

DII COE Real-Time Extensions

David Emery

The MITRE Corporation

(not associated with DISA's
DII COE Engineering office)

Outline

- DII COE overview
 - Current DII COE focus on workstation-based Command and Control systems
 - Provides Interoperability and Portability
 - Reduces maintenance costs via software reuse, reduce training costs via common user interface, system and security administration
 - Provides a common Kernel Platform Certification program
- RT Extensions
 - Extend DII COE into Real-Time C2 and eventually into weapon systems
- Issues for the Vendor & Standardization communities
 - Technical Challenges
 - Business Issues

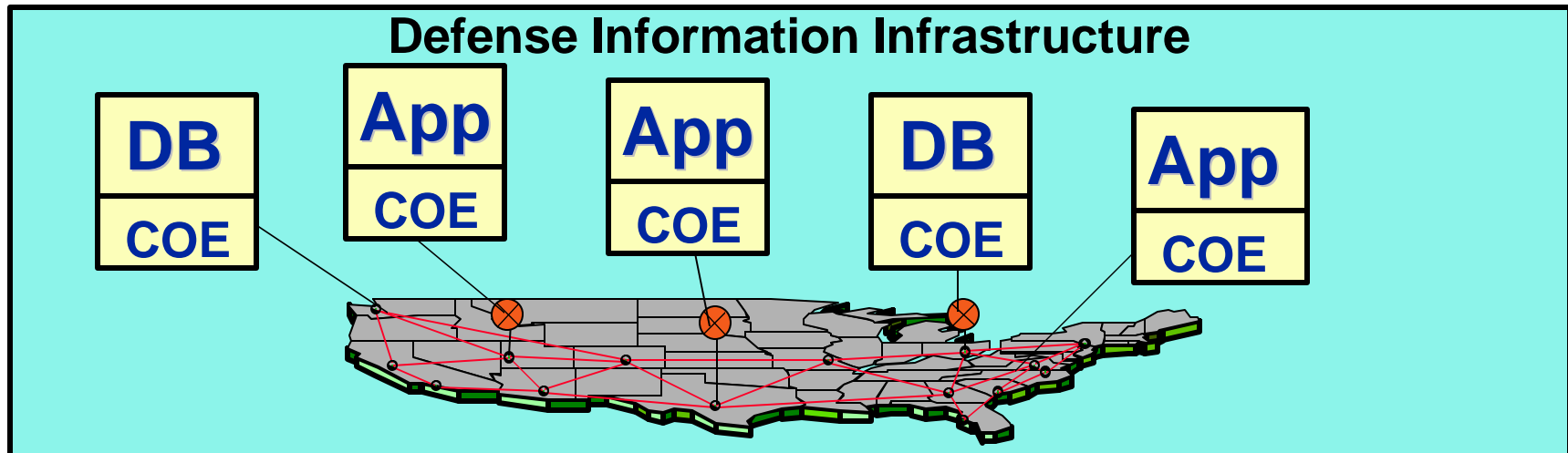
DII COE Overview

Defense Information Infrastructure

DII COE Relationship to the DII

The Defense Information Infrastructure (DII) is a seamless web of communications networks, computers, software, databases, applications, and other capabilities that meets the information processing and transport needs of DoD users in ...

The DII COE provides the common software infrastructure which the DII software, databases, and applications execute on.



Common Operating Environment
COE Objectives

Lower Cost

Early Integration

Commonality

Ease of Installation

Elimination of Redundancy

Standardization

Interoperability

Scalability/Portability

Security

Common Operating Environment

DII COE Components

Mission Applications

- Mission-unique functionality
- Developed by Services/Agencies
- Controlled & deployed by Services/Agencies

Common Support Applications

- Emphasizes interoperability via common view of data
- Functionality common within domains
- Developed by Services/Agencies
- Controlled & deployed by DISA

