PEER-TO-PEER DISCOVERY: A KEY TO ENABLING ROBUST, INTEROPERABLE C2 ARCHITECTURES

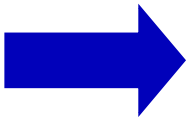
June 2005

Ray Prouty, NCW Chief Engineer
Kurt Kalbus, Discovery Project Lead
Dave Heddle, Chief Engineer

Laura Lee, Director C2 Systems
SPARTA, Inc.
13400 Sabre Springs Pkwy, Ste 220
San Diego, CA 92128
(858) 668-3570
laura.lee@sparta.com

Troy Crites, President, Mission Systems Sector
SPARTA, Inc.
1911 North Fort Myer Drive, Suite 1100
Arlington, VA 22209
(703) 558-0036
troy.crites@sparta.com

Outline

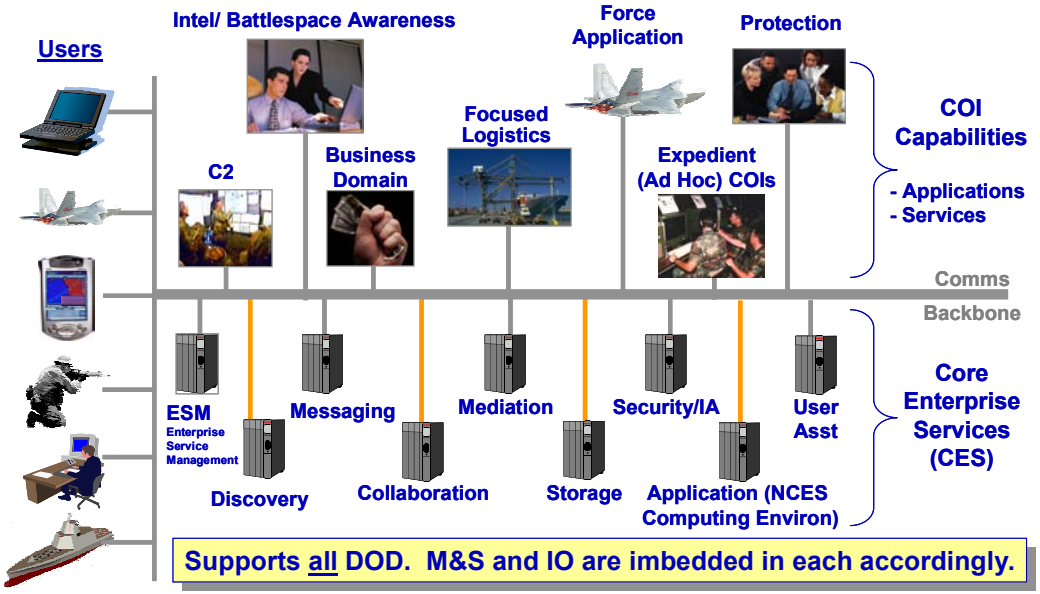


- NESTOR Project
- Discovery Background
- Non-Homogeneous Peer-to-Peer Discovery
- Summary

NESTOR: SPARTA IRAD Project

- **What is NESTOR…**
  - **N**et-Centric **E**nvironment for **S**ystem **T**esting and **O**perational **R**esearch
  - SPARTA's Distributed Testbed for the Design, Implementation and Quantitative Evaluation of
    - » NCW Concepts
    - » NCW Infrastructure
      - SPARTA's scale-model of the GIG

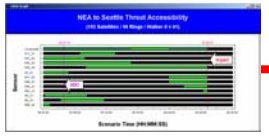
GIG Functional Capabilities: (CES) and Communities of Interest (COI)

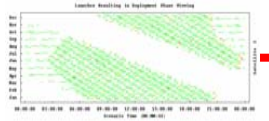


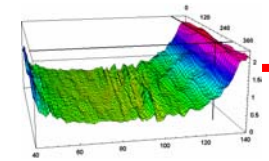
- **What we want to accomplish…**
  - Develop a Core Group of NCW Expertise Within SPARTA
  - Focused on SPARTA's Niche Areas (System Architecture, System Engineering/Integration, M&S, Information Assurance)
    - » Demonstrate Capabilities to NCW Customers and Potential Teammates

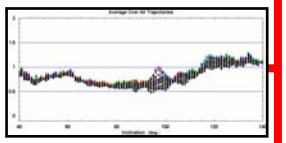
NESTOR Tasks

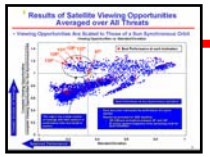
Metrics

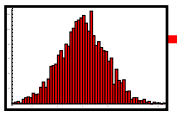




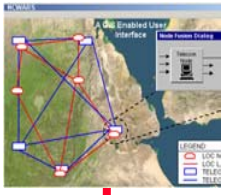








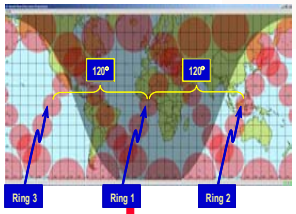
NEMO: Effects-Based Ops



C4ISR Arch, Testing, Training for Army FCS: OneSAF Testbed

NCW End-to-End (NETE) Model

Sensors: CERA



Mission Planning: JEDA

NESTOR COIs

Applications

NESTOR Backbone

Web-Services

Messaging: JMS

Discovery

Collaboration

Agents

GIS Feature Data (ERTHA)

