plug ‘n play independent of particular basic contract vehicle
  - Applicable to SW development and repair
  - Enables PEO to institutionalize RAPIDS concepts and processes into s/w procurements
- Drafted PEO Memo announcing RAPIDS as SOP for SW development and repair

Coordination Efforts

- SPAWAR Forcenet
- Open Architecture PEO-IWS
  - Developer’s Guidance convergence
  - Joint Experiments
- PEO W
- JOINT
  - DISA - GIG-ES, NCES
  - Collaborative Software development and Open Source
  - Joint Tactical Radio System (JTRS) SCA
- Air Force C2ERA
  - RAPIDS provides detailed implementation guidance for C2ERA
  - C2 Constellation (C2C)
  - Joint Experiments
    - ✓ Prove out Guidance Documents
- Army Future Combat Systems (FCS)
  - RAPIDS provides Collaborative software development guidance

Standards

- PEO IWS Open Architecture (OA) currently provides well-defined standards guidance at a high level
  - Use of CORBA 3.0, adherence to POSIX, etc.
- PEO C4I experience is that lower-level specifications are required to ensure interoperability and portability
- Due to different levels of focus, initial view is that OADG and RAPIDS are highly complimentary
  - Portions that are C4I-specific (i.e. web services) and CDS-specific (i.e. real-time, adaptive middleware) may remain domain-specific
- Efforts ongoing to consolidate OADG and RAPIDS
  - Use example functions (e.g. CEC track processing, ELINT track processing, time) to determine which portions of combined outline apply

Tools

- Both PEO IWS and PEO C4I & SPACE have tools that may potentially be applied in both domains
- PEO IWS rigor on resource management and failover may serve C4I requirements
  - Re-utilization would reduce training, pave the way towards more commonality
- PEO C4I experience with Information Assurance and INFOSEC much more significant
  - Re-utilization would ensure consistent implementation of defense-in-depth strategy
- Some tools may only make sense within Combat System sensor processing domain, others within C4I domain

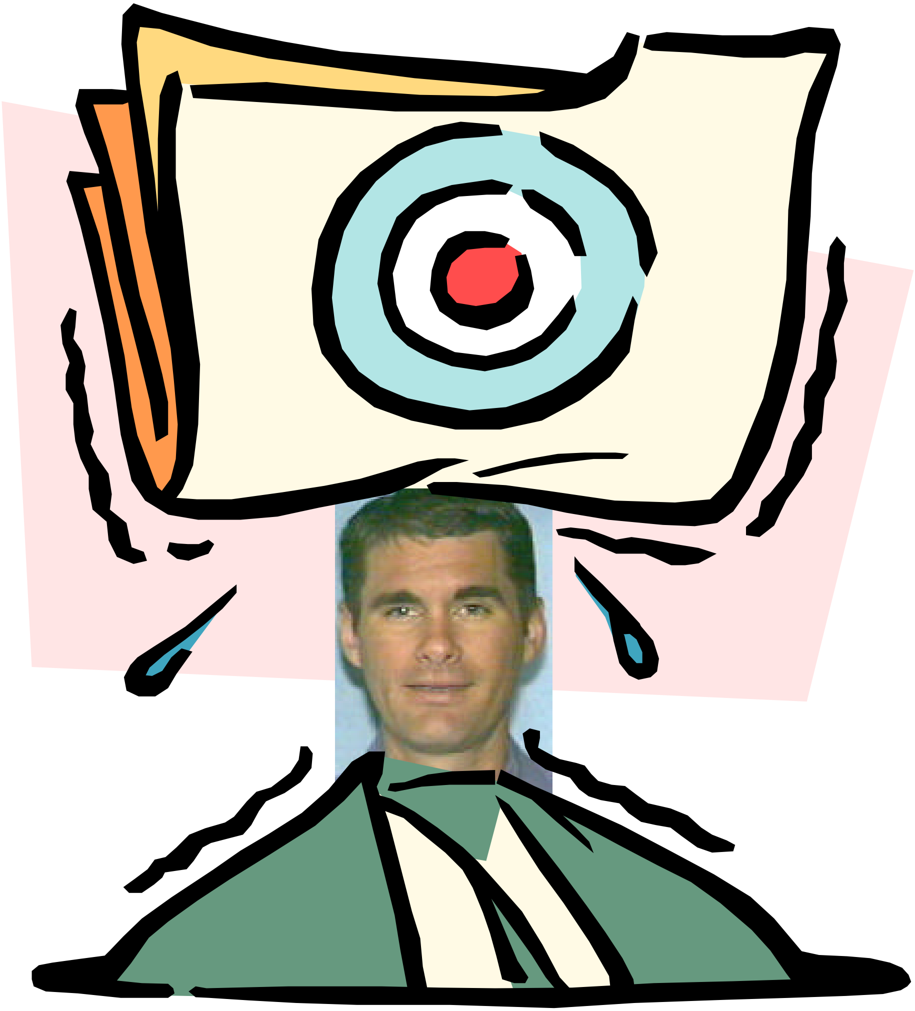
Issues

- Tactical Picture Re-design
  - Redesign of JPN, JDN, and JCTN are underway
    - All are aligning under similar design philosophies
  - Publish/subscribe framework approaches are different
    - COP = UDDI, WSDL framework for plug-and play data sources
    - Combat Systems = NDDI by Real Time Innovations
      - ✓ Not yet a standard…evolving to OMG DDS
- Operating System Independence
  - Use of a GOTS “adaptive middleware” may not solve C4I requirements
    - Need to make old code base work on new hardware is rare
- Resource Management/Failover
  - Current OA product (DRM) is a GOTS product that promotes centralized (not federated) computing environment
    - Will not work for specialized server environments
  - Need to understand roles COTS could play