Common Infrastructure Services

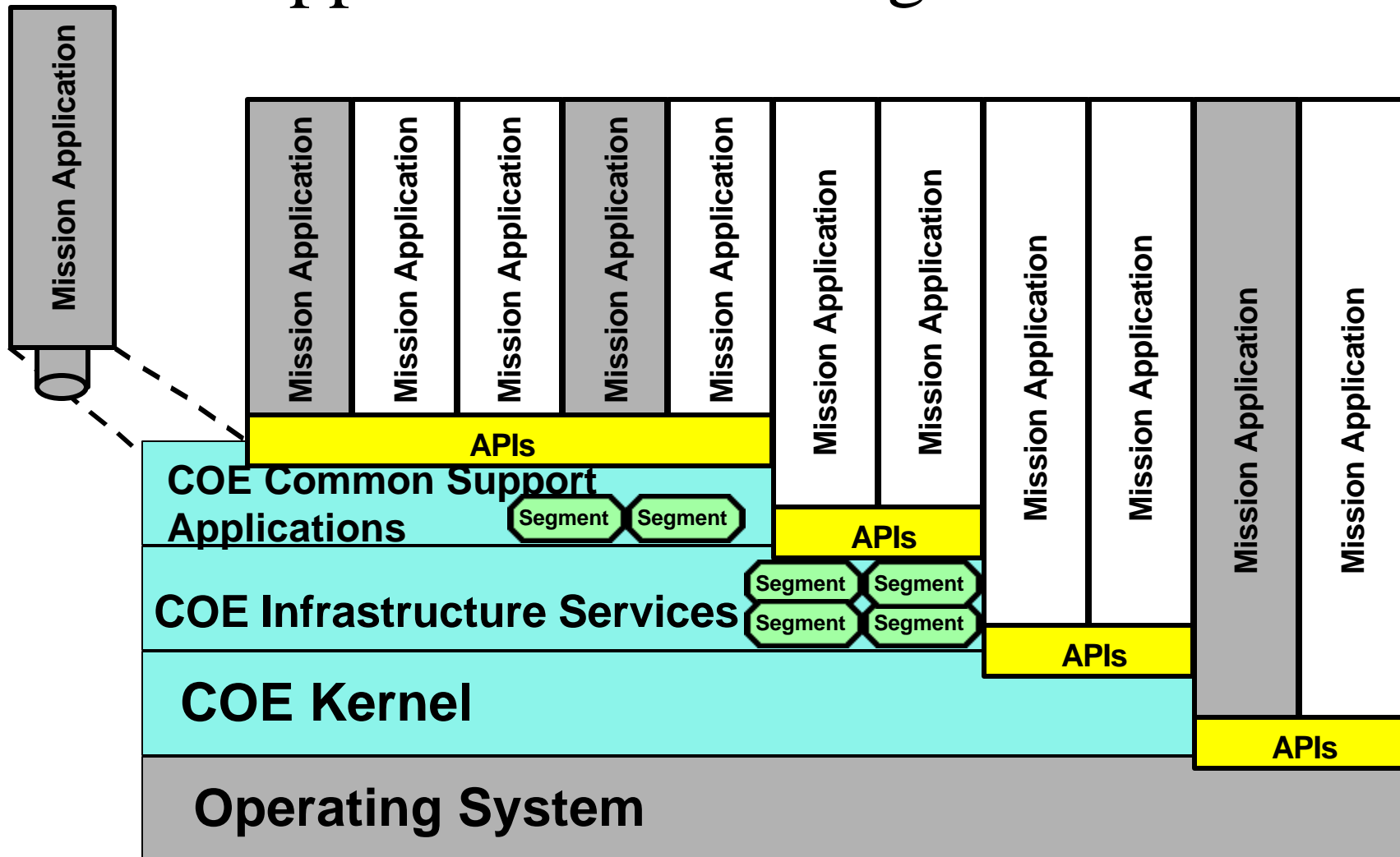
- Emphasizes movement of data through the network
- Functionality common across DoD
- Developed by Services/Agencies & DISA; Controlled & deployed by DISA

DII COE Kernel

- Present on every DII COE compliant workstation
- Developed by DISA
- Controlled & deployed by DISA

