Booz | Allen | Hamilton

Collaboration space, Alexandria, VA

An Agile, Cloud Based Common Software Framework

Containerization of the PGMM Framework and benefits of leveraging common frameworks and development environments to accelerate development and deployment of new capabilities

FEB 2018

OVERVIEW

- Background
- PGMM Framework Evolution Approach
- Common Framework and Development Environment Benefits

PGMM BACKGROUND

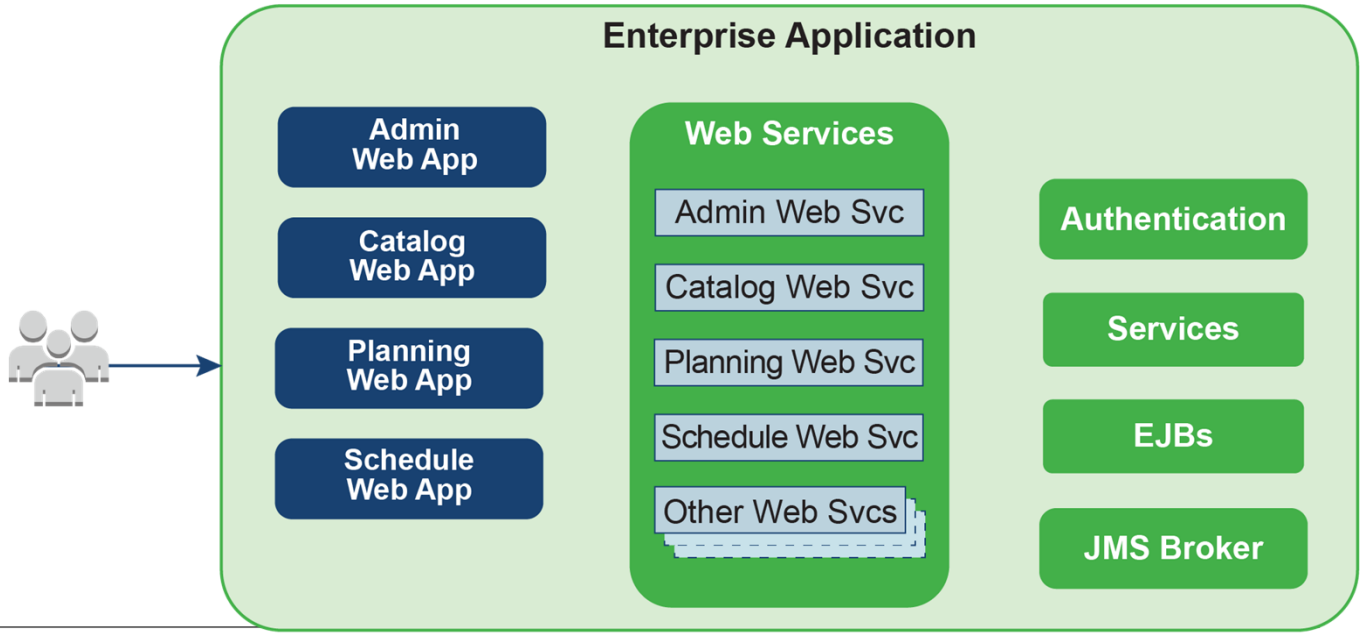
- Persistent GEOINT Mission Manager (PGMM) develops a common software infrastructure/framework for OPIR constellation mission management
- Evolve Virtual Mission Operations Center (VMOC) to open, modular, loosely coupled cloud-based architecture with standard interfaces and data models
  - VMOC is a single tightly-coupled monolithic deployment of a Java Enterprise Architecture
- Extend framework to add new and updated mission capabilities
  - Sensor agnostic constellation scheduling
  - Advanced scheduling and feasibility capabilities
- Provision and provide a common Government-owned cloud-based development environment for third-party app developers
  - Support third-party developers throughout their development efforts
  - Serve as system integrator responsible for end-to-end system performance

PGMM FRAMEWORK EVOLUTION APPROACH

- Leverage Containerization technologies and micro service architecture principles
  - Migrate framework functions to containers incrementally over time
  - Enables open, modular and loosely coupled architecture
  - Allows third-party application developers to deliver self-contained applications to PGMM's plug and play architecture
  - Allows simplified deployments, finer grained scalability, improved resiliency and resource consumption
  - Provides ability to update individual components without redeploying entire Framework
- Replace COTS with FOSS
  - Replace Weblogic and Oracle DB with Wildfly, Artemis and Postgress DB
  - Keycloak for identity and access management