Security/IA

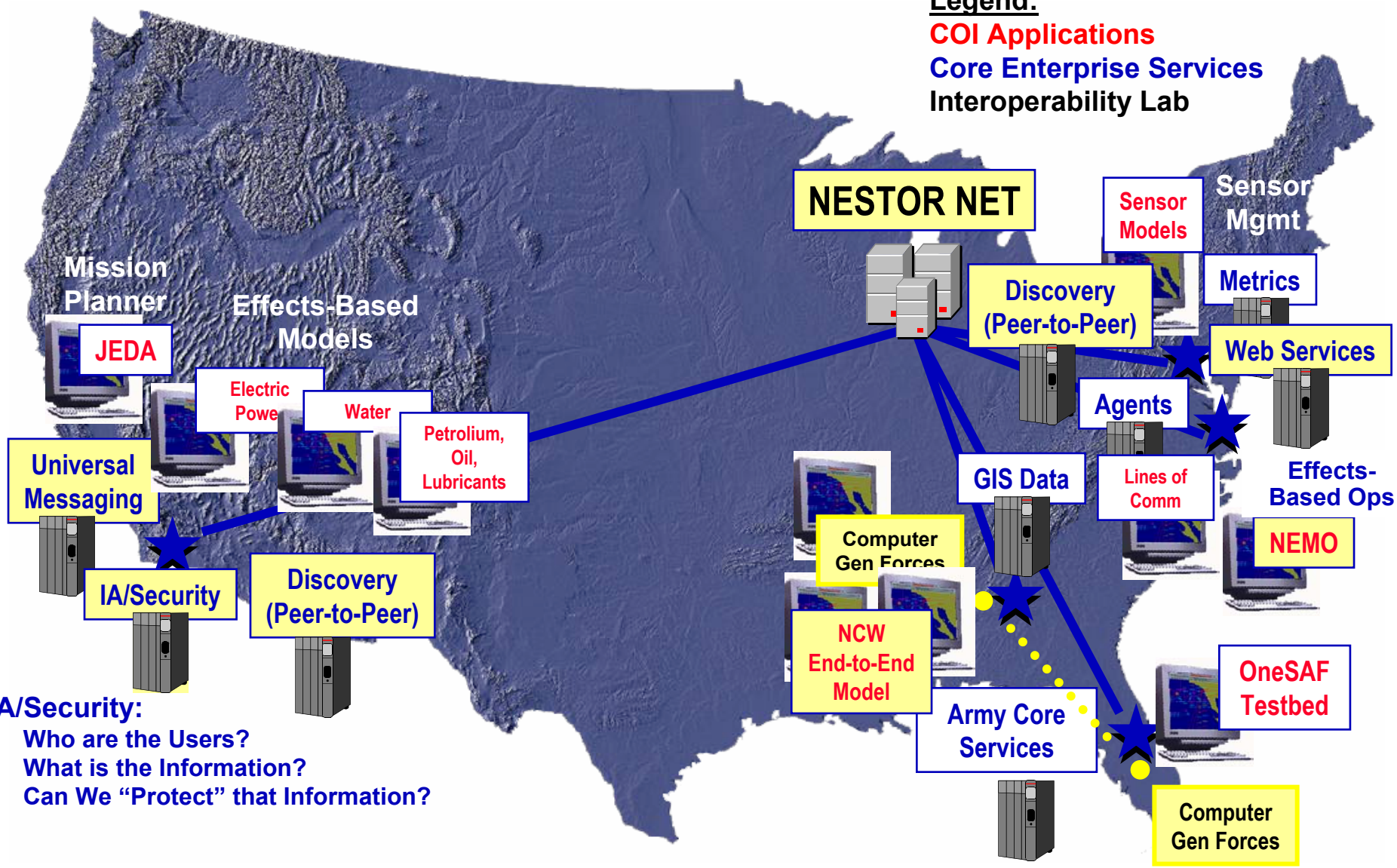
Metrics

NESTOR CES

Infrastructure

This Briefing Describes the Discovery Task

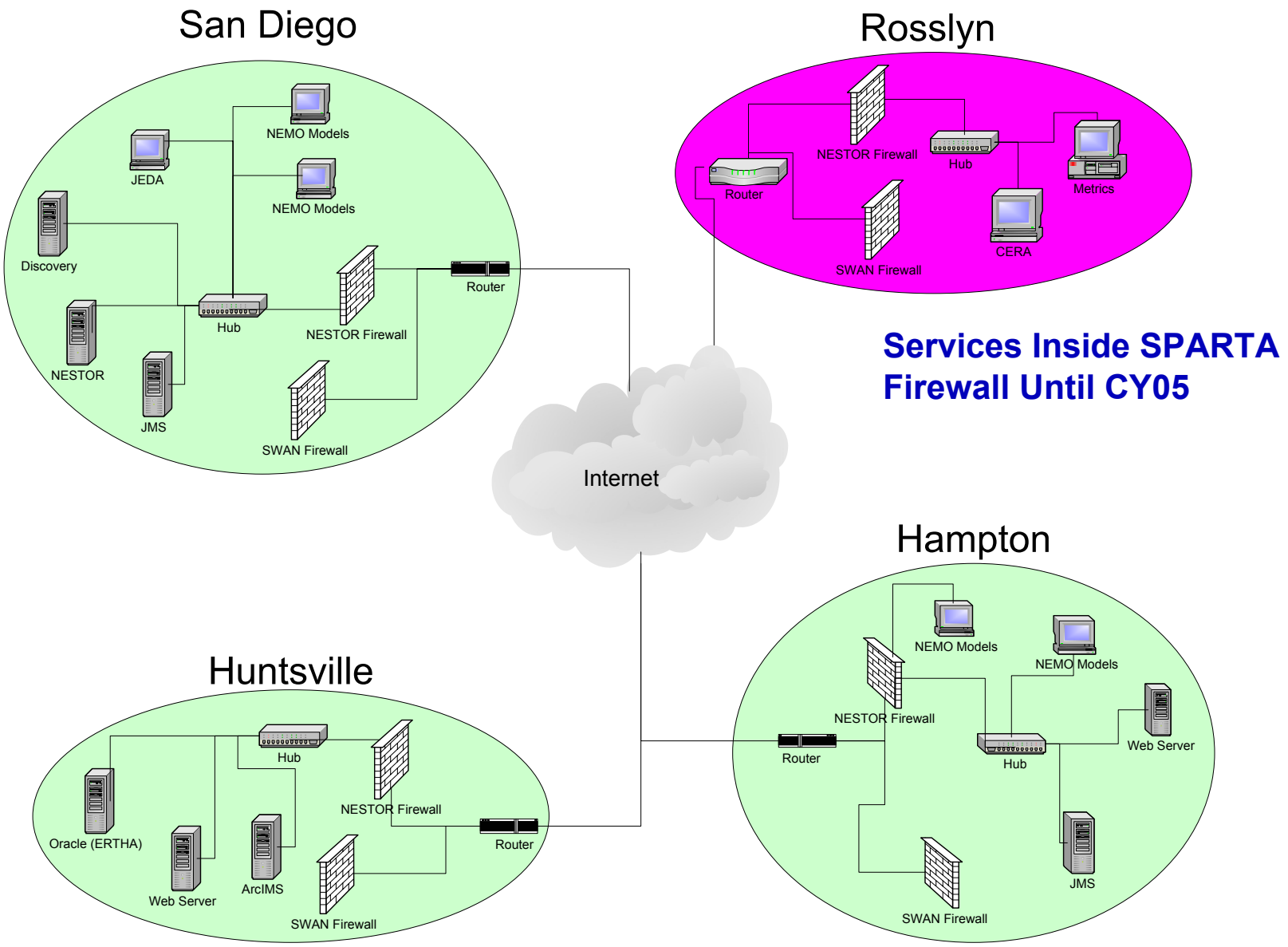
NESTOR Physical Architecture



IA/Security:

- Who are the Users?