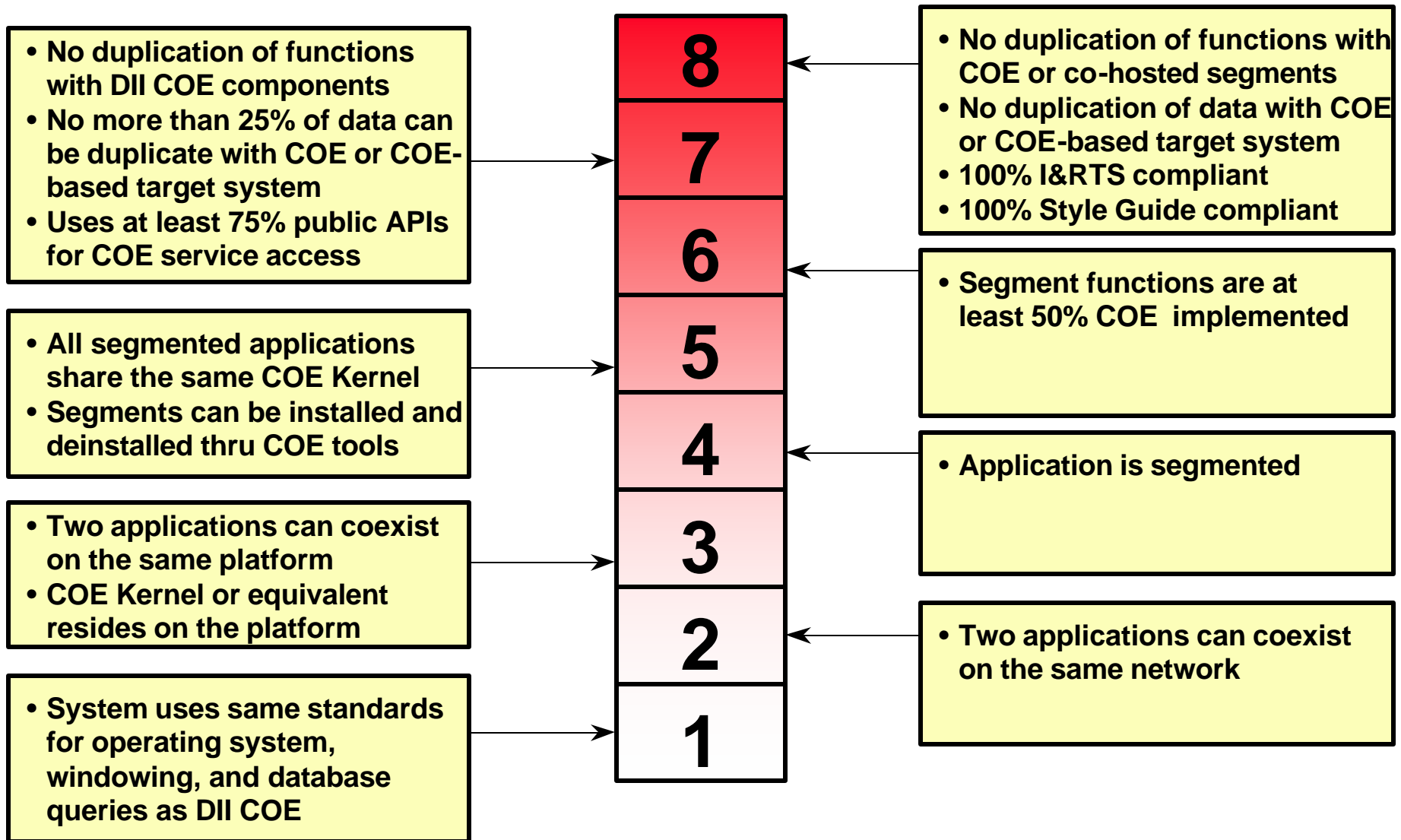
Common Operating Environment

Applications Running on the COE



DII COE Migration and Compliance

DII COE Runtime Compliance Overview



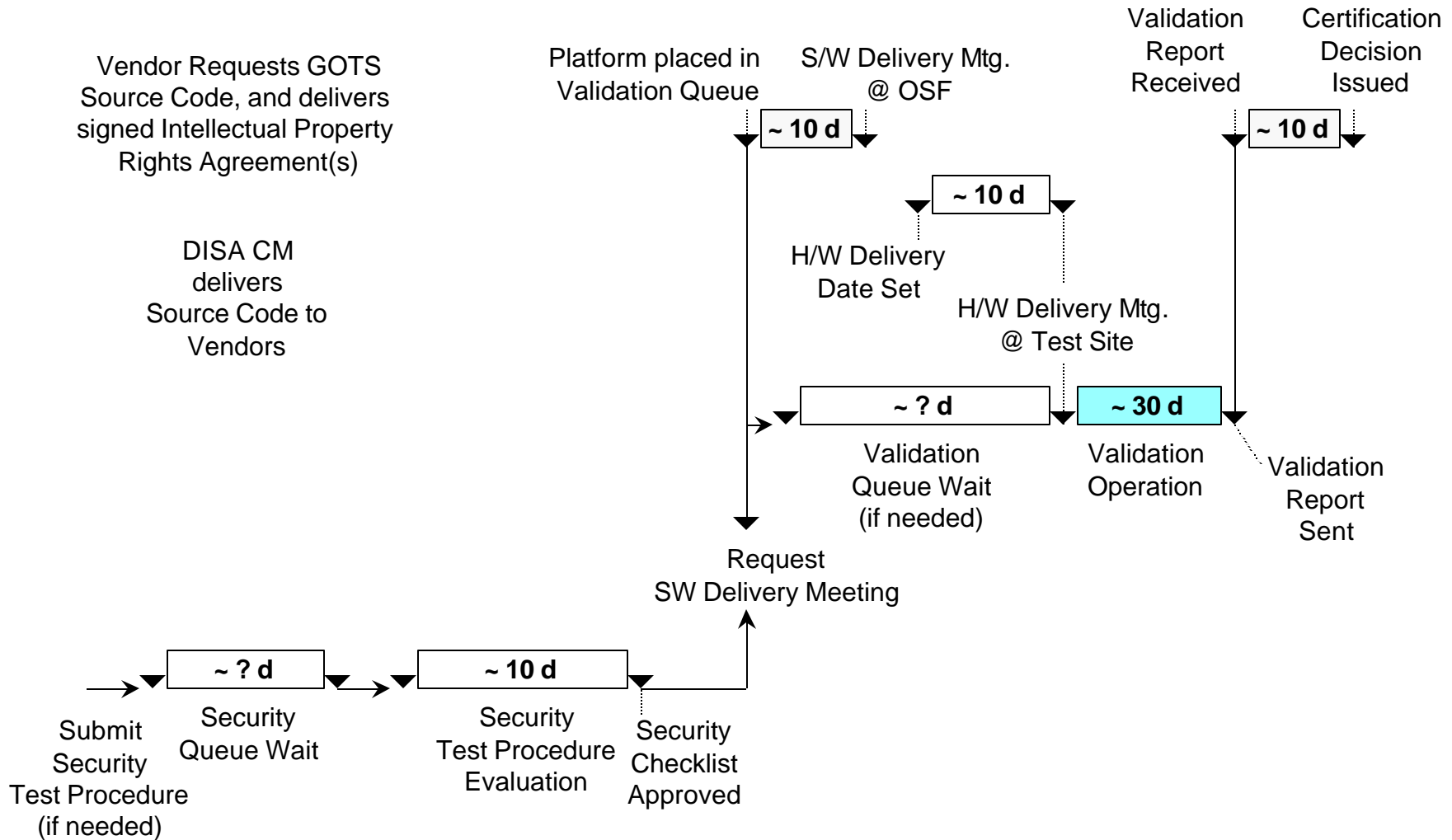
Kernel Platform Certification

Objective: Posix based Application Platform Compliance

The Kernel Platform Certification (KPC) Program provides:

- a broader set of operationally ready application Posix based platforms for key DOD systems ... while reducing cost.
- a clearly defined target platform for Posix development and migration of support and mission applications.
- an effective and equitable process to assist and encourage industry in providing DII COE functionality in their systems.

