

Beyond Web Services







A network-centric approach to system design

David Crute
Principal Architect
Integrated Systems Division
General Dynamics C4 Systems

GENERAL DYNAMICS
C4 Systems



Network Services Evolution

Catch Phrase	Software Integration	Object Integration	Web Integration		Open-Ended Network Integration	
	The Network is the computer	Objects	Legacy to the Web	The Computer is the Network	Network of embedded things	Network of things
Scale	100s	1000s	1000000s	10000000s	100000000s	100000000s
When/ Peak	1984/ 1987	1990/ 1993	1996/ 1999	2001/ 2003	1998/ 2004	2004/ 2007
Leaf Protocol(s)	X	X	+HTTP (+JVM)	+XML, Portal	+RMI	Unknown
Directory(s)	NIS, NIS+	+CDS	+LDAP(*)	+UDDI	+Jini	+?
Session	RPC, XDR C / S	+CORBA C / S Objects	+CORBA, RMI Portable GUIs	+SOAP, XML Web Services	+RMI/Jini Network Appliances	+? Network Services
Schematic						

The Future

“...possibly the single-most transforming thing in our forces will not be a weapons system, but a set of interconnections and a substantially enhanced capability because of that awareness.”

-- Defense Secretary Rumsfeld

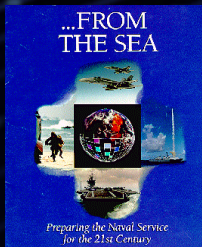
Current Trend Vs Current Capabilities

- Data Centric VS Network Centric
 - Data Centric is the current environment
 - Data Centric supports Situational Awareness
 - Traditional Client Server
 - Enables Web Services
 - Network Services can be deployed Today
 - Extends Legacy Systems Effectiveness
 - Provides Resources & Applications in addition to Data across the entire Network
 - Enables time-critical applications
 - Frameworks can let any system work in a System of Systems Environment
 - Empowers Web Services

The Vision for Change



Joint Vision 2020



Naval Power 21



Army Vision 2020



Marine Expeditionary
Maneuver Warfare



AF Vision 2020

Common Requirements

- Secure C4ISR Information Networks
- Global Access to Critical Data
- Information Interoperability
- Shared Awareness
- Collaborative Environment
- Support for Joint/Coalition Operations
- etc.

Requirements Point to the
need for a new architecture!

The New Imperatives

How to Improve Design of large, complex Net-centric capabilities?

How to Lower Risks inherent in designing and deploying large, complex Net-centric capabilities?

**Shape evolution of Enterprise
IT
Vice**

Net-centric Characteristics

– Heterogeneous

- Variety is essential and inevitable – basis for healthy evolutionary growth and survival within dynamic threat environment

– Parallel

- Multiple implementation and concurrent use of components and processes – increases speed and provides fail-over capability

– Market-driven

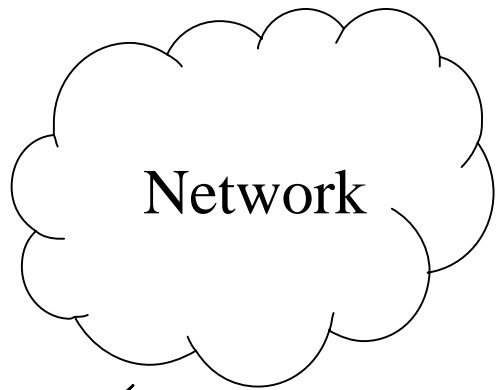
- Emphasizes Market principles vice top down direction to optimize – “Survival of the fit” (v. selection of the fittest from a single perspective)
- Developers “experiment early and often” to find the right niche

– Agile and adaptable

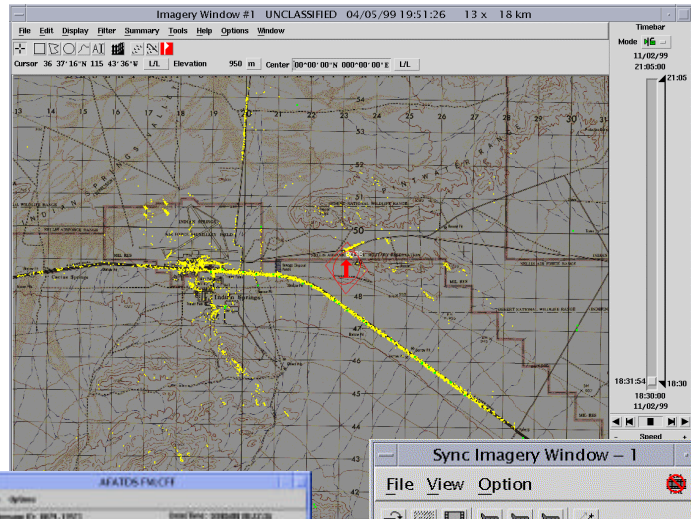
- Capable of rapid reconfiguration to meet new and unanticipated requirements or circumvent disruption
- Expedient task-oriented collaborations vice static bureaucracy

