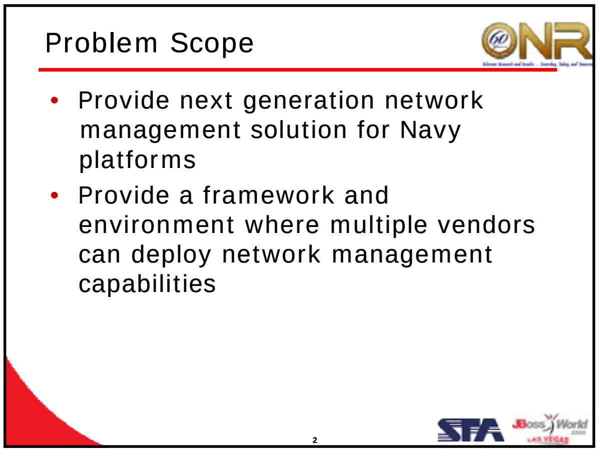
Extending Future Naval Communications with JBoss

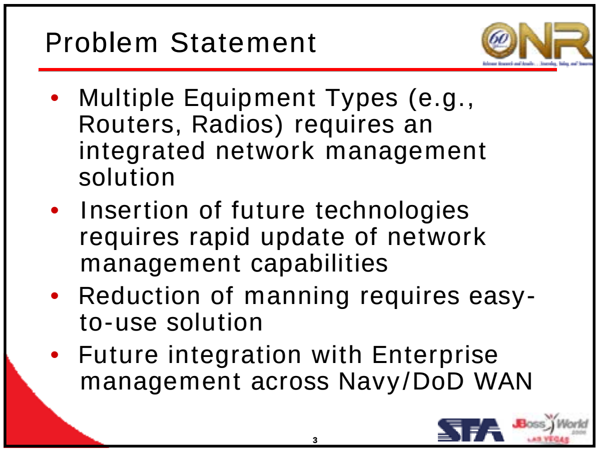
Dean Knickerbocker
SFA Inc
dknick@sfa.com
757-962-3960

Brett Carpenter
SFA Inc
bcarpenter@sfa.com
757-962-3945



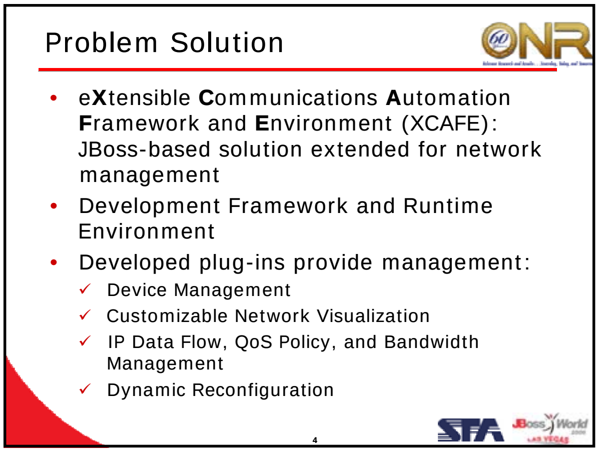
Problem Scope

- Provide next generation network management solution for Navy platforms
- Provide a framework and environment where multiple vendors can deploy network management capabilities



Problem Statement

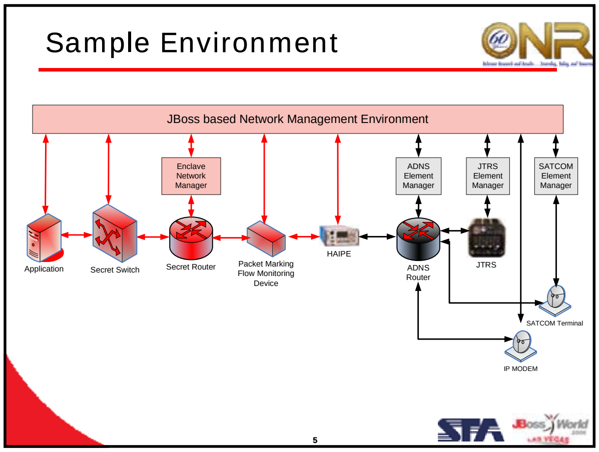
- Multiple Equipment Types (e.g., Routers, Radios) requires an integrated network management solution
- Insertion of future technologies requires rapid update of network management capabilities
- Reduction of manning requires easy-to-use solution
- Future integration with Enterprise management across Navy/DoD WAN