- What is the Information?
- Can We "Protect" that Information?

NESTOR Firewall Architecture



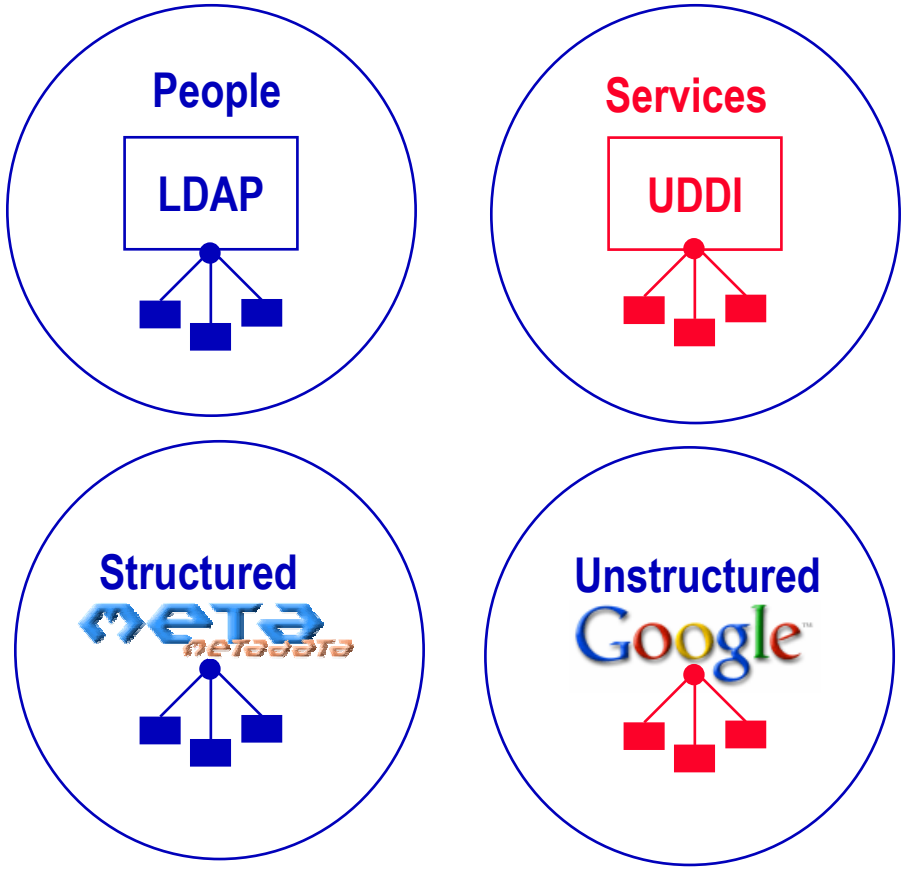
Outline

- NESTOR Project
- Discovery Background
- Non-Homogeneous Peer-to-Peer Discovery
- Summary