Vision of Networked Services

Sensor Systems



Command & Control System



Engagement Systems

Targeting Interface
Provided by Fire
Support Element

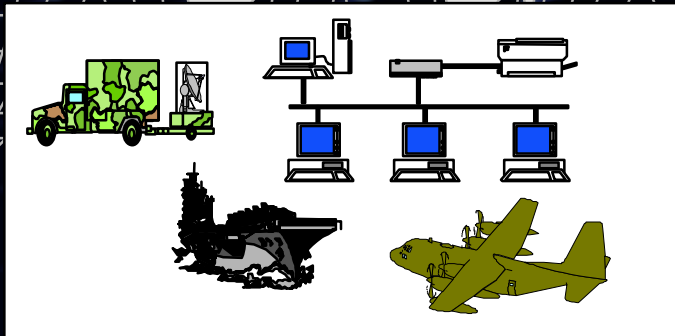
Viewing Application
Provided by UAV Sensor

OSD Vision for the Future

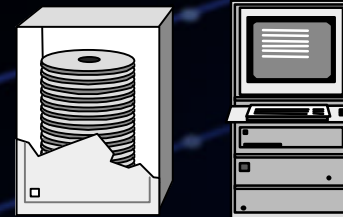
A Top Level Architecture Perspective



Global Perspective
A Network Centric Distributed
Multi-INT support System

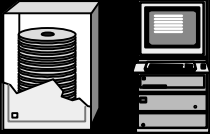


Element Perspective
A Deployable, Modular, Scaleable
Combat System

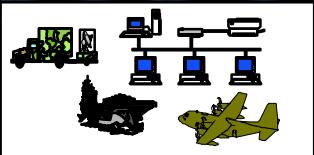


Component Perspective
Parts of the Elements
Basic architectural modules

Next Generation Architecture



- Network-Centric starts at the system design
 - Everything connected to the network
 - Plug-n-Play System Infrastructure

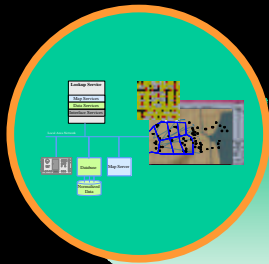


- Extend the concept to System-of-Systems
 - Plug-n-Play Distributed Systems
 - Shared Information Environment



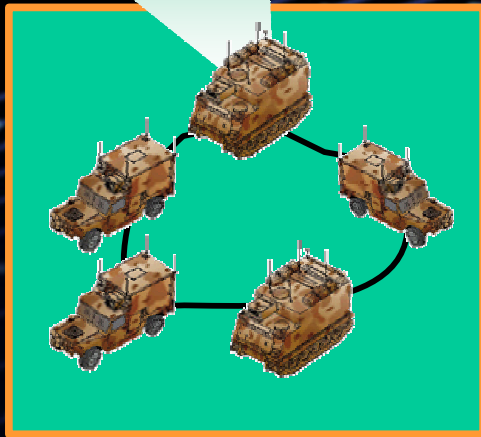
- Enable Global Reach
 - Support Discovery of services in a global environment