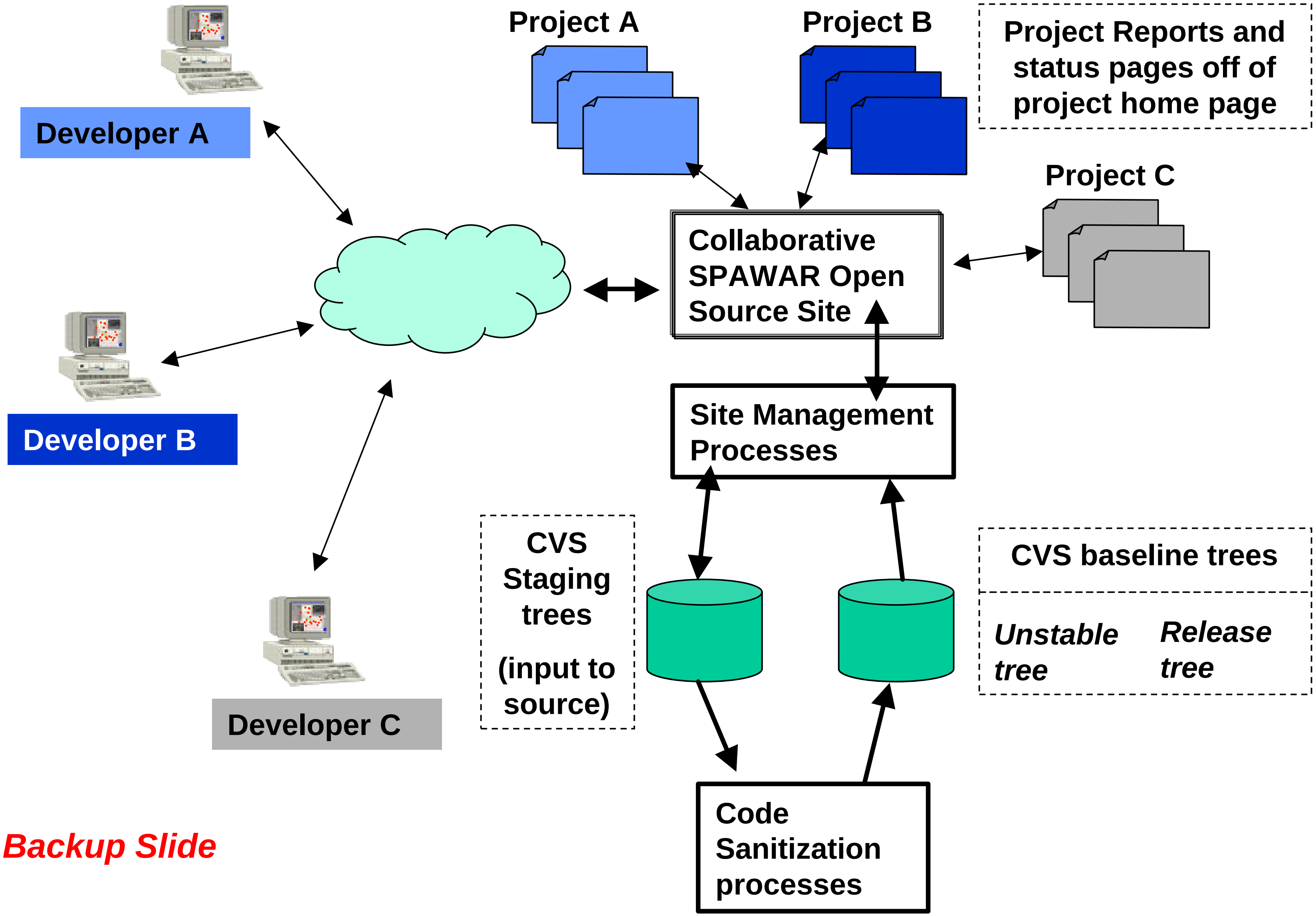
Future Vision

- Infrastructure providers
  - Open standards/interfaces
  - Commercial tools, products, practices
- Interface providers
  - Robust, decoupled interfaces that insulate the service provider from being disrupted by activities of external applications.
- content providers
  - Appropriate interfaces that enforce necessary integrity and security requirements while maximizing accessibility of content.
- Application providers
  - Build portable, componentized business logic in accordance with commercial standards and specifications.
- Warfighters
  - Leverage tools provided by IT-21 imitative to provide “VAR” extensions to applications, completely new components

Questions?



RAPIDS Process Overview



Developer A

Developer B

Developer C

Project A

Project B

Project C

Project Reports and status pages off of project home page

Collaborative SPAWAR Open Source Site

Site Management Processes

CVS Staging trees

(input to source)

CVS baseline trees

Unstable tree

Release tree

Code Sanitization processes

Backup Slide