Discovery Background

- **The GIG is A Service Oriented Architecture (SOA) with Discovery as a Key enabler**
  - Central to converting legacy applications from stovepipes to services
  - Enables Runtime Integration and Self assembling applications (e.g., Find a Track Service)
  - Critical for Ad Hoc COIs
- **Common Approach to Discovery**
  - Static Design for Four Types of Information (i.e., People, Services, Structured and Unstructured Data)
  - Homogenous (Monolithic) Design
  - Centralized Servers

Four types of Discovery: Services, People, Structured (Metadata), Unstructured



LDAP: Lightweight Directory Access Protocol
UDDI: Universal Description Discovery and Integration

Using UDDI Registries in the GIG

- **Subscription / Notification is critical for the efficiency and reliability of applications**
  - Services on the GIG will be very dynamic.
  - Without Notification, applications will be unaware if a Web Service they are using has been removed or changed
- **Subscription / Notification as defined is meant for Business to Business transactions.**
  - Notification via Email or Web service
  - Delivery is not guaranteed
- **V3 standard has been approved (Feb 05)**
  - More like DNS (Domain Naming Service)
    - » Real names (SPARTA.COM vs AF239-234F-AED1-449C)
    - » Distributed Queries
  - All of the major UDDI vendors have announced support

Reference: www.uddi.org

Comments on Current GIG Approach

- **Benefits of Approach:**
  - Well Known
  - Centralized Security
    - » Requires Centralized User Management (Problem for Scalability)
  - Interoperability Not an Issue
  - Brute Force Scalability

- **Disadvantages of Approach:**
  - UDDI is designed for Semi-Static B2B applications
    - » Dynamic nature of GIG services is not handled well
  - Complicated mess of API's, different versions and associated software
    - » Requires four separate "browsers"
    - » Multi-Vendor Queries are problematic
  - Rigid Architecture
    - » Hard to support ad hoc COIs, Cross-Service or Agency Areas or Coalition Participation
  - No standard for how to do a "Federated Query"
  - Centralized Servers susceptible to outages
  - Health/Status has been moved to the "Management" Service
    - » Makes finding a service is awkward

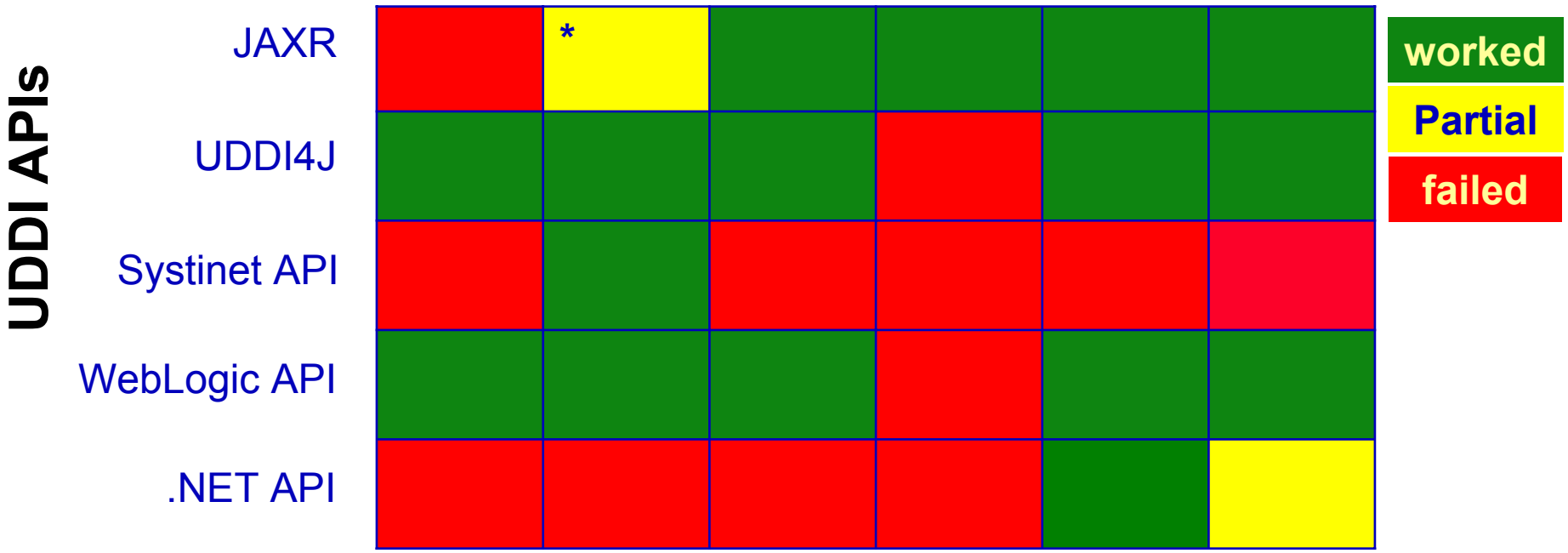
Discovery Approach

- **Examine the Gap Between Vision and reality**
  - Vision Attributes: Broad & Dynamic Customers
  - Reality: Capabilities Planned Today
- **Work with DISA to Understand "Proposed" technologies**
  - Gained useful insights into DISA's thinking
  - Relevant to October and Next Fest Activities
  - Used DISA's approach to discovery as starting point
- **Focus on Universal Description Discovery and Integration (UDDI) Servers**
  - DISA standard for Discovery of Services
  - Least Mature Today
  - Can Leverage COTS, GOTS and Freeware

SPARTA Discovery Research: UDDI Server/API Summary

** The current Systinet V5 Server can partially operate with JAXR. It will not access all V5 features, such as Subscription/Notification. To get those features, you must use the vendor provided API.*