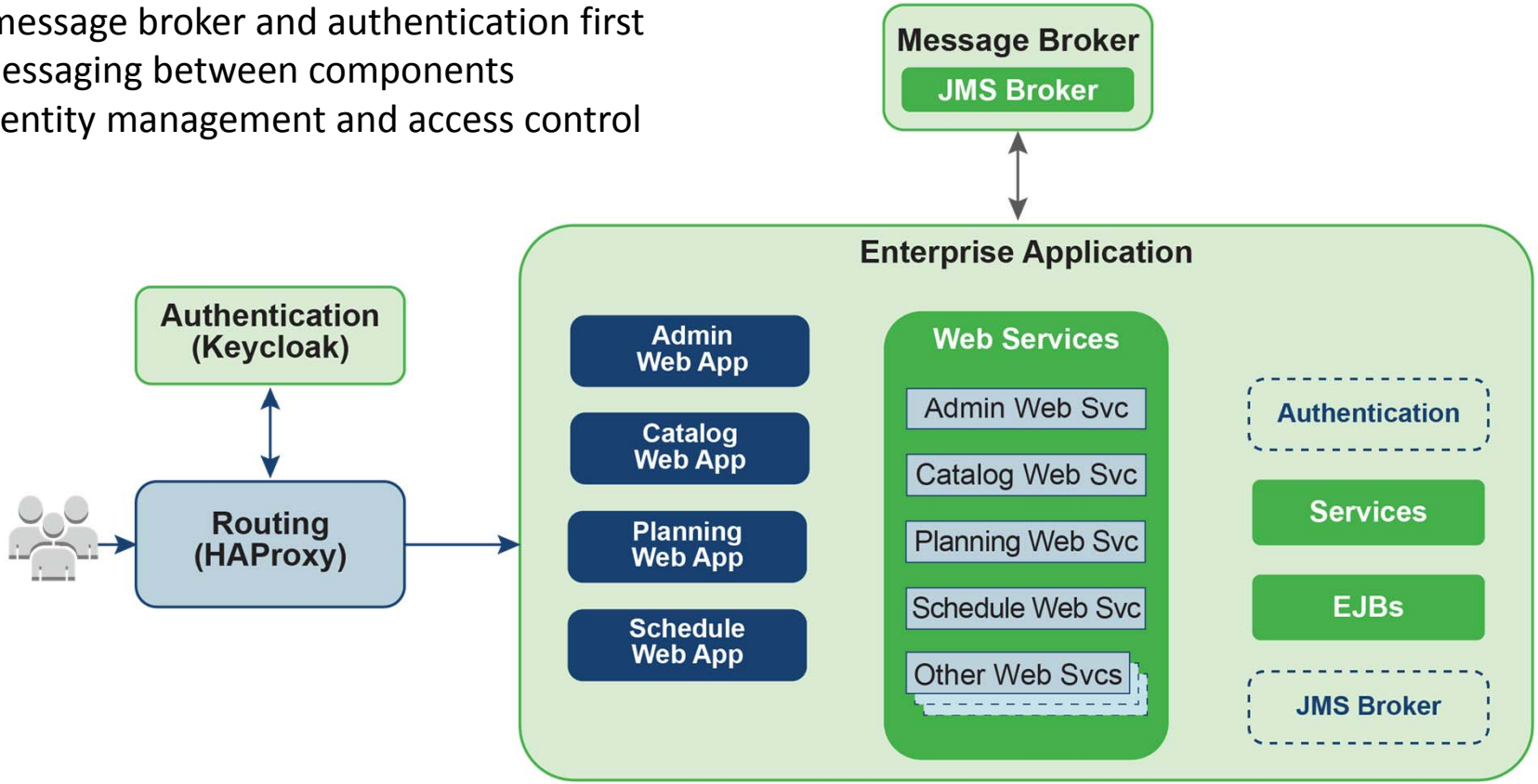
ORIGINAL VMOC GFE FRAMEWORK

- GFE Framework is a single tightly-coupled monolithic deployment of a Java Enterprise Architecture
  - Third party components integrated at build time
  - Upgrades to any single component requires new build and deployment of the entire Enterprise Application
  - Existing design does not easily scale and requires significant server resources
  - Not easily extensible