Scaleable, Network-Centric Approach

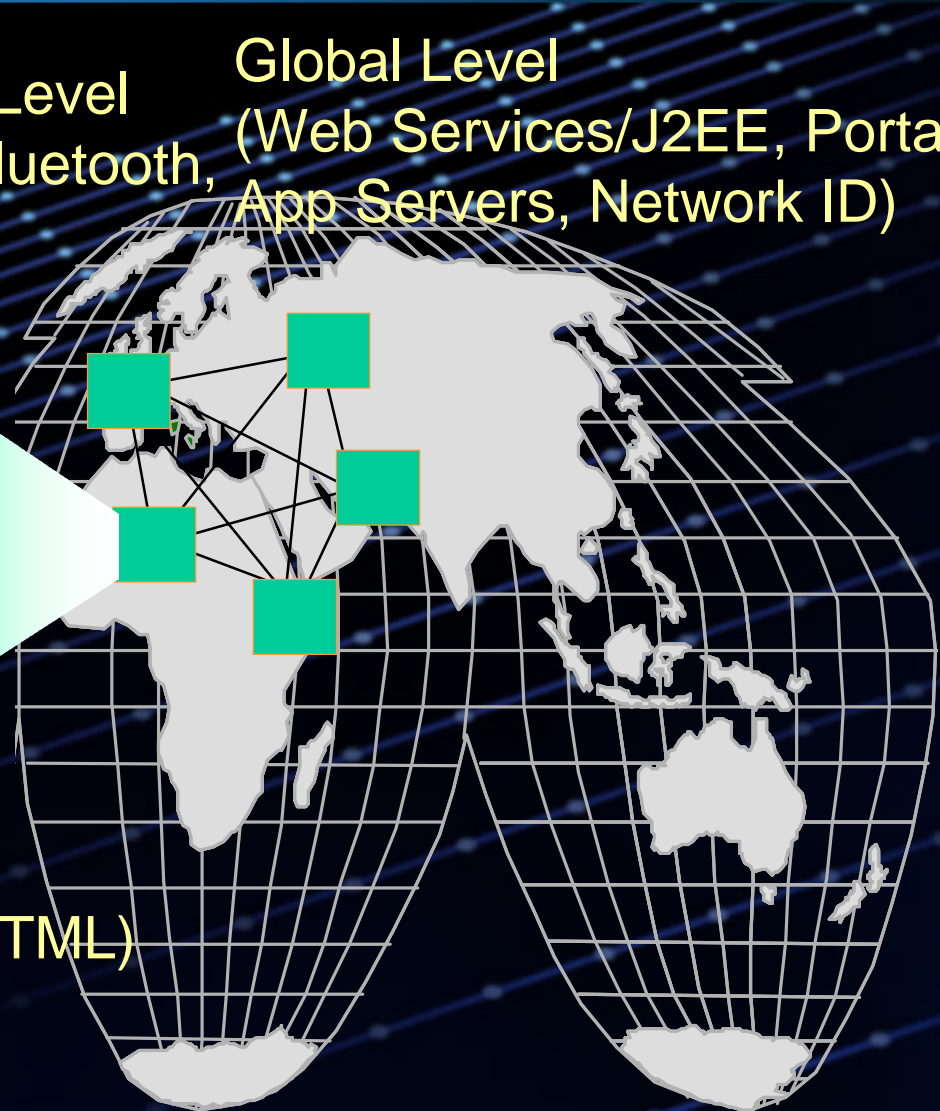


Component Level
(Jini, JXTA, Bluetooth,
IEEE 1394)

Global Level
(Web Services/J2EE, Portals,
App Servers, Network ID)



Element Level
(Java/Jini/JXTA/XML/HTML)



Foundation for Web Services

- Publish, Find, Use Services: **UDDI**
- Formal Service Descriptions: **WSDL**
- Service Interactions: **SOAP**
- Universal Data Format: **XML**
- Ubiquitous Communications: **Internet**

Simple, Open, Broad Industry Support

But are web services enough?

Web Services fit well here

Some web services technologies can be applied here

Not a good fit here

Enterprise Architecture
(10 Second Resolution)

System
Architecture
(Second Resolution)

Platform Architecture
(ms Resolution)

