Workshop #2

connected vehicle Core System Architecture/Requirements

September 20-22, 2011

San Jose, CA

Day 1

Systems Engineering Team

Topics

- Welcome, Program Overview
- Introduction – Needs, Architecture, Requirements
- Architecture Viewpoints Discussions
  - Enterprise
  - Functional
  - Connectivity
  - Communications
  - Information
- Core System Deployment
- Core System Risks
- Next Steps

Agenda – Tuesday 9/20

9:00	Welcome & Introduction
9:30	Core System Background & Overview
10:15	Break
10:30	Core System Needs Evolution
11:00	Core System Architecture Framework
12:00	Lunch
1:15	Core System Requirements Overview
2:15	Break
2:30	Architecture, Enterprise Views Discussion
4:30	Adjourn for the day

Agenda – Wednesday 9/21

9:00	Welcome & Recap
9:30	Architecture, Functional Views Discussion
10:15	Break
10:30	Architecture, Functional Views Discussion
12:00	Lunch
1:15	Architecture, Functional Views Discussion
2:30	Break
2:45	Architecture, Connectivity Views Discussion
4:30	Adjourn for the day

Agenda – Thursday 9/22

9:00	Welcome & Recap
9:30	Architecture, Communications Views Discussion
10:15	Break
10:30	Architecture, Information Viewpoint Discussion
11:00	Core System Deployment Options
11:45	Lunch
1:00	Core System Risks (Barriers to Deployment)
1:45	Next Steps
2:30	Adjourn

Opening Comments

- Welcome, Program Overview

Welcome & Introductions

- Who we are, how we got here
- Purpose for this week
  - Present the Core System Requirements and Architecture
  - Open discussion, questions/comments captured

Welcome & Introductions

- Speakers/Facilitators
  - Walt Fehr, US DOT
  - David Binkley, Lockheed Martin
  - Kevin Hunter, Lockheed Martin
  - Tom Lusco, Iteris
- Logistics
  - Breaks, Lunch – see agenda
  - Comments/Questions on Web – please use “chat” box

ITS JPO Program Structure

Applications

Safety			Mobility		Environment	
V2V	V2I	Safety Pilot	Real Time Data Capture & Management	Dynamic Mobility Applications	AERIS	Road Weather Applications