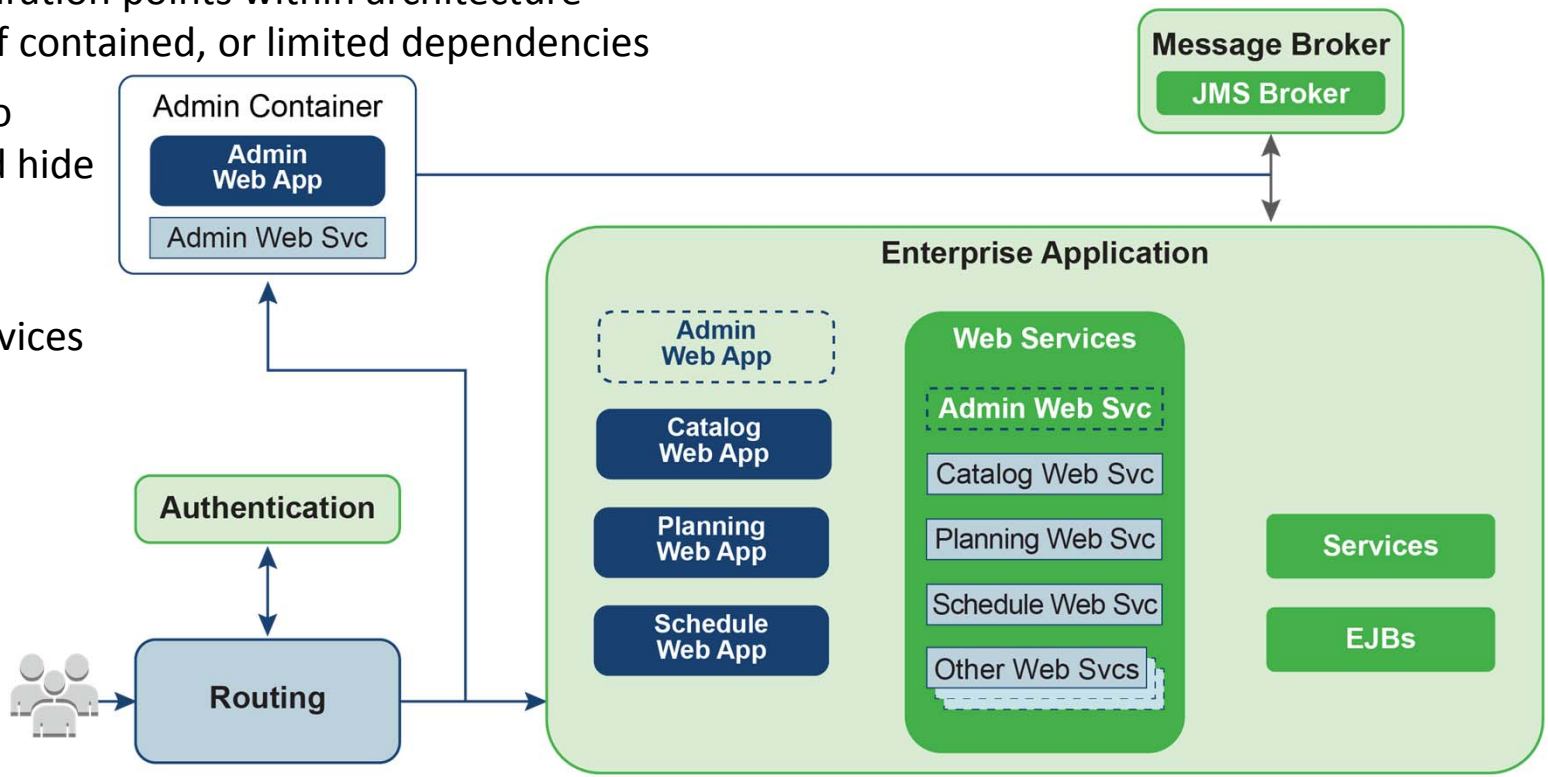
MIGRATING TO CONTAINERIZED ARCHITECTURE

- Centralize message broker and authentication first
  - Enable messaging between components
  - Enable identity management and access control