Limitations with web services

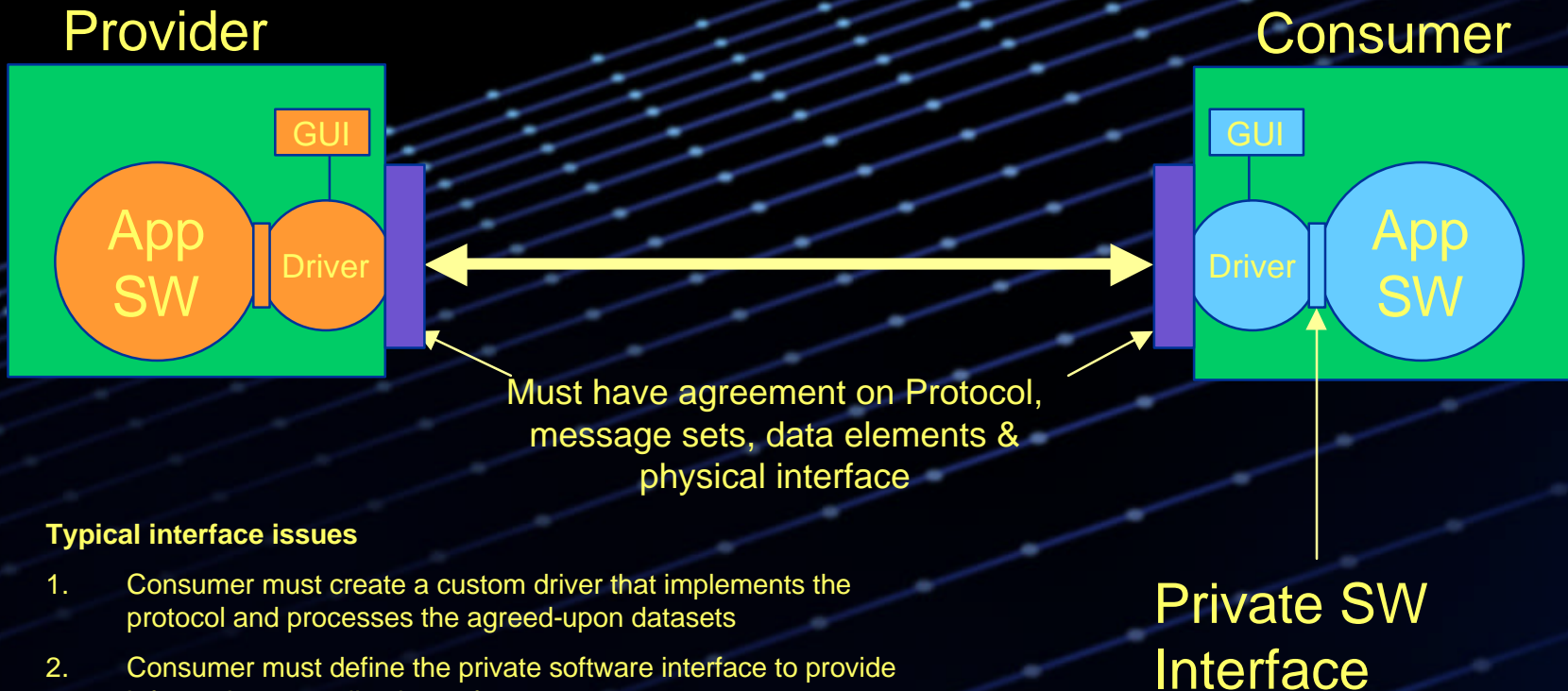
- Web Services are great for enterprise applications, but are of limited use in high-performance systems
 - **SOAP** protocol is not sufficient for time-critical interfaces
 - **UDDI** lacks notification mechanisms when services fail
 - **XML** processing adds significant overhead to network and processing

Network Services for System and Component Architectures

Net-Centric System Design

“Network Centric” design needs to move beyond sharing data in the enterprise and be applied to how we architect our systems from the ground up

An Example: Typical interface implementation



Typical interface issues

1. Consumer must create a custom driver that implements the protocol and processes the agreed-upon datasets
2. Consumer must define the private software interface to provide information to application software
3. Must implement software to establish and manage the connection (requires foreknowledge of parameters)
4. Requires significant interface testing
5. Typically must implement a new driver to handle any additional providers or consumers

Service implementation



Service-based approach

1. Software driver (proxy) is provided by the service provider
2. Underlying message set/protocol is hidden from the consumer system
3. Connection parameters are also hidden from the consumer – proxy knows how to connect to provider
4. Consumer can get service from anyone implementing the service interface

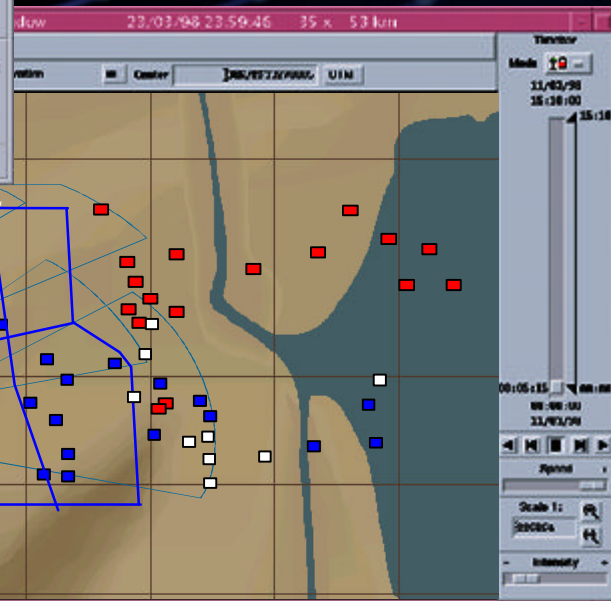
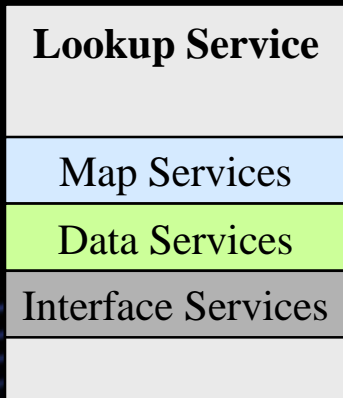
Application Services