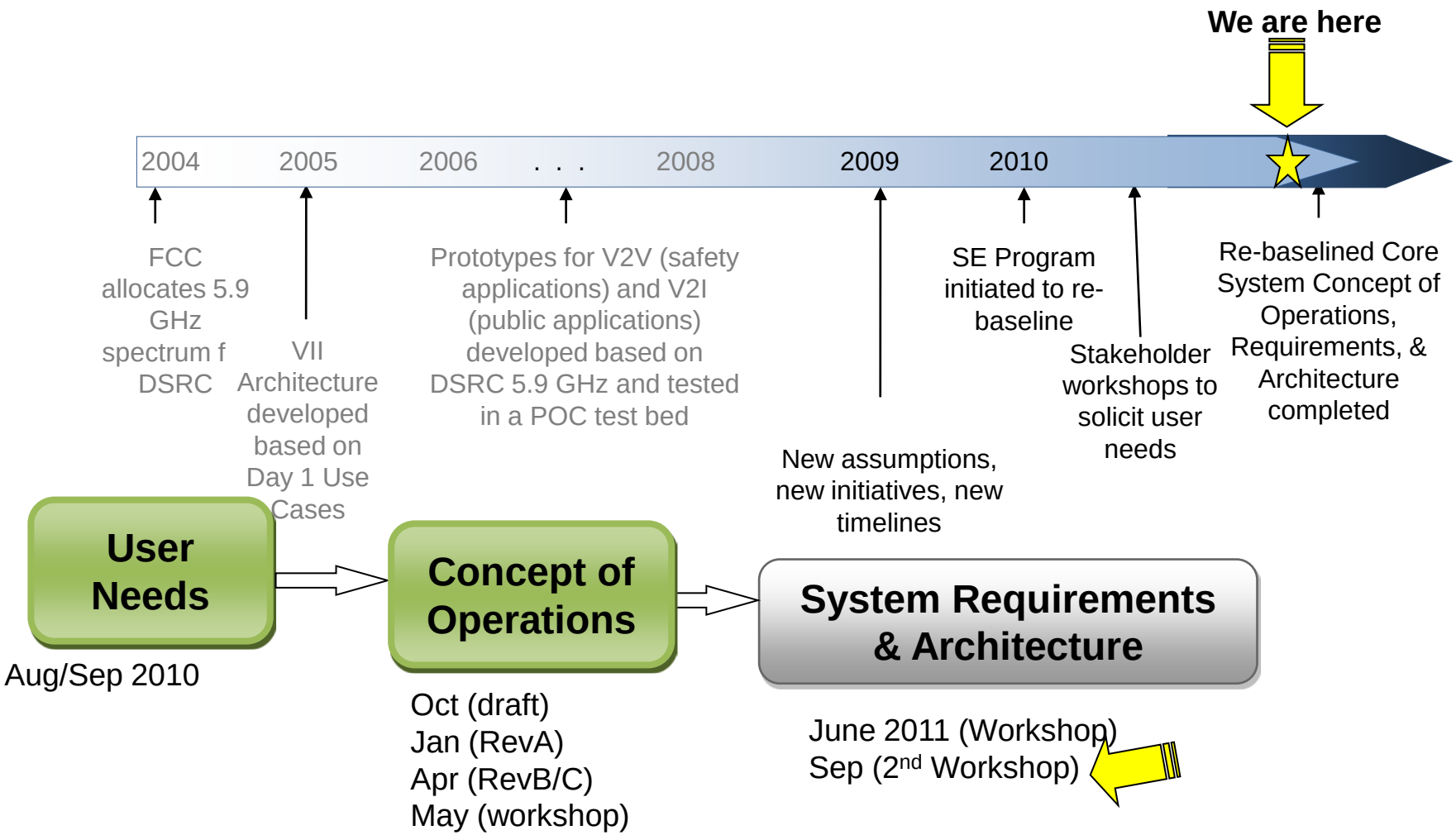
Technology

- Harmonization of International Standards & Architecture
- Human Factors
- **Core Systems Engineering**
- Certification
- Test Environments