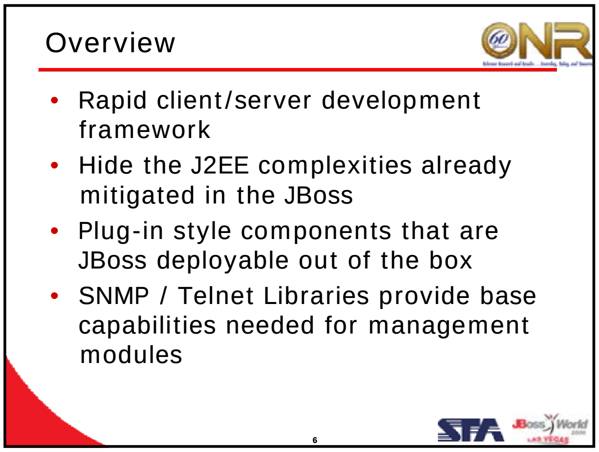


Problem Solution

- e**X**tensible **C**ommunications **A**utomation **F**ramework and **E**nvironment (XCAFE): JBoss-based solution extended for network management
- Development Framework and Runtime Environment
- Developed plug-ins provide management:
  - ✓ Device Management
  - ✓ Customizable Network Visualization
  - ✓ IP Data Flow, QoS Policy, and Bandwidth Management
  - ✓ Dynamic Reconfiguration

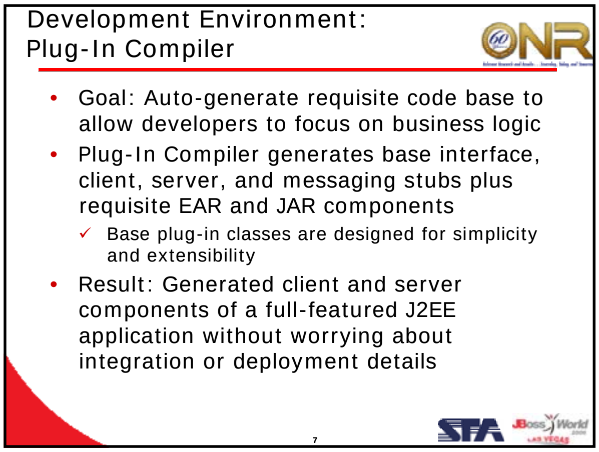
Sample Environment





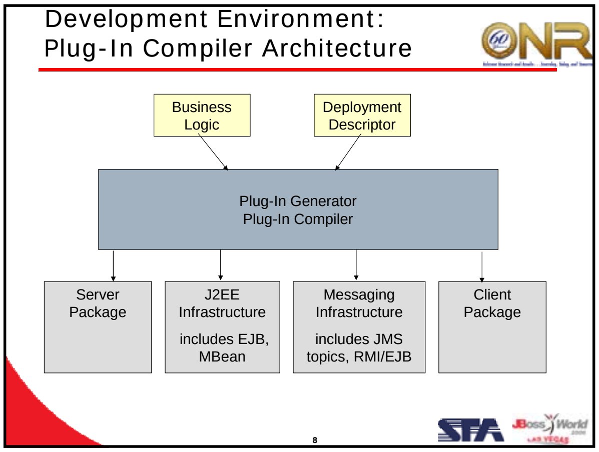
Overview

- Rapid client/server development framework
- Hide the J2EE complexities already mitigated in the JBoss
- Plug-in style components that are JBoss deployable out of the box
- SNMP / Telnet Libraries provide base capabilities needed for management modules

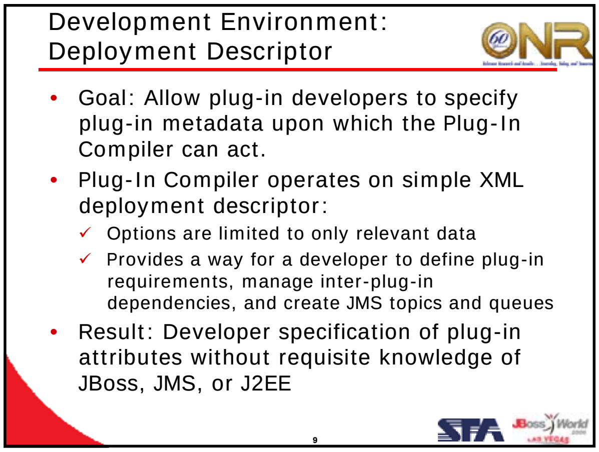


Development Environment: Plug-In Compiler

- Goal: Auto-generate requisite code base to allow developers to focus on business logic
- Plug-In Compiler generates base interface, client, server, and messaging stubs plus requisite EAR and JAR components
  - ✓ Base plug-in classes are designed for simplicity and extensibility
- Result: Generated client and server components of a full-featured J2EE application without worrying about integration or deployment details