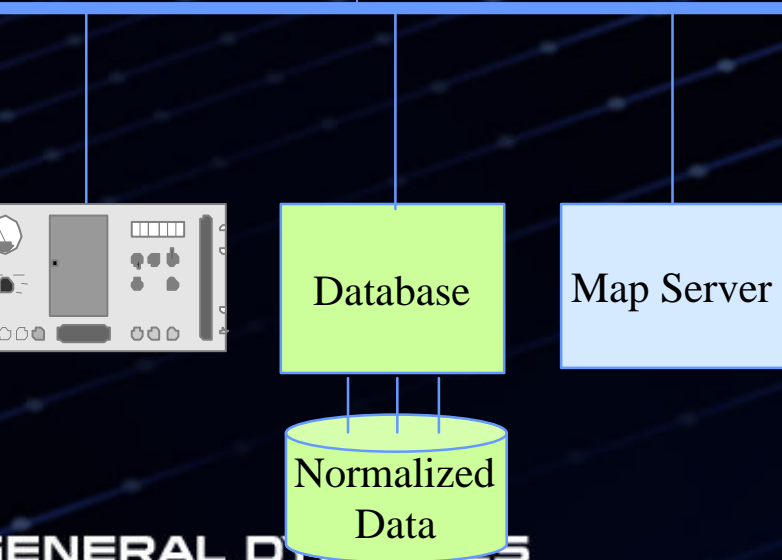
Service-based approach

1. Entire Software application is provided by the service provider
2. Underlying message set/protocol is hidden from the consumer system
3. Connection parameters are also hidden from the consumer – Application knows how to connect to provider
4. Consumer can get service from anyone implementing the service interface