UDDI Server Registries



Successful Discovery Requires Applications to Either Handle All UDDI Server Vendors or Mandate That All Applications Use a Single Vendor

Or Provide an Approach that Allows Applications to Discover Services Across Multiple UDDI Server Registry Vendors (e.g., Peer-to-Peer)

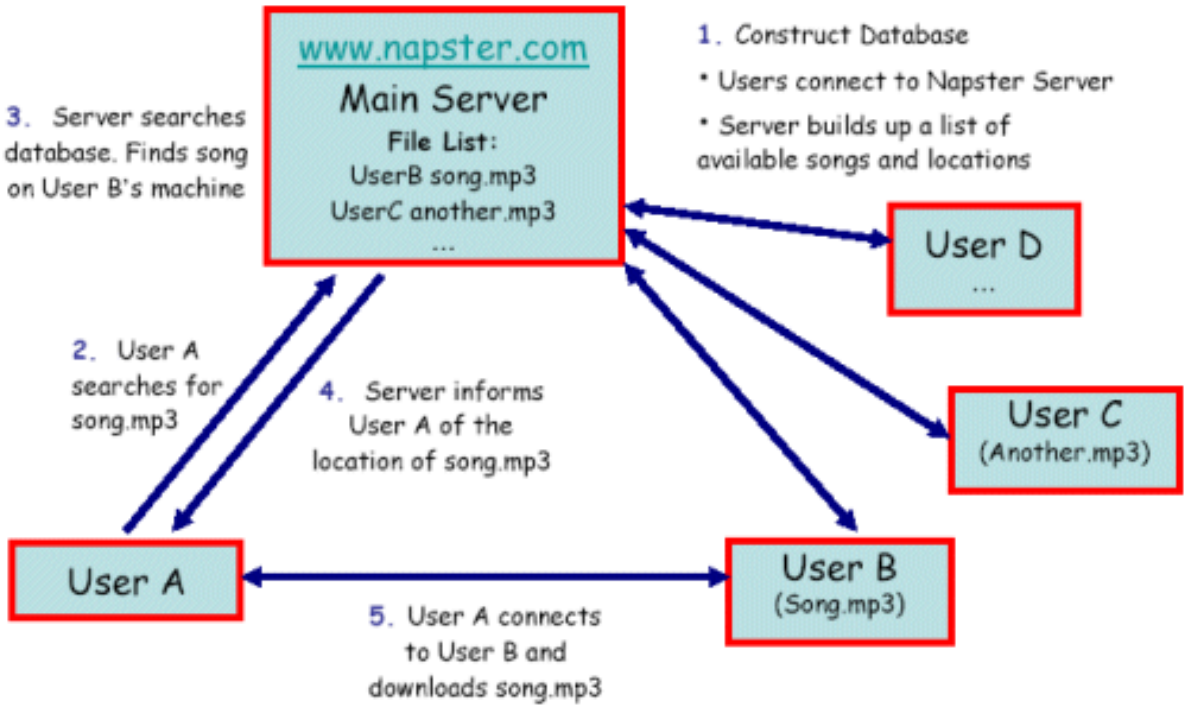
Peer to Peer Topologies

- Brokered peer to peer is a P2P network in which an index of content is maintained at a centralized server, but the actual content is kept at the peers (e.g., NAPSTER).

- Decentralized P2P is a network of peers that are all "equal". There is no centralized service at all except for possibly a bootstrap initial peer discovery mechanism (e.g., Gnutella).

- Semi-centralized P2P consists of "super-nodes" which are organized in a decentralized P2P network, but each super-node has weaker nodes reporting to it and the content of each weaker node is periodically uploaded to the super-node (e.g., LimeWire).

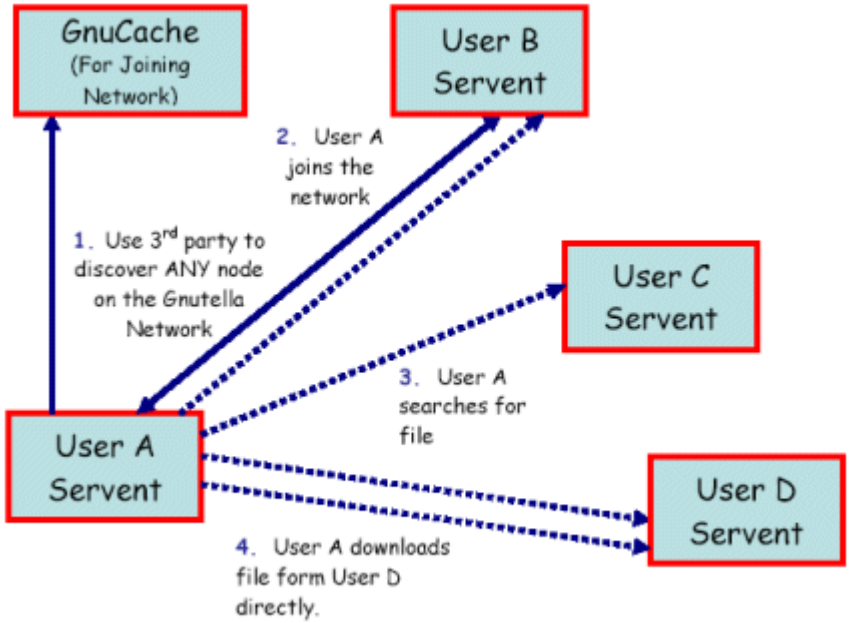
Brokered P2P



Napster is an example of a brokered P2P network.