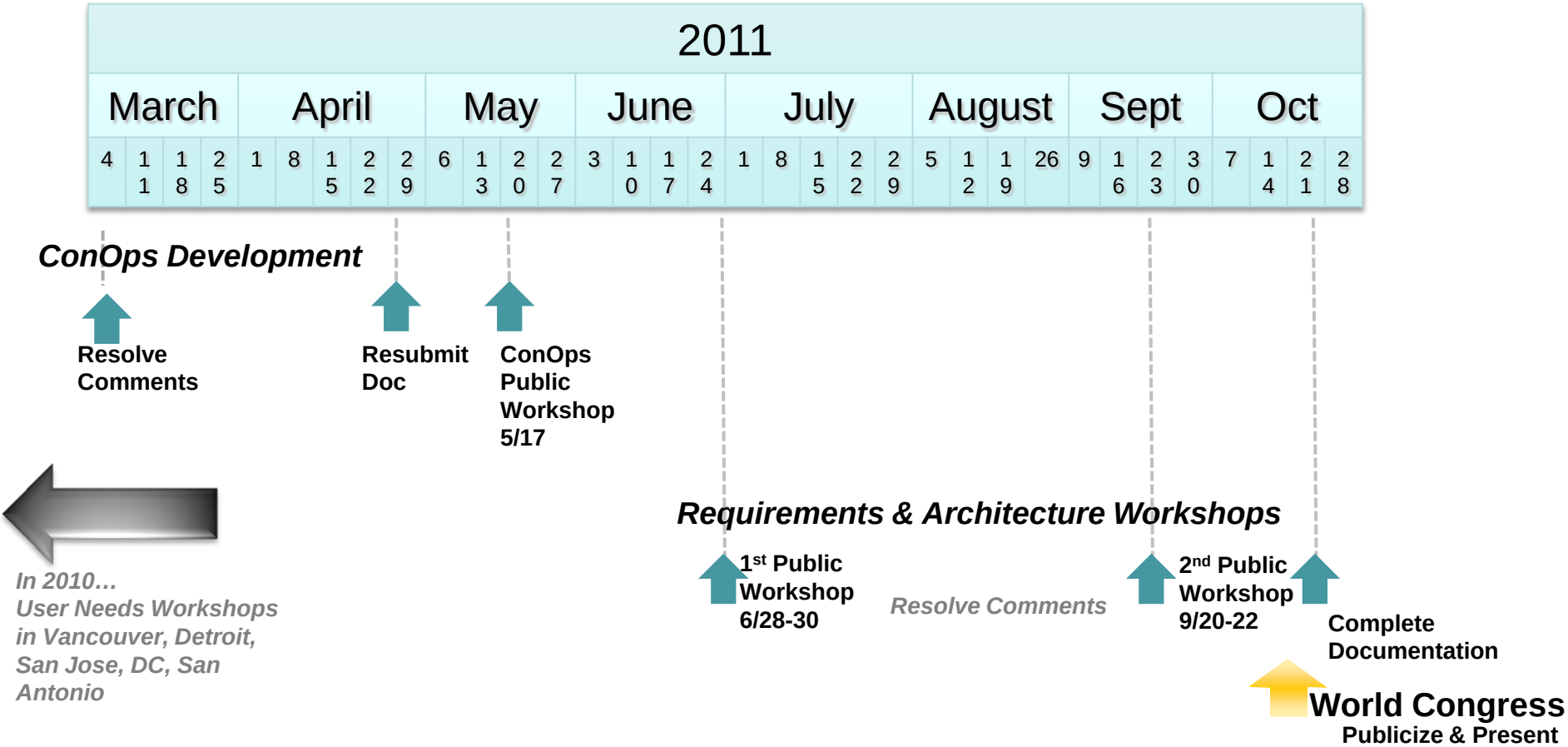
Policy

- Deployment Scenarios
- Financing & Investment Models
- Operations & Governance
- Institutional Issues

Core System Timeline



Core System Development Process

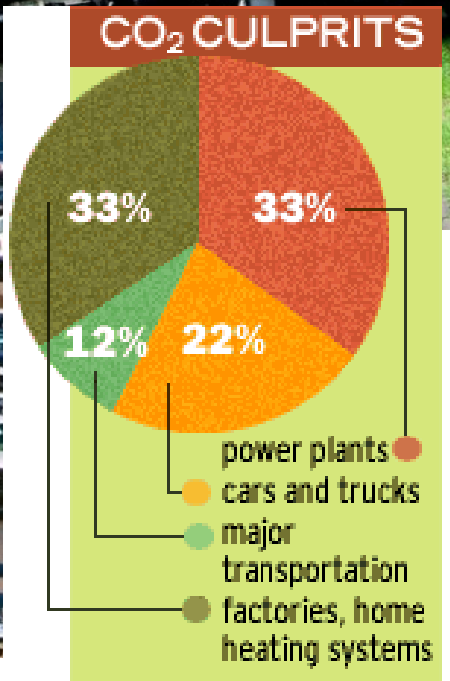


Core System Background

- Background, Overview of the overall environment, Core System and its components