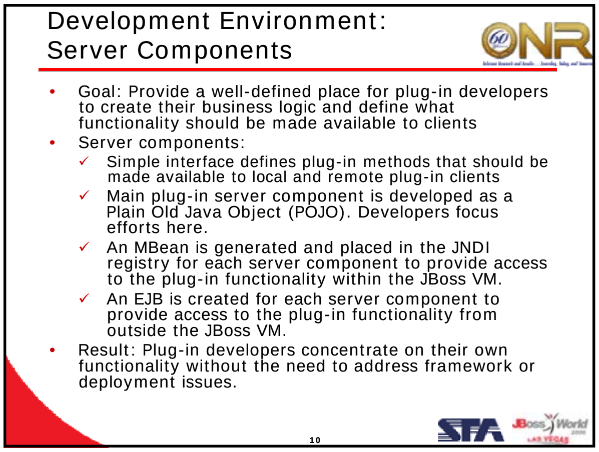


Development Environment: Plug-In Compiler Architecture



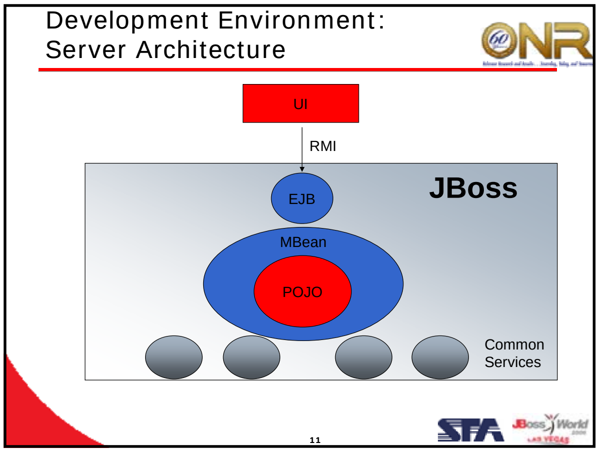
Development Environment: Deployment Descriptor

- Goal: Allow plug-in developers to specify plug-in metadata upon which the Plug-In Compiler can act.
- Plug-In Compiler operates on simple XML deployment descriptor:
  - ✓ Options are limited to only relevant data
  - ✓ Provides a way for a developer to define plug-in requirements, manage inter-plug-in dependencies, and create JMS topics and queues
- Result: Developer specification of plug-in attributes without requisite knowledge of JBoss, JMS, or J2EE

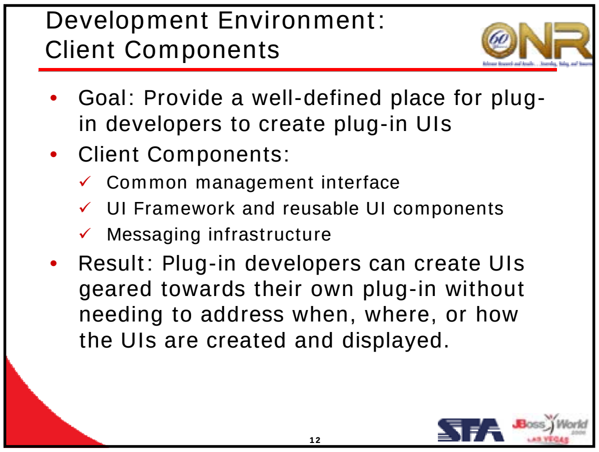


Development Environment: Server Components

- Goal: Provide a well-defined place for plug-in developers to create their business logic and define what functionality should be made available to clients
- Server components:
  - ✓ Simple interface defines plug-in methods that should be made available to local and remote plug-in clients
  - ✓ Main plug-in server component is developed as a Plain Old Java Object (POJO). Developers focus efforts here.
  - ✓ An MBean is generated and placed in the JNDI registry for each server component to provide access to the plug-in functionality within the JBoss VM.
  - ✓ An EJB is created for each server component to provide access to the plug-in functionality from outside the JBoss VM.
- Result: Plug-in developers concentrate on their own functionality without the need to address framework or deployment issues.