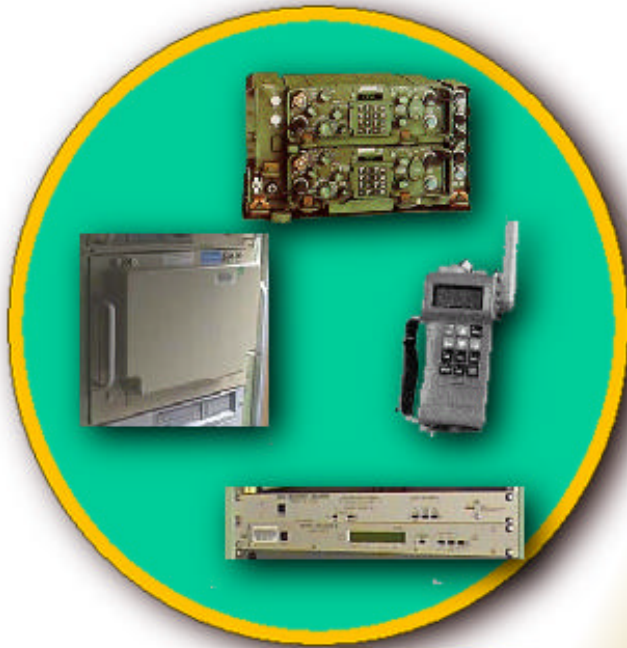
Ad-hoc Component Integration



Local Area Network



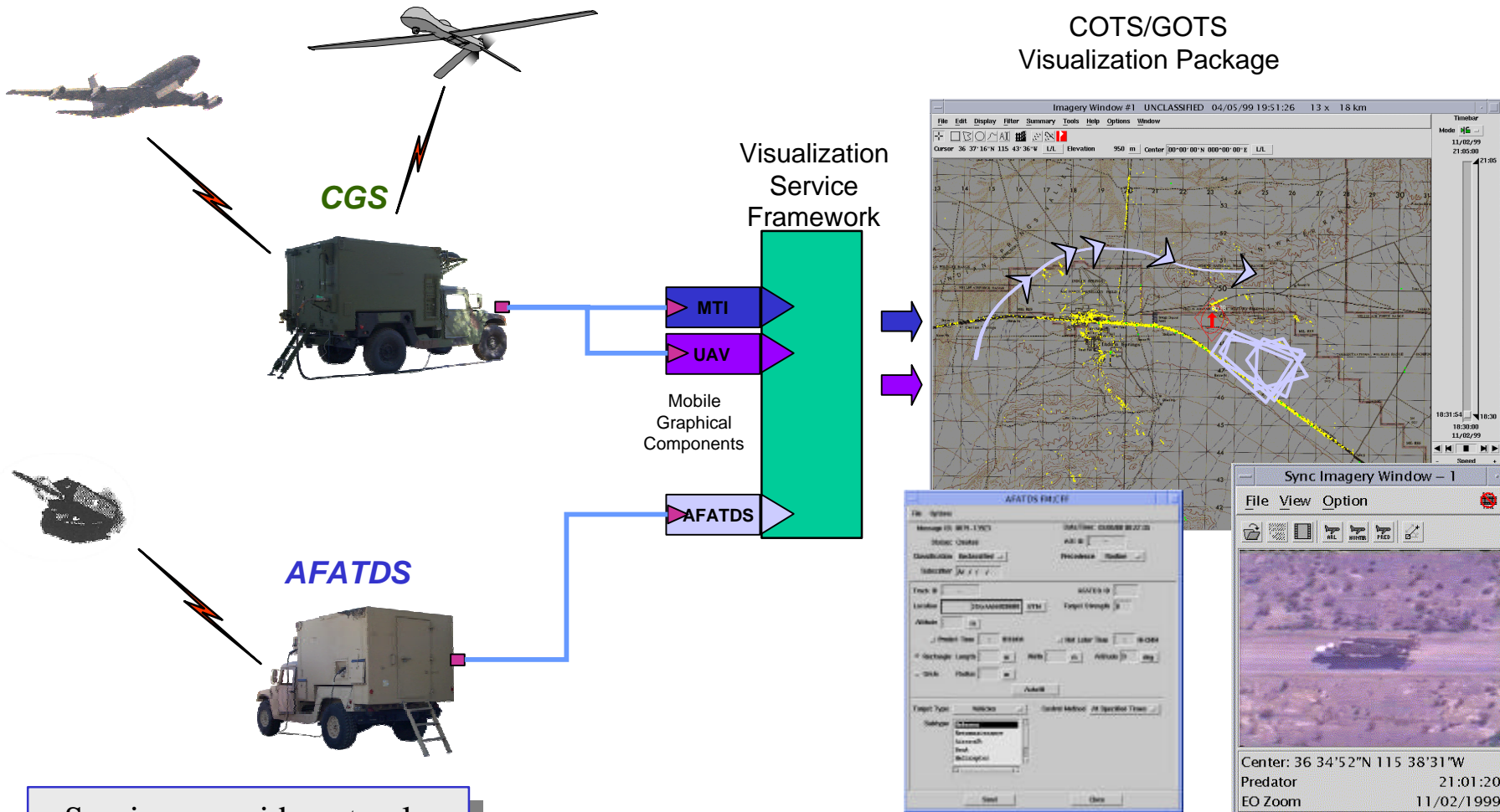
Goal: Plug-n-Play Systems Design



- Interface software packaged with the Hardware
- Hardware can be added with no software modifications