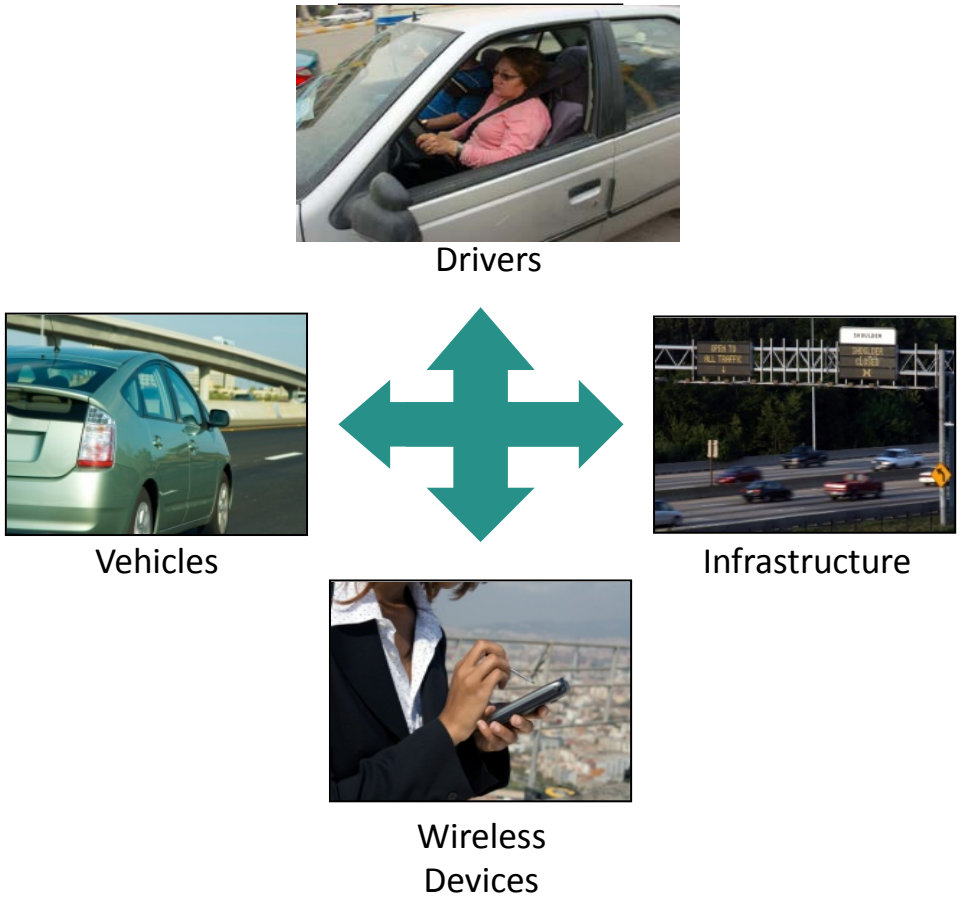
The Problem

- Safety
  - 32,788 deaths in ‘10
  - 5.5M crashes/year
  - Leading cause of death for ages 4-34
- Mobility
  - 4.8 billion hours of travel delay
  - $115 billion cost of urban congestion
- Environment
  - 3.9 billion gallons of wasted fuel
  - Emissions