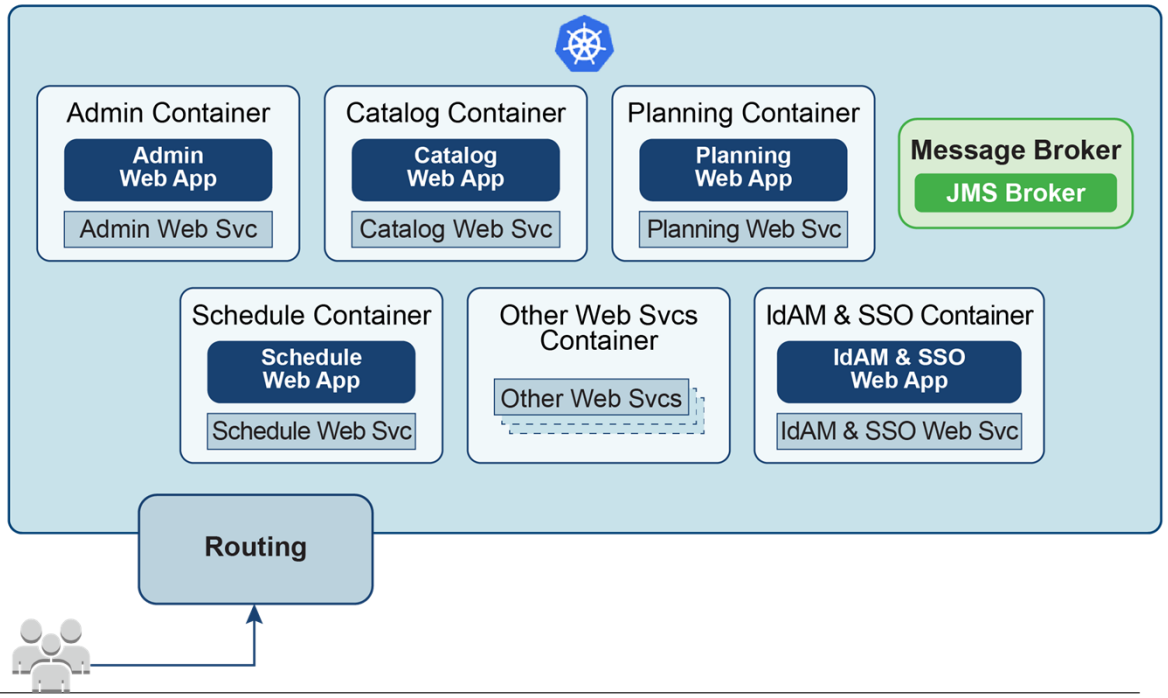
MIGRATING FRAMEWORK COMPONENTS

- New features are added separately from main Enterprise Application
- Look for common separation points within architecture
  - Loosely coupled, self contained, or limited dependencies
- Create service layers to abstract interfaces and hide migration
- Create Rest or JMS implementation of services