The disadvantage of such a system is that if the Main Server goes down, the entire network fails.

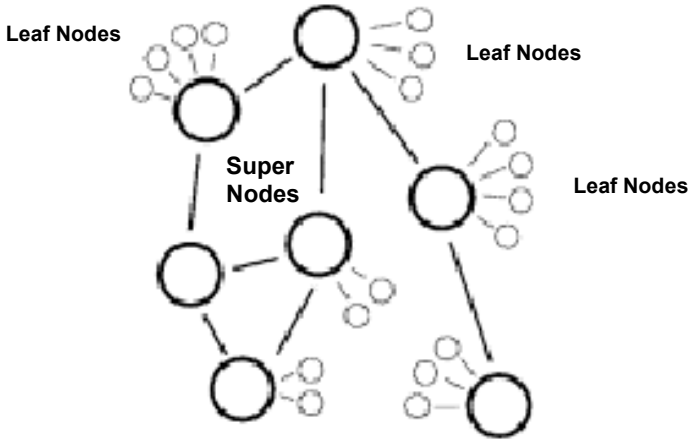
Decentralized P2P



The Gnutella model is fully decentralized

The disadvantage is that all peers are treated equally and are part of the queries causing bandwidth utilization (across all links, even small ones).

Semi-centralized P2P



The Gnutella Protocol uses a semi-centralized structure (e.g., LimeWire) in which the "super-nodes" form a decentralized P2P network.

In actual practice, fully decentralized P2P networks will usually self organize into a semi-centralized P2P network.

This is the model that will be used for the NESTOR P2P Discovery network.

Pros and Cons of P2P models

P2P Technology	Pro's	Con's
Brokered (Centralized) e.g., Napster	Easiest to implement, Inherent "Mandated" interoperability	Single point of failure, Rigid (Hard to Adapt to Ad-Hoc COIs)
De-centralized e.g., Gnutella	Robust/Reliable Network, Vendor Neutral for Discovery Servers	All peers are treated equally possibly leading to bandwidth problems
Semi-Centralized e.g., Limewire	Compromise Solution (Uses super nodes to solve bandwidth problems)	Increases Firewall complexity

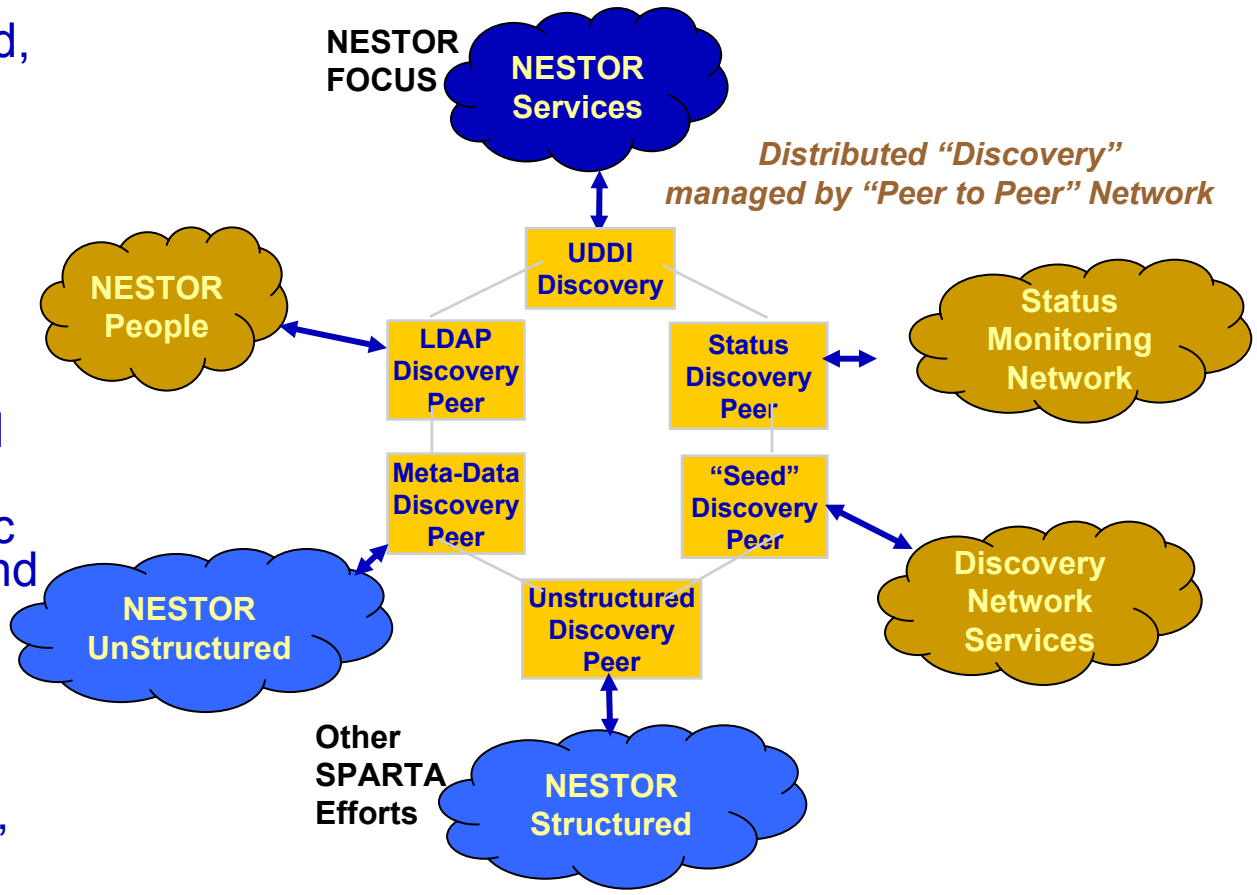
Outline

- NESTOR Project
- Discovery Background
- Non-Homogeneous Peer-to-Peer Discovery
- Summary