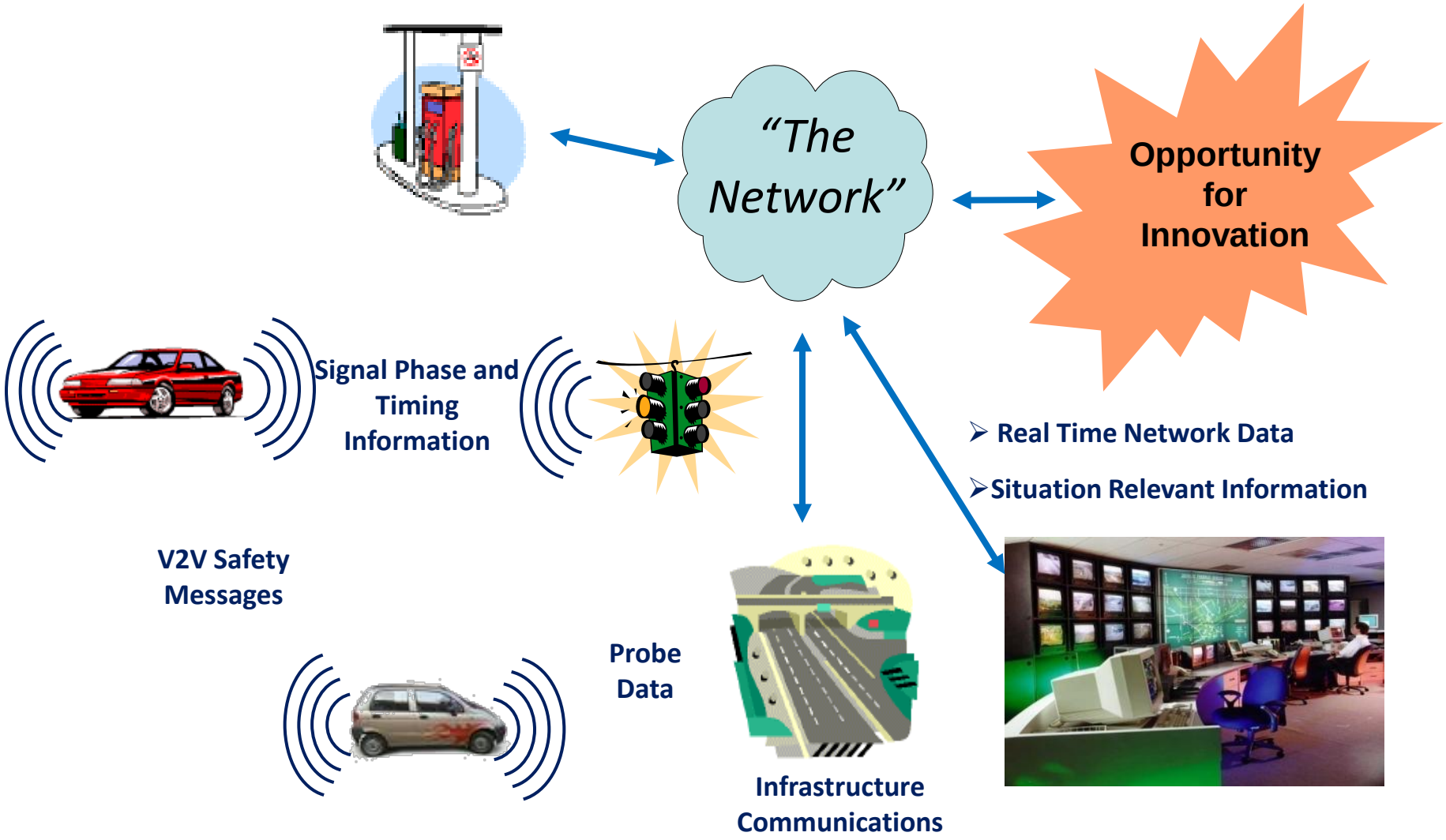


In an environment of *connected vehicles*…

- Drivers, Passengers, System Operators
- Using wireless communications technologies and applications
- Realize
  - Safety, Mobility, Environmental benefits



Connectivity drives the benefits



Driving Influences

- Since VII, additional communications media available
- Expanding of both mobile platforms (all vehicle types, plus pedestrians and other road users) and potential users of data, providers of data (not just traditional transportation players)
- Applications development expanding
- Data capture and usage decoupled from a single large system

Core System Needed

- The *connected vehicle* environment needs some enabling system that
  - Provides common services and interfaces
  - Provides trusted environment for the applications and users
  - Supports diversity
    - Applications, communications media, deployment models
  - Supports the future
    - Future technologies, extensible architecture