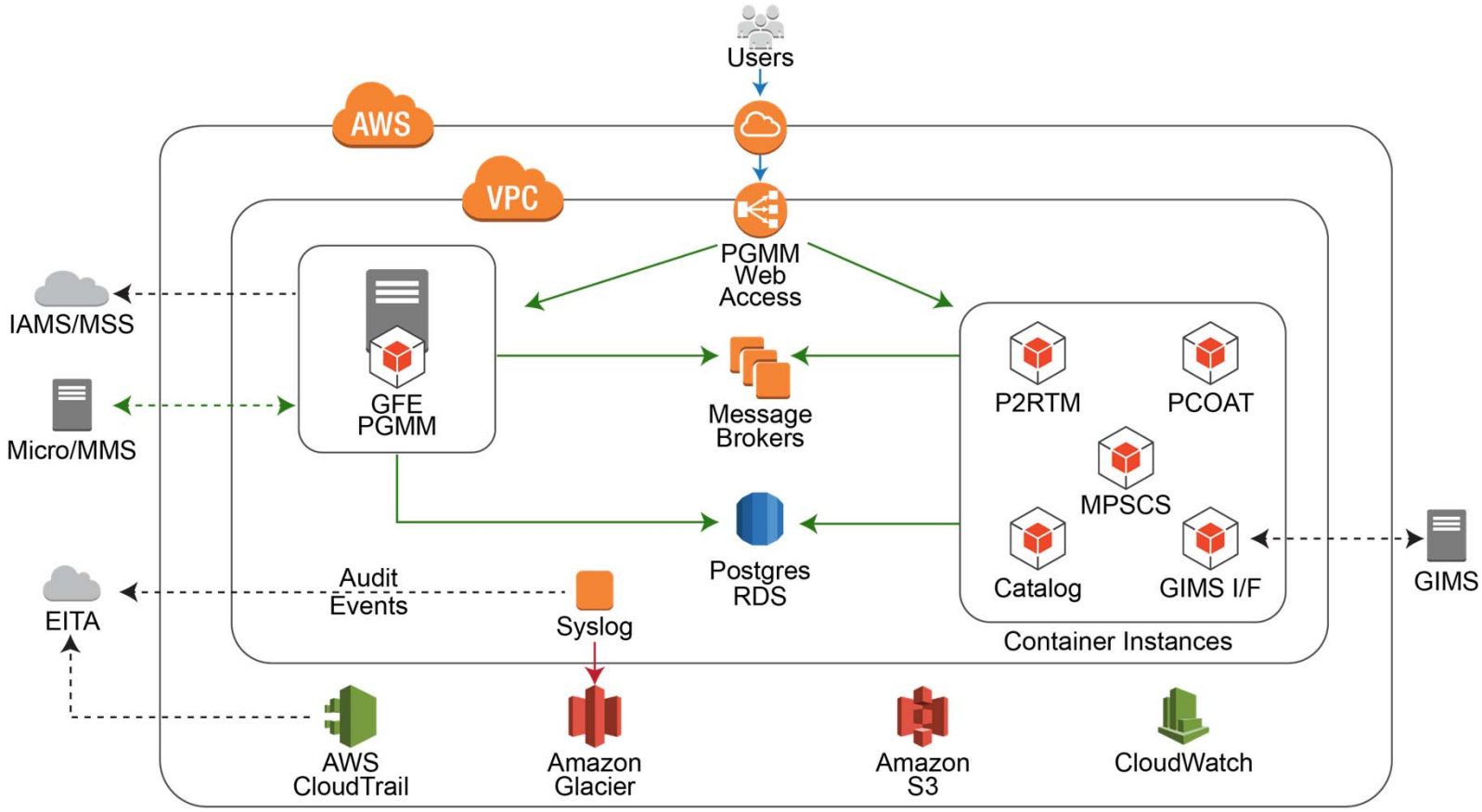


OBJECTIVE CONTAINERIZED ARCHITECTURE

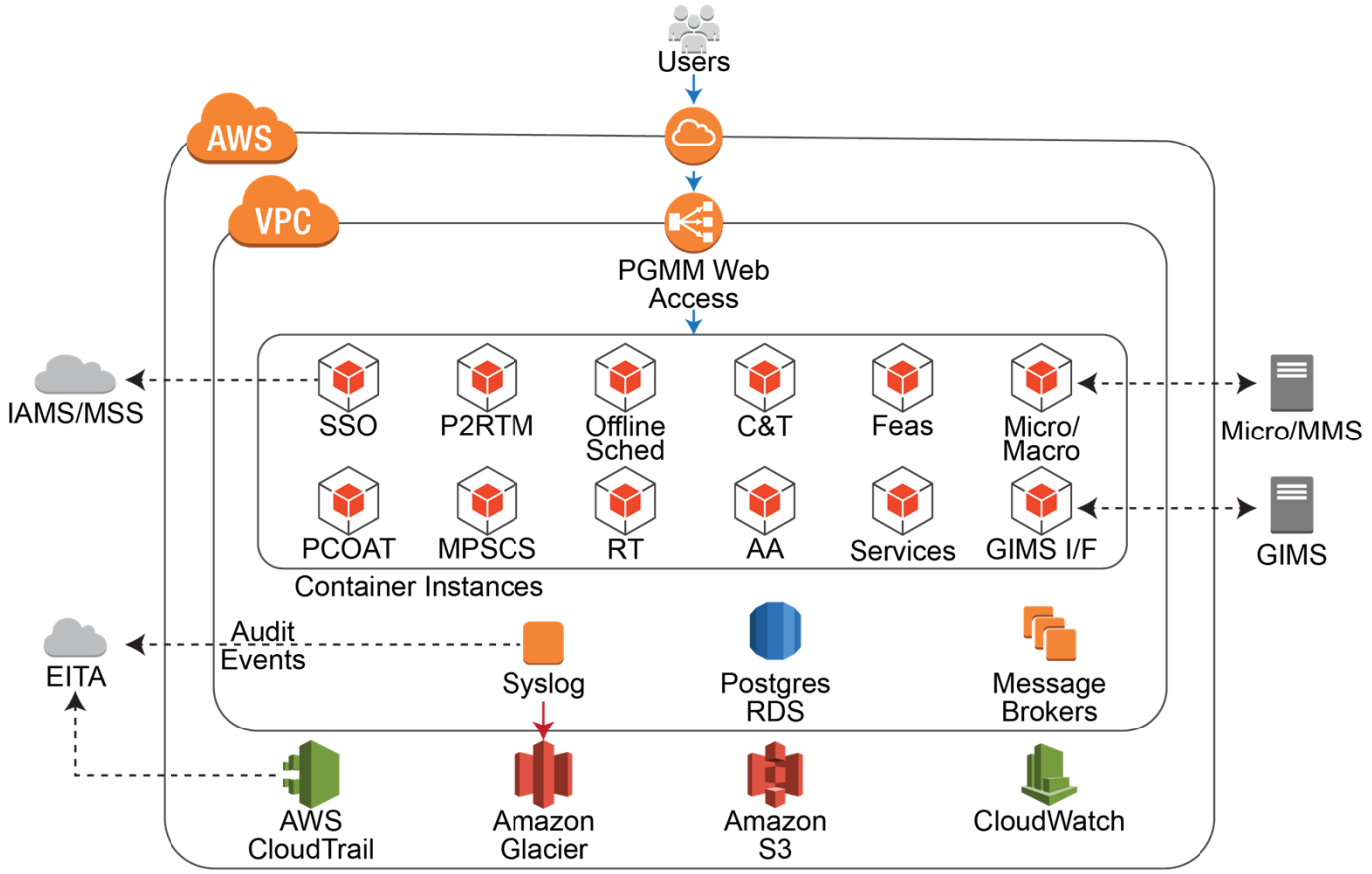
- Objective architecture is fully containerized
  - Message broker and Restful services
- Modular loosely coupled framework that is easily scalable and extensible
  - Micro services architecture
- Modernize as services are rebuilt
  - Springboot, NodeJS, React
  - Minimalistic container OS
  - Lower resource usage
  - Common base container image
  - Reduced A&A effort