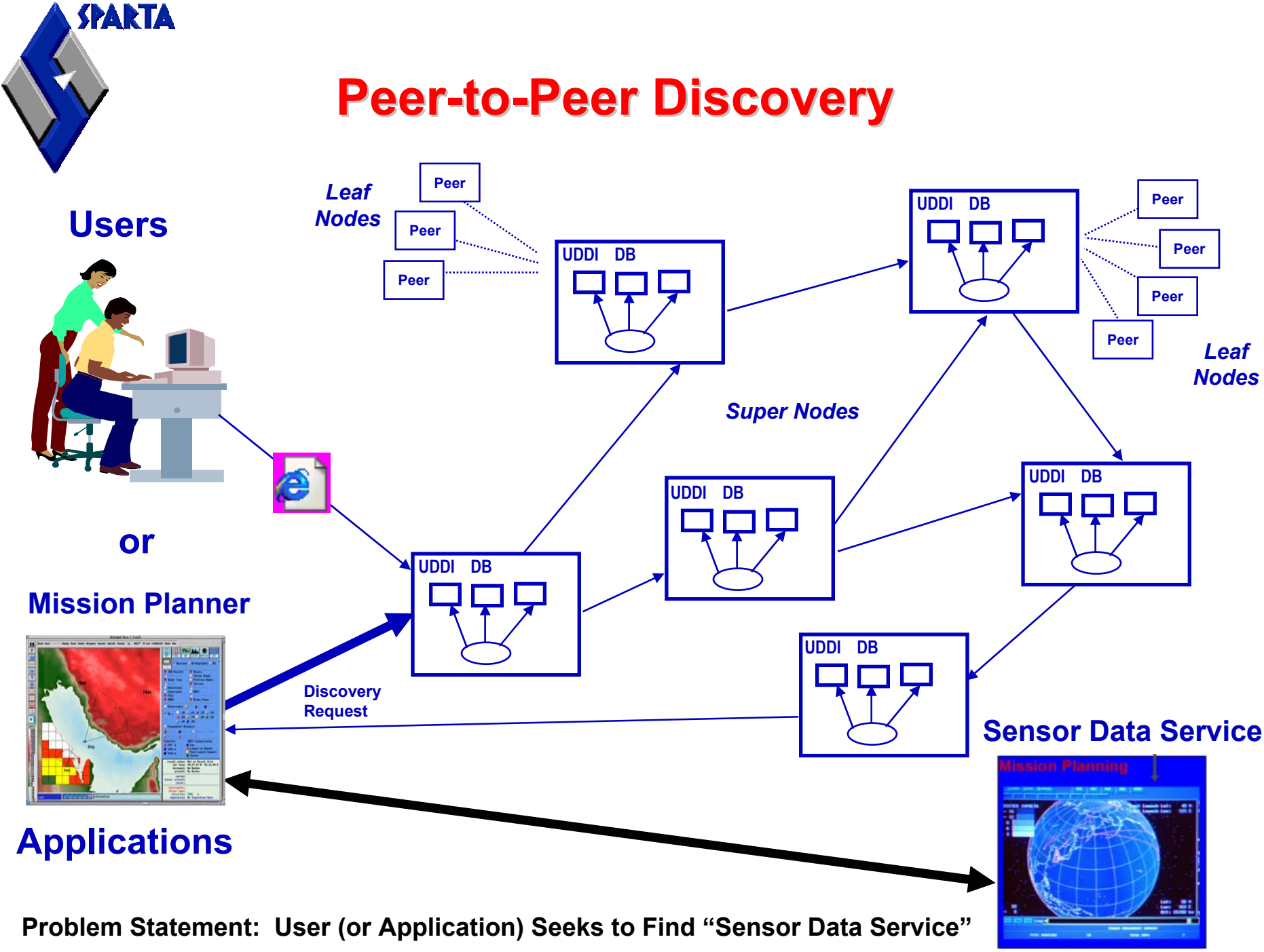
NESTOR Approach to Discovery

- **We mimic the four DISA "Discovery" services**
  - Structured, Unstructured, People, Services
- **Enhanced with**
  - Unified "browser"
  - Peer To Peer (P2P) technology
    - » Allows distributed/federated queries
    - » Handles the dynamic nature of services and data
- **We are concentrating on the "Services" area now**
  - UDDI (least understood, most problematic)
  - Three other Discovery areas are COTS

Persistent resources to be registered with UDDI

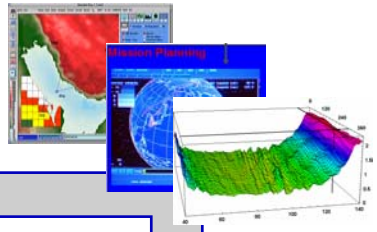


Peer-to-Peer Discovery



Problem Statement: User (or Application) Seeks to Find "Sensor Data Service"