Certification Process

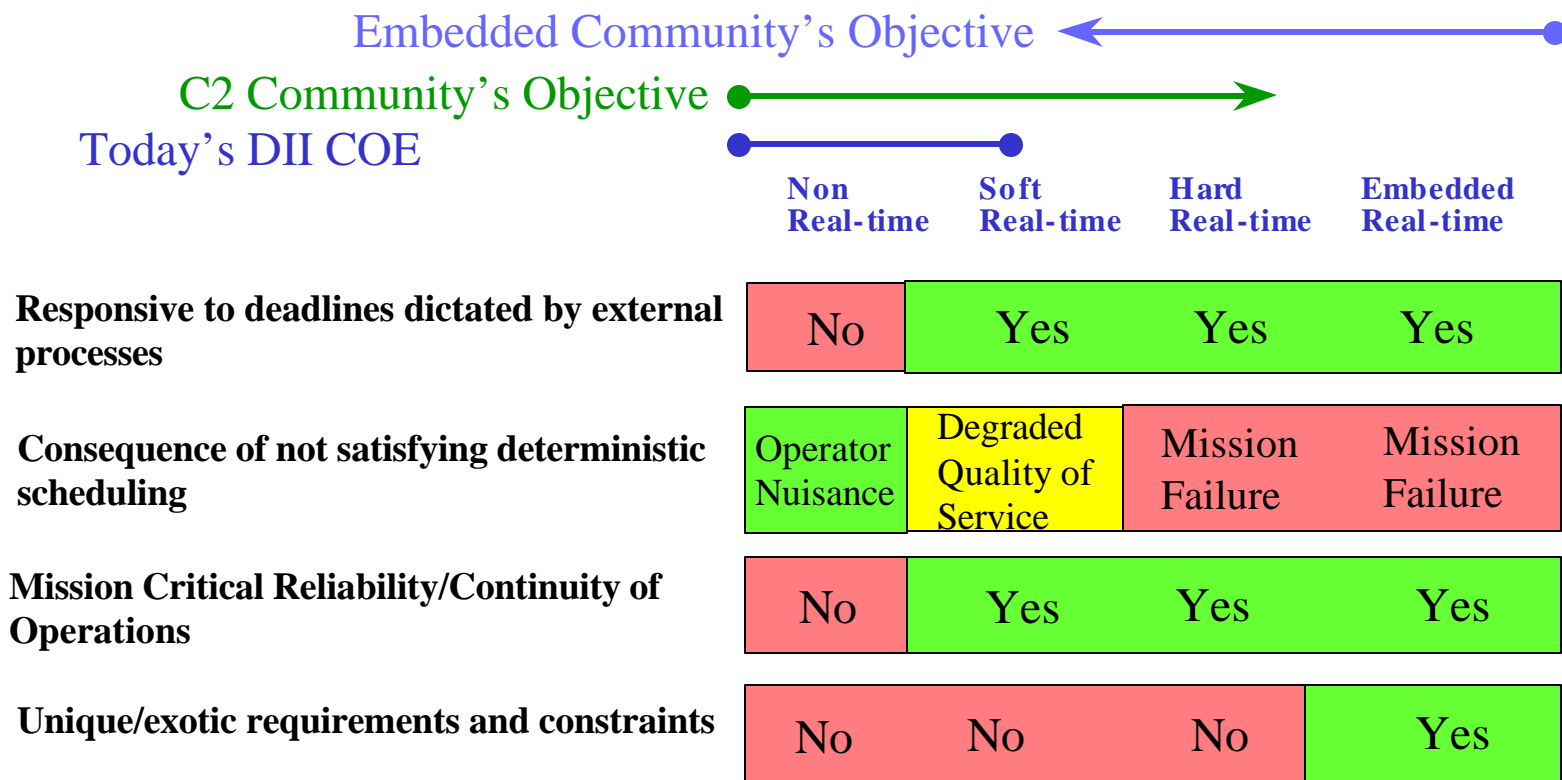


DII COE Real-Time Extensions

DII COE Real-Time Vision

- Foster/inspire the implementation of a DII COE configuration with support for real-time applications
 - Performs predictably with respect to time at useful levels of performance
 - Allows reuse of components across real-time systems
- Maintain strict compliance with DII COE standards and APIs
 - Extensions only when absolutely necessary and individually negotiated with DISA
 - Objective is to extend existing DII COE implementations, and not alternative/replacement DII COE