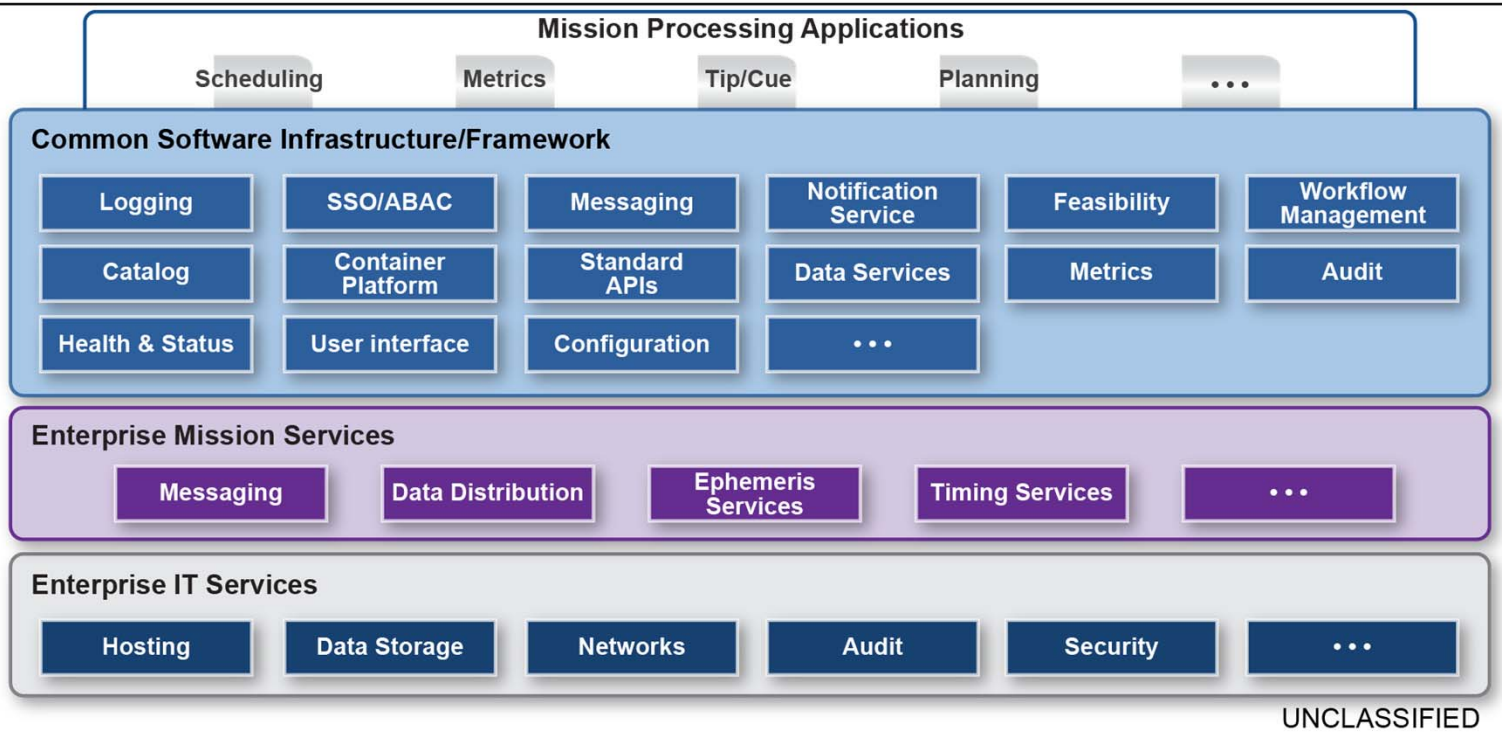
PGMM IOC ARCHITECTURE



PGMM FOC ARCHITECTURE



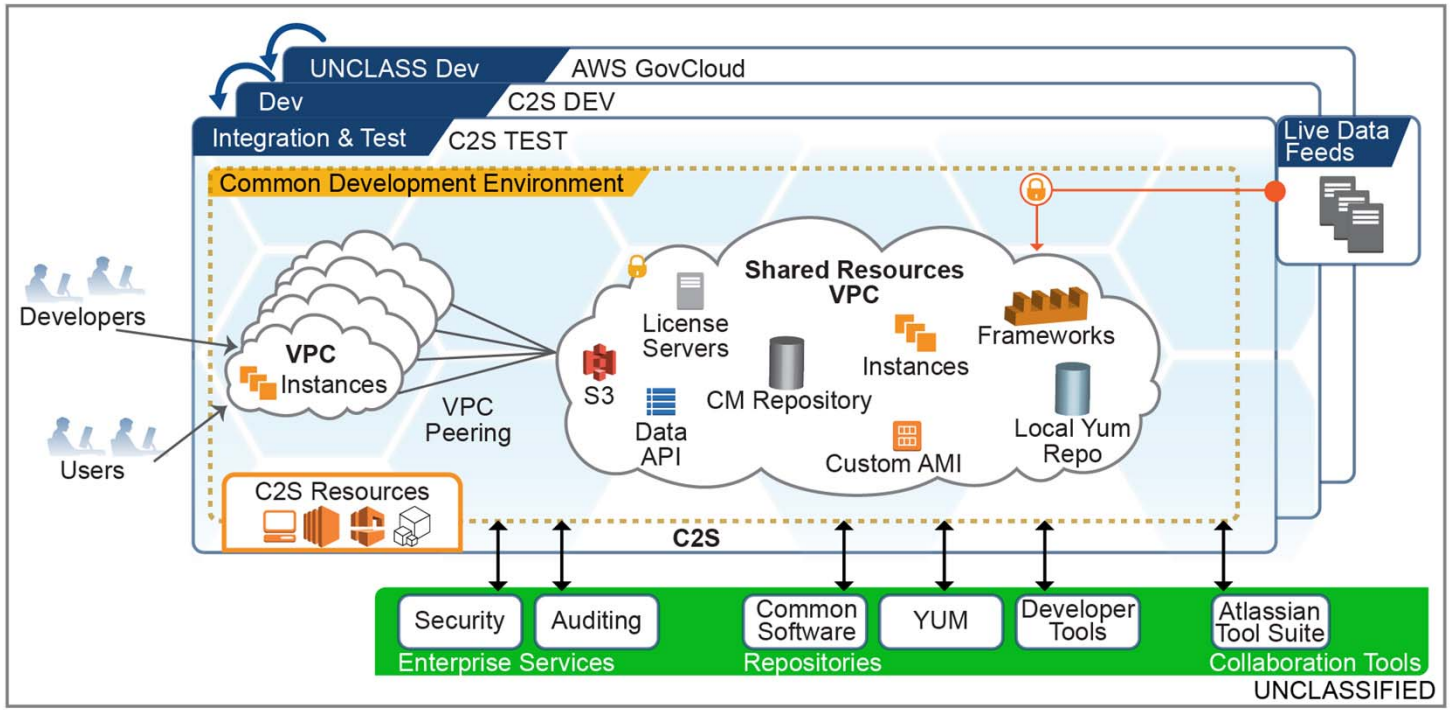
PGMM COMMON MISSION PROCESSING FRAMEWORK