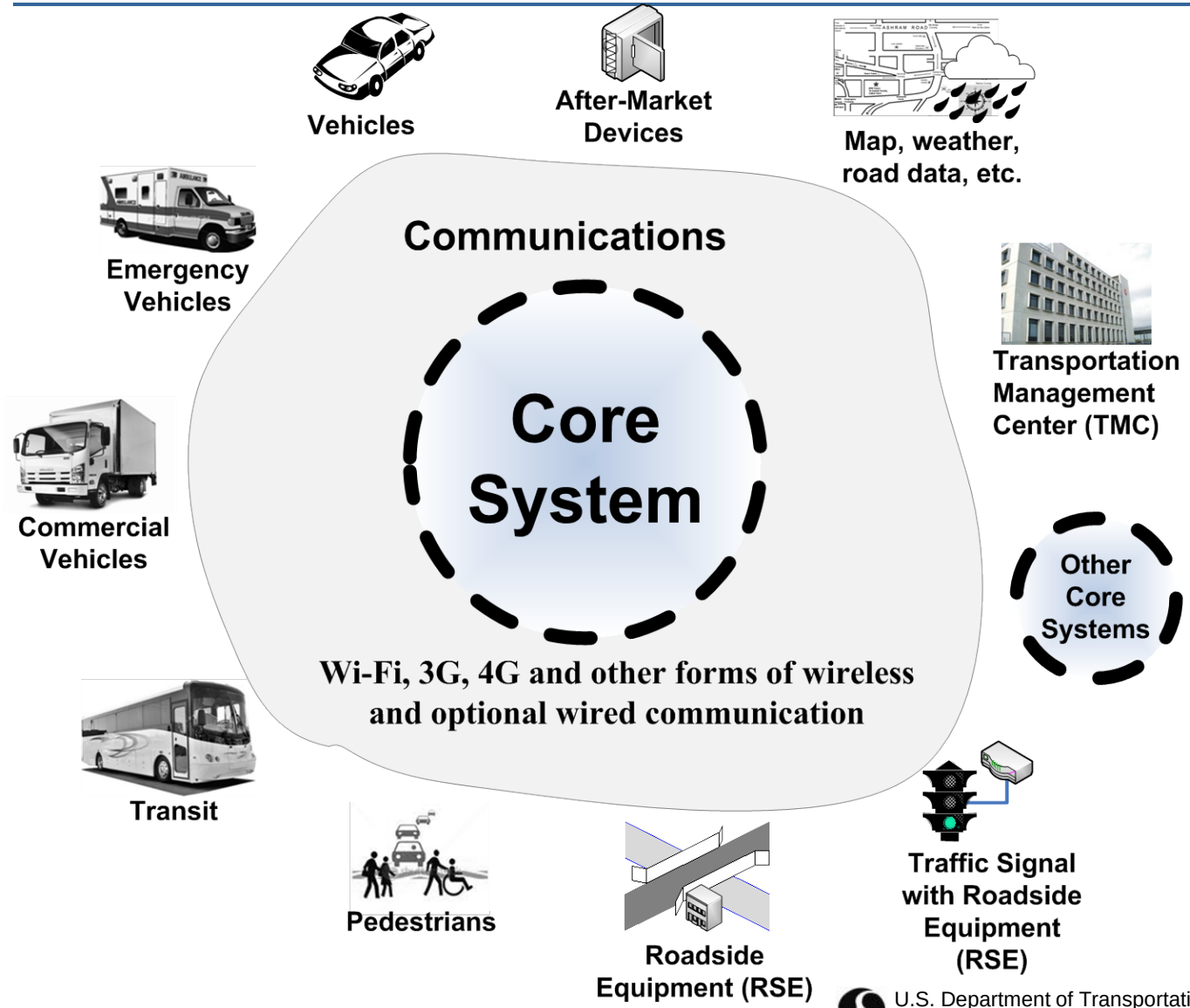
Core System provides services that…

- Enable data transfers between system users
  - Mobile
  - Field
  - Center
- Are in a secure, trusted environment
  - Enabling trust between parties that have no direct relationship
  - Enabling secure data exchange between parties that have no direct relationship
  - Enabling the exchange of data between parties that have data and parties that want data

Core System Scope

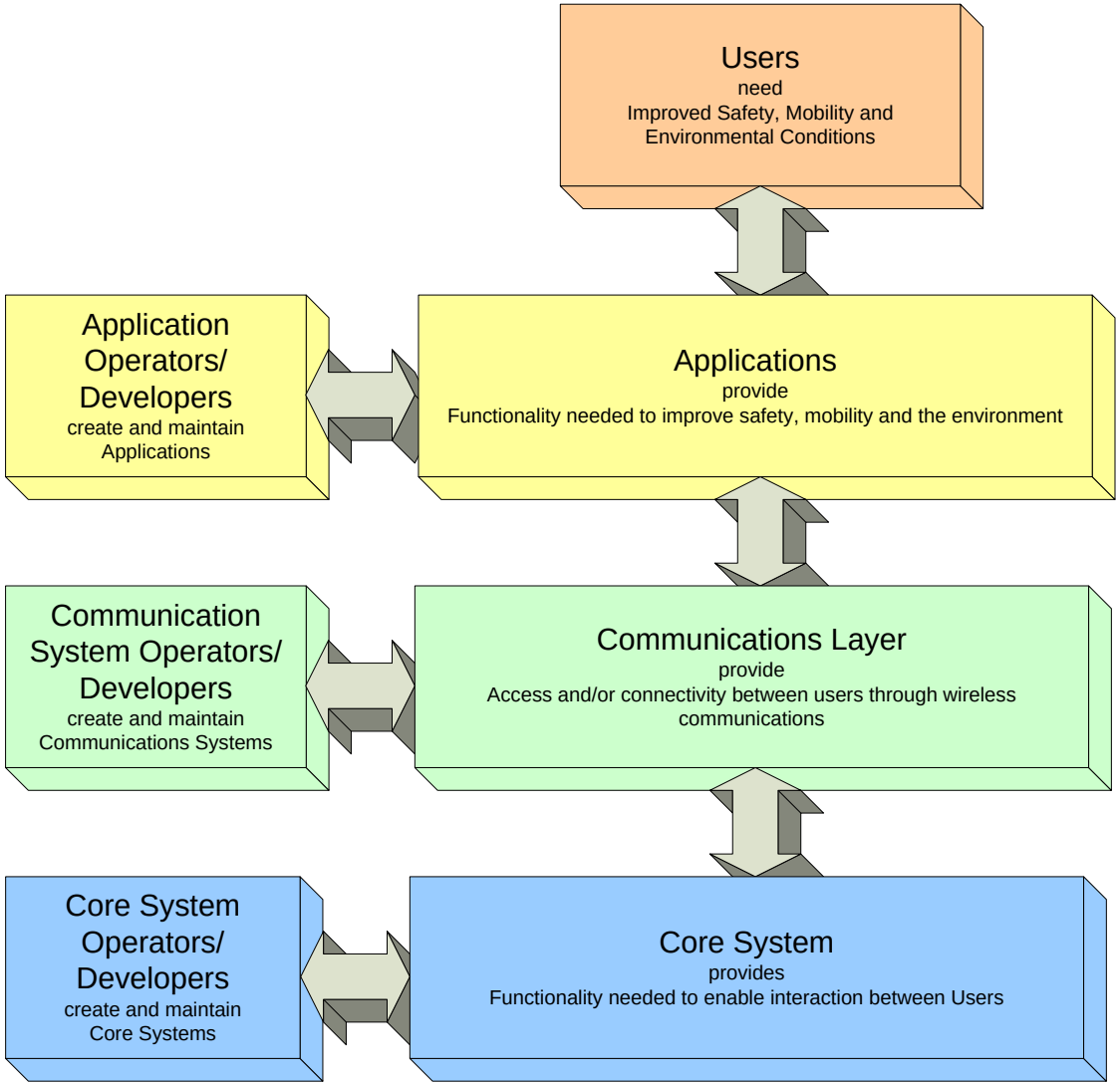
- What the Core System Does NOT do…
  - Store data for long periods of time
  - Host applications
  - Sit in a single location
  - Require any particular communications or hardware technologies (other than what will be needed to support the requirements)
  - Provide 1609.2 CA/RA functionality