Characterizing Real-time Systems

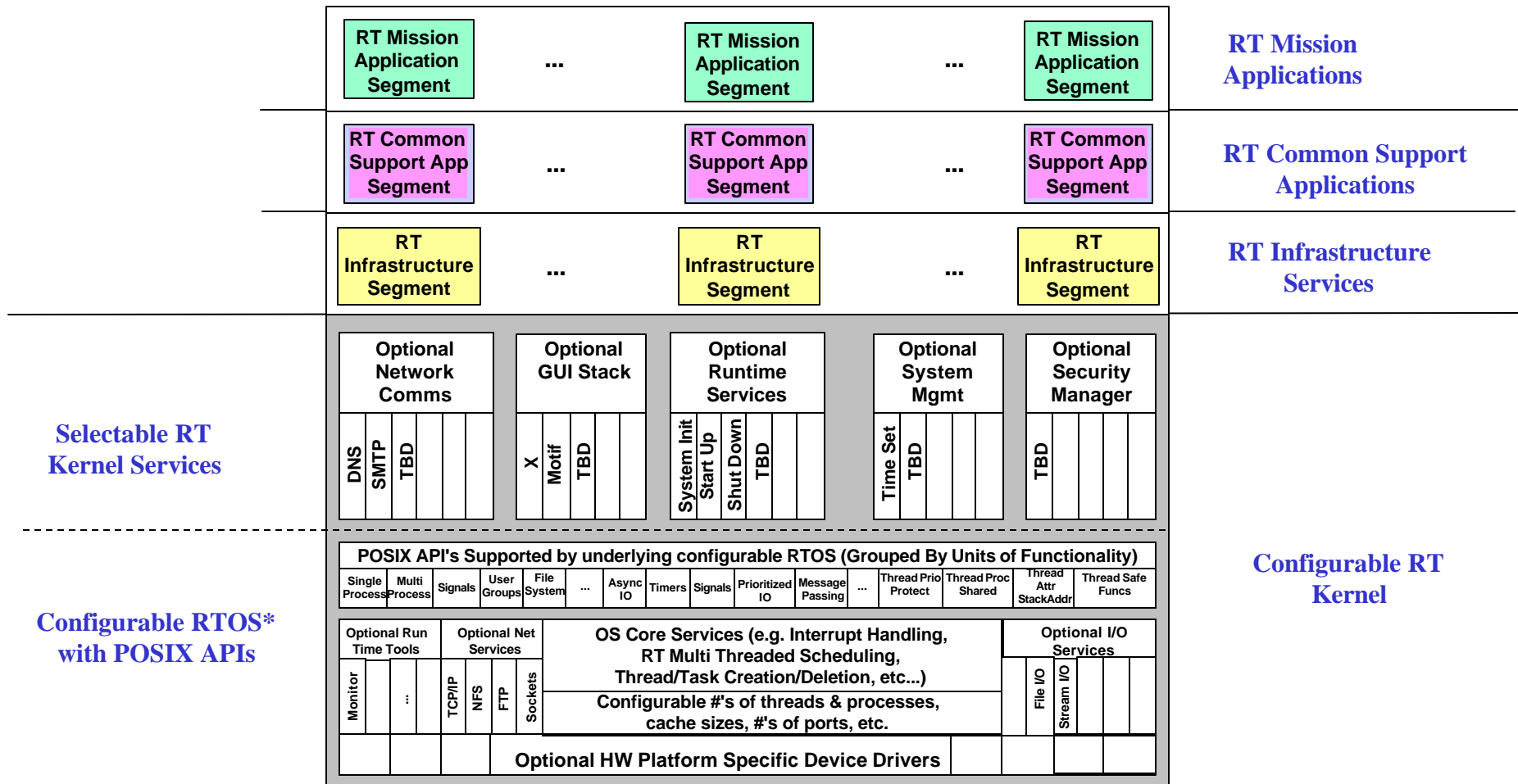


Note that the boundaries are qualitative and tend to overlap

RT DII COE Target Domain

Use of DII RT COE Features	Possible Target Systems
<ul style="list-style-type: none"> • Tactical/Strategic C2 Systems: <ul style="list-style-type: none"> – Real-Time infrastructure – RT extensions to DII COE capability – New RT capabilities 	<ul style="list-style-type: none"> – AWACS – Joint STARS – R/SAOC – Navy Common C&D – JTRS and JTT – NMD BMC3 – N/UWSS – Space Systems
<ul style="list-style-type: none"> • Real-Time C2 <ul style="list-style-type: none"> – Real-Time infrastructure – RT communications for situation awareness 	<ul style="list-style-type: none"> – Satellite Data Processing – Army EBC – ABL (hybrid) – JSF and other fighters
<ul style="list-style-type: none"> • Embedded Hardware Platforms - C2 <ul style="list-style-type: none"> – Operating System – Common portability interface layer (operating system wrapper) 	<ul style="list-style-type: none"> – AWACS Radar & ESM Upgrade – Crusader, Bradley, Grizzly – UEWR