Discovery Example Proof of Concept



WebLogic UDDI
- Missile Web Service
- Cera Web Service
- Metrics Web Service

WL UDDI API | JAXR UDDI API

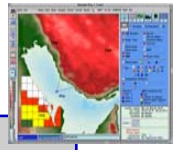
Discovery Peer

Chakotay – 157.185.24.253 (San Diego)

Firewall

Internet

Firewall



Systinet UDDI

Missile Web Service

WL UDDI API | JAXR UDDI API

Discovery Peer

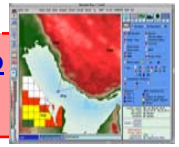
Kirk – 157.185.52.20 (Hampton)

Sun UDDI
Missile Web Service

Potato – 157.185.24.29 (San Diego)

Firewall

Sun UDDI on Potato
Missile Web Service

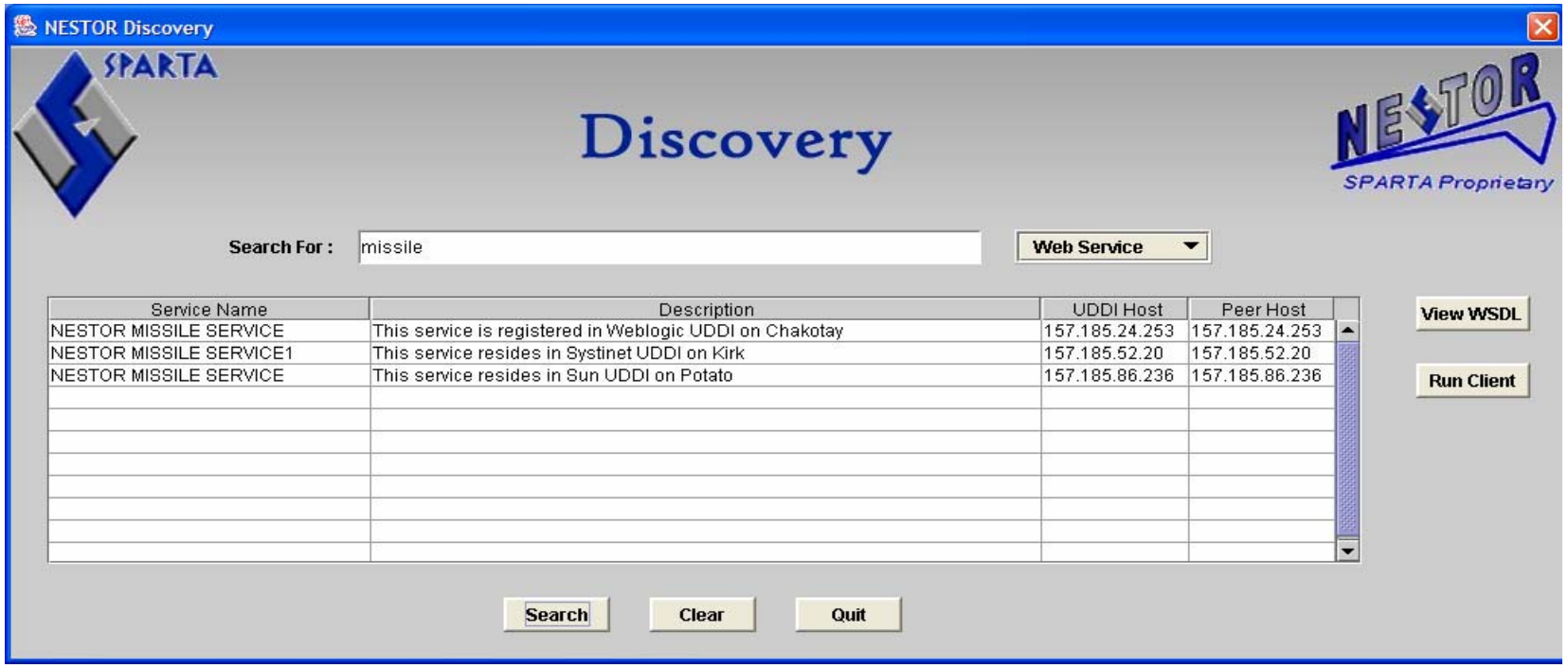


WL UDDI API | JAXR UDDI API

Discovery Peer

Watergate – 157.185.86.236 (Rosslyn, VA)

Discovery “Demo”



1) The above query application is running on Potato. Initially Potato only knows about Chakotay - so its sends the "missile" discovery request to Chakotay, which queries its WebLogic UDDI registry and sends a hit back to Potato (line 1 above).

2) Chakotay, however, knows about the Kirk and Watergate peers - so it successfully forwards the requests to those machines. They each query their UDDI registries get hits that they send directly back to Potato (line 2,3 above).

3) As an additional example of flexibility, the Discover Peer on Watergate does not actually query a UDDI registry on Watergate - but queries a UDDI registry running on a different machine (Potato).

4) The above query ends up accessing three different vendors of UDDI registry - WebLogic, Systinet and Sun.

5) The four machines in the above scenario reside behind 3 different firewalls.

Summary

- **We Understand Discovery Approaches and Have Researched Key Ideas**
  - GIG Operational Needs
  - Industry (Vendor) Capabilities and Standards
  - Underlying Advantages and Disadvantages
    - » Flexibility
    - » Security
    - » Interoperability
- **We Developed a Flexible, Robust Peer-to-Peer Concept that Can Assist the GIG Discovery**
  - Bridge Between CoIs (e.g., embodied in DISA/NCES and DGCS Efforts)
  - Foundation for Cross-Service and Allied Capability