Distributed System Services

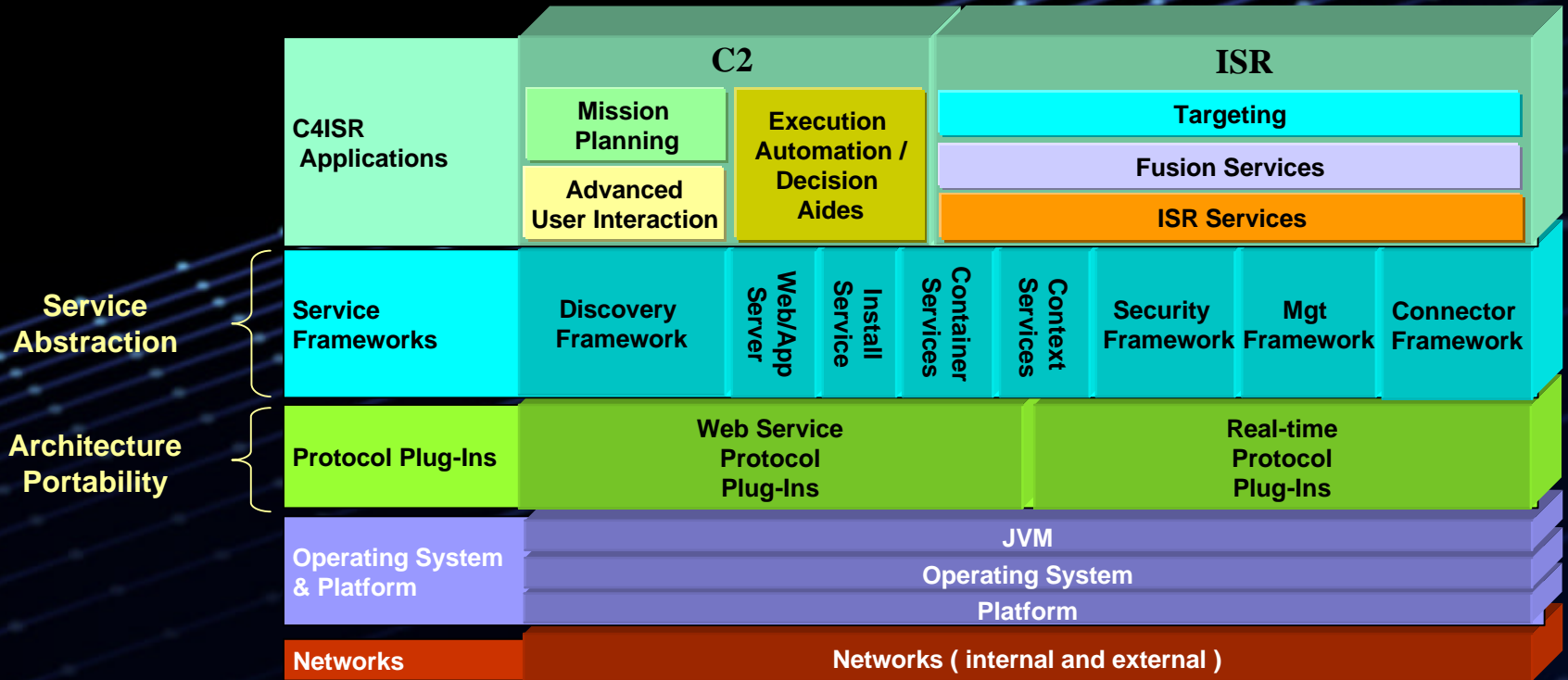


Services provide not only the access to data, but the “smarts” for how to deal with the data