Real-Time DII COE Reference Architecture

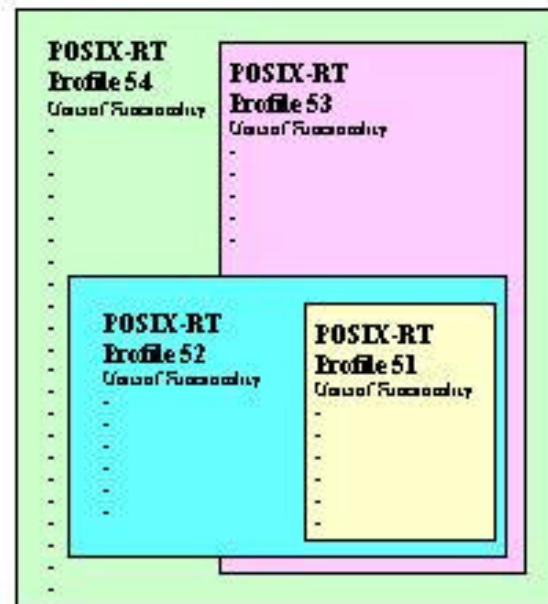


*Actual OS configuration depends on packaging options provided by RTOS vendor

Configurable/Selectable RT DII COE for Systems

POSIX Profiles

POSIX Approach to RT Kernel Configuration POSIX RT Profiles: Relationships



Profile 54: All POSIX.1,
Multi-process, Threads, File System

Profile 53:
Multi-process, Threads, No File System

Profile 52:
Single Process, Threads, File System

Profile 51:
Single Process, Threads; No File System

Portable Operating System Interface (POSIX)
IEEE Industry Standard
ISO 9001 Industry Standard

Allows Portability of Applications



Beyond POSIX...

Standardization Opportunities

- "Real-Time" flavors of commonly used network/distributed computing services, providing bounded time options and Quality of Service support
 - TCP
 - Sockets
 - Directory Services
 - Guards, Firewalls, etc
- Reliable and Unreliable network broadcast protocols
- Real-Time Security
 - Including provisions for Biometrics, Public Key
- Real-Time Data Management/Database Services
 - "I need the answer to this query in .03 seconds, or we'll be dead..."

Issues and Challenges

Products vs Standards: The Big Debate in DII COE

- Standards have not proven to be 'sufficient' for either portability or interoperability
 - Portability has not been 'absolute'
 - Common products provide much better interoperability
- DII COE generally organized around "products" called "segments"
 - Infrastructure segments are COTS products or Government-provided code
 - Mission application segments are normally developed by specific DoD programs
 - All segments ride on a "platform" of approved hardware/software
 - Intent is to provide 'plug and play' for constructing C2 systems
- Product-based approach strongly limits the choices for the system developer/integrator

Integration-time vs Run-time: When do we assemble systems?

- Current DII COE segments are installed at run-time on each target workstation/server
 - Facilitates workstation reconfiguration and maintenance
- RT systems in general do not support run-time configuration
 - Total system performance can be affected by new components
 - Clearly not practical for embedded systems
 - Not practical for systems with high assurance requirements (e.g. flight safety test requirements)
- One class of "RT extension" is tools to integrate segments on an 'integration environment', and then deploy full software load to target system
 - Useful for non-RT C2 systems, too

Specifying Real-Time Behavior

- Few examples exist on how to specify Real-Time behavior of software components and standards
 - Key issue is predictable behavior
 - Current approach in DII COE to provide 'Integration Notes' to pass information from segment developer to system integrator
- How should Real-Time behaviors be tested/validated?
- How to move QoS into standards, allowing QoS properties across COTS products and systems?
- What are the effects of 'composing' Real-Time systems from COTS/GOTS/custom software written by different developers?

Challenges for the Real-Time Open Systems Community

- Improvements in 'specification technology'
 - Standards that are more complete, less ambiguous, more interoperable
- Improvements in test and verification approaches
 - Open Source test suites?
- Improvements in specification of Real-Time properties
 - What should we specify? (How do we specify QoS?)
 - How should we measure?
- Understanding the impact of 'composition' on RT systems
 - Few systems written from scratch any more
 - But putting X and Y together often produces unintended results...
- Models for long-lived systems
 - In-the-field upgrade of software components