- Provides re-use of common infrastructure elements and services
- Provides applications access to Enterprise IT and Mission Services

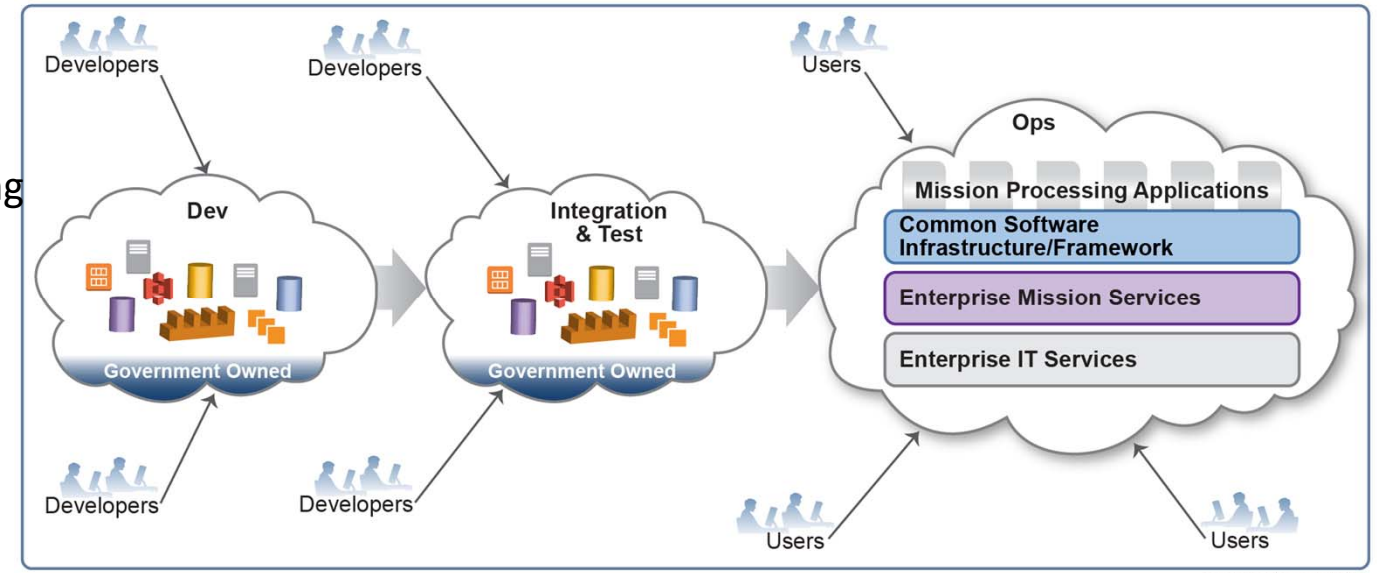
PGMM COMMON DEVELOPMENT ENVIRONMENT BENEFITS

- Provides access to and sharing of common capabilities; framework APIs, software developer guides, repositories, data sets, simulators, security
- Faster project start-ups, lowers cost and safely lowers the bar for entry
- Increased developer efficiency during dev, integration and test
- Lowers bar of entry for new/small industry partners



COMMON DEV ENVIRONMENT AND PROCESSING FRAMEWORK

- Provides Gov possession of software baselines and CM, build, CI/CD environments
- Prevents vendor lock-in and allows smooth hand-off between outgoing and incoming contractors
- Maximizes re-use of common dev environment and framework capabilities
- Reduces integration, test and O&M costs, speeds ops transition
- Enables rapid insertion of new technologies from QRC/R&D efforts



UNCLASSIFIED

QUESTIONS


why
when
who
where
what
why
QUESTIONS
who
how
where
who
what
how
how