Targeting Interface Provided by AFATDS

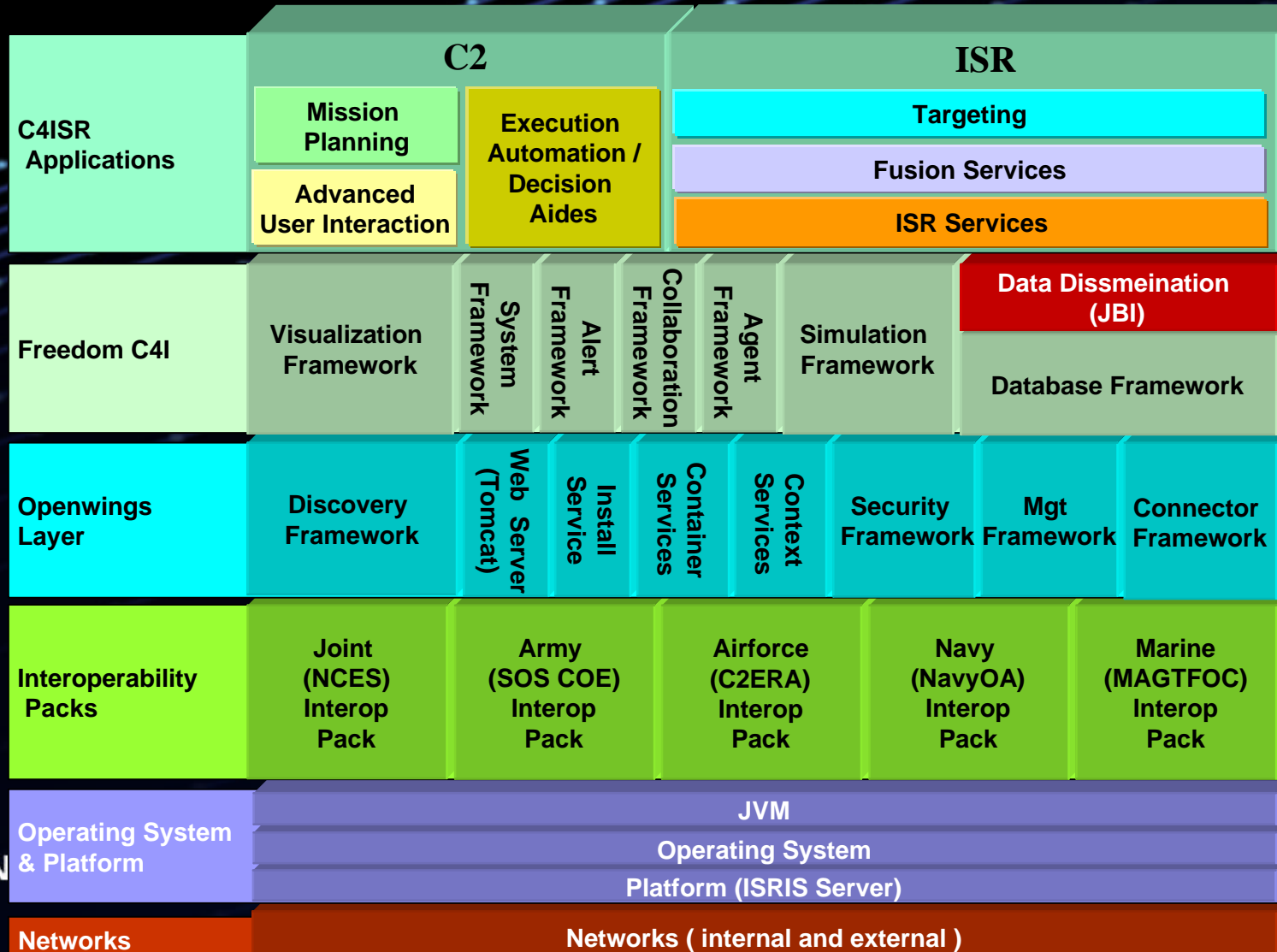
UAV Viewing App Provided by CGS

Interoperability Strategy



A common service abstraction allows services to be deployed in Enterprise, System or Platform architectures

Interoperability Strategy



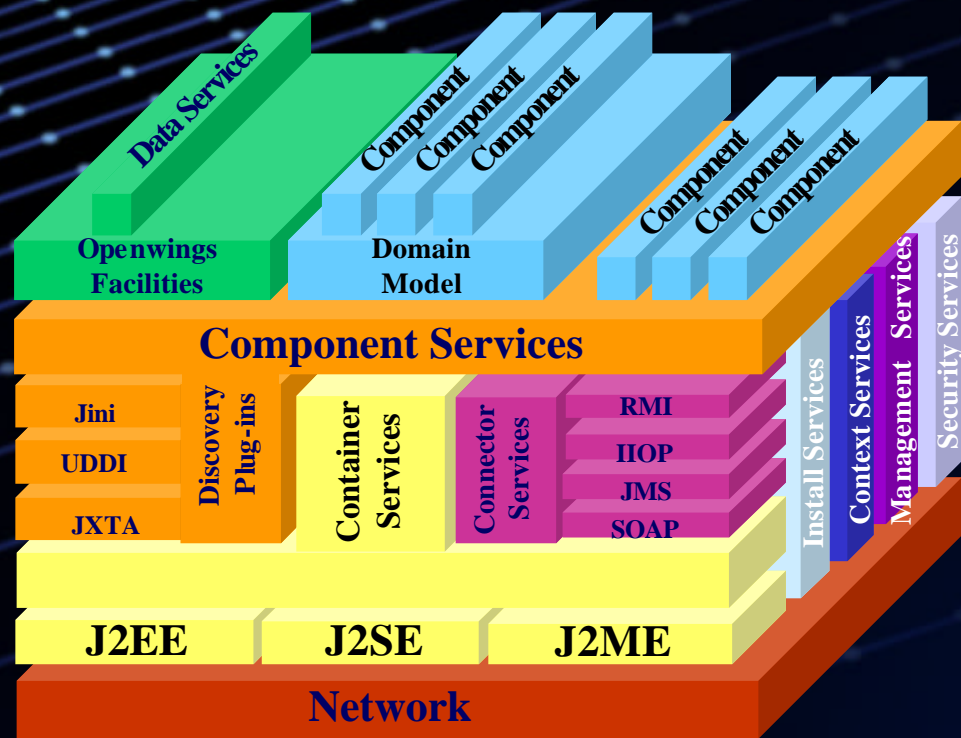
Service Abstraction - Openwings

- What is Openwings?
 - An open architecture forum established in 1999 by Sun Microsystems and General Dynamics
- Primary Goal
 - Provide a standard infrastructure that is developed by industry for time-critical, distributed, networked systems
- Openwings Community
 - 9 Expert teams, 100+ organizations, 350+ members

Reference version freely available to all via the Openwings web site
(www.openwings.org)

Openwings Architecture Model

- Openwings provides abstractions from platforms, discovery mechanisms and middleware technologies
- The Openwings framework also provides interfaces for security, management and installation services



Summary

- Web Services provide a model for enterprise integration, but aren't sufficient for time-critical applications
- Network-Centric System Design needs to extend to how we architect systems from the ground up.
- There is a need for a single architecture framework that bridges both enterprise and component architectures – Openwings provides a good foundation to build from

OPENWINGS



www.openwings.org

Questions?