Core System provides services to serve a large set of applications on diverse platforms

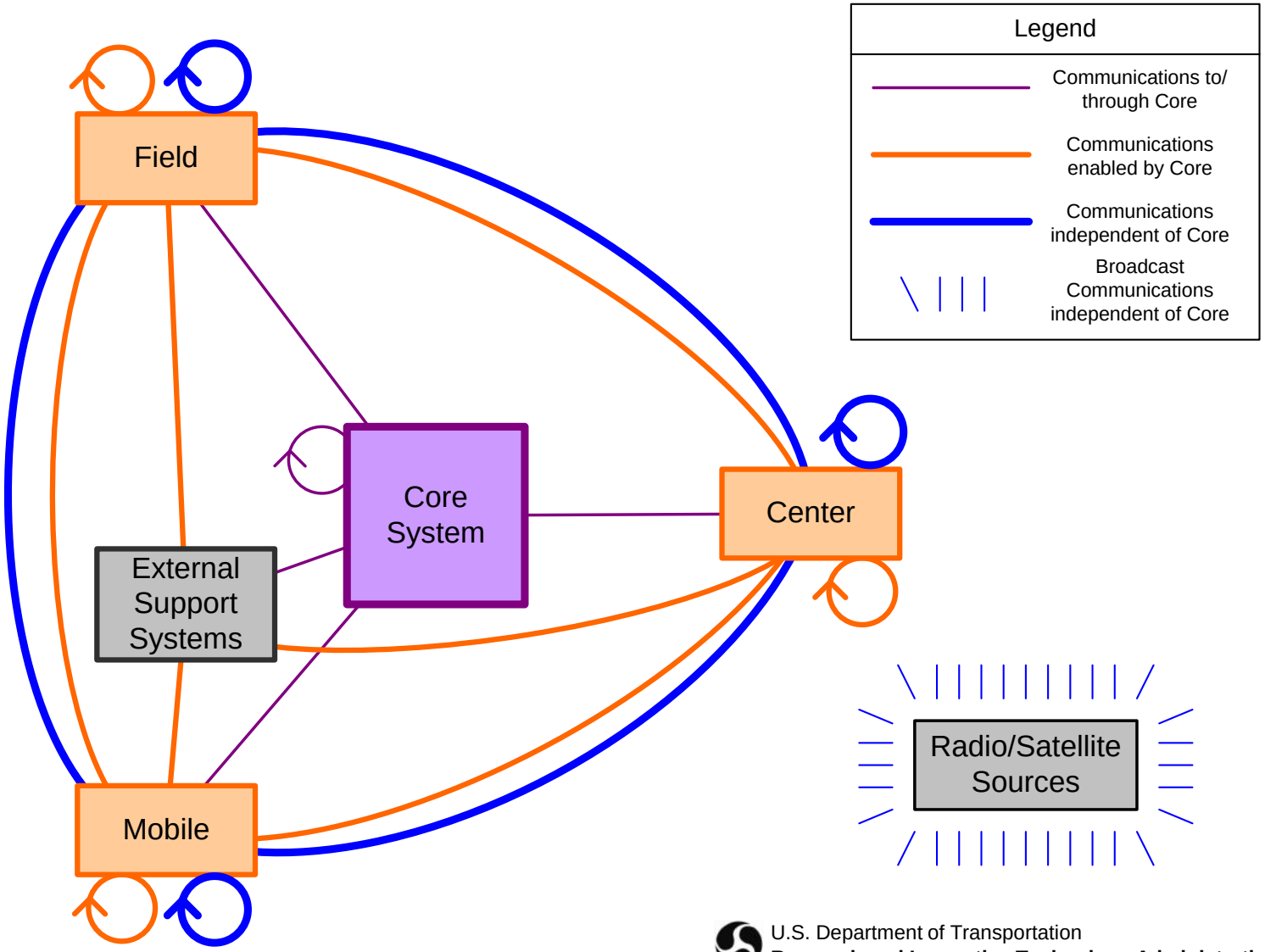


Connected Vehicle Environment Layers?