Development Environment: Server Architecture



Development Environment: Client Components

- Goal: Provide a well-defined place for plug-in developers to create plug-in UIs
- Client Components:
  - ✓ Common management interface
  - ✓ UI Framework and reusable UI components
  - ✓ Messaging infrastructure
- Result: Plug-in developers can create UIs geared towards their own plug-in without needing to address when, where, or how the UIs are created and displayed.

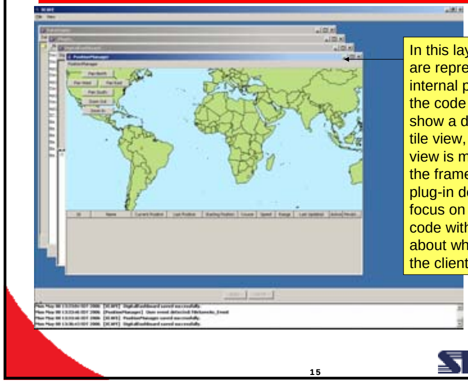
Runtime Environment: Common Services

- Goal: Provide common support for services like SNMP, Telnet, and SOAP.
- Standard services made available for all plug-ins:
  - ✓ Common services easily accessible
  - ✓ Access to these services is exported similar to other plug-ins
  - ✓ Provides mechanism for coordinating access to system resources across disparate plug-ins
- Result: Simplified and coordinated use of basic services needed for network management application

Runtime Environment: UI Management

- Goal: Provide a client architecture that automates tasks such as server communication and JMS connections as well as provides UI integration across multiple plug-ins and gives the end user flexibility in viewing and interacting with different XCAFE plug-ins.
- Developed UI management tool:
  - ✓ Handles details of plug-in deployment and overall presentation by interacting with plug-ins via the common management interface.
  - ✓ Interacts with plug-ins through generic interfaces, so the tool can be updated, extended, or replaced to add additional functionality such as layout management.
- Result: An XCAFE client framework deployable locally or remotely as a Java application or applet that requires only a single, simple implementation at the plug-in level.

Runtime Environment: UI Management

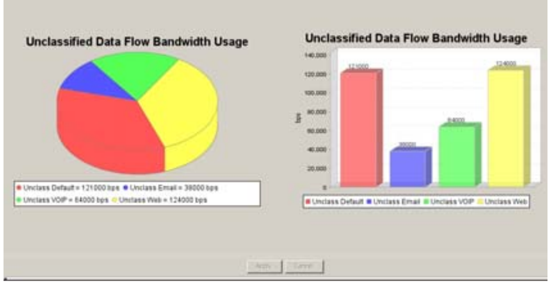


In this layout, clients are represented as internal panes. All of the code necessary to show a desktop view, a tile view, or a tabbed view is maintained at the framework level, so plug-in developers can focus on their own code without worrying about where or how the clients are shown.

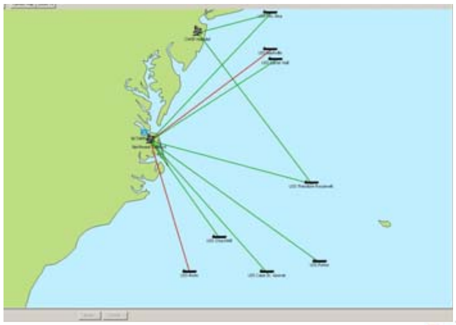
Runtime Environment: Other Features

- Access Control capability to provide single login mechanism and manage access to plug-ins and features
  - ✓ Single Sign On across HTML Portal pages and Java applets
- Portal to navigate across all plug-ins from single location
- Aggregates parameters, performance, and trends from multiple devices/managers into a common data repository
- Custom data logging and report generation

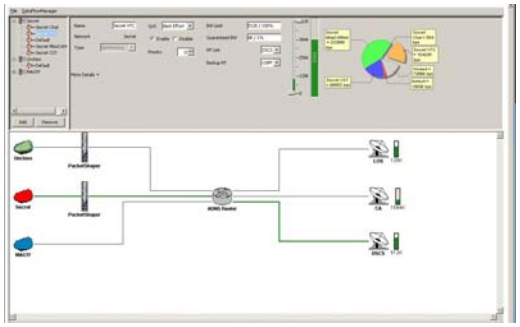
Results: Customizable Network Visualization



Results: Customizable Network Visualization (Geospatial Display)



Results: Drag and Drop IP Management



Summary

- Navy’s next generation network management solution is being built on top of JBoss
- Multiple developers across multiple companies are developing components without having to understand intricacies of J2EE:
  - ✓ Reduced development time
  - ✓ Robust and reliable deployment
- Focus is on management requirements not software infrastructure