Organizing how users interact with each other and the Core System



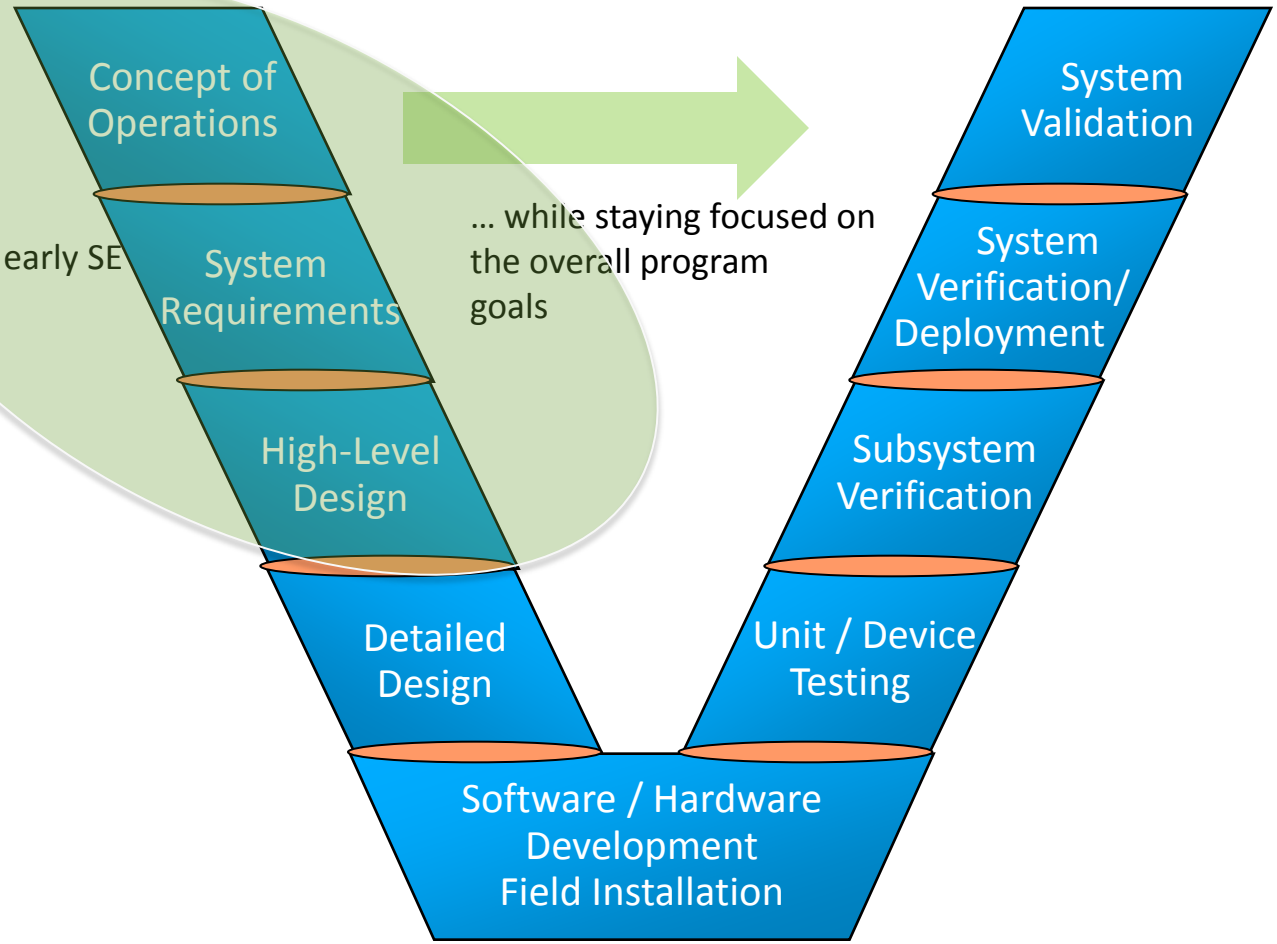
Core System in the context of the *connected vehicle* environment



The System Engineering Process

Concentrate on the early SE life cycle…

… while staying focused on the overall program goals



Gathering User Needs, What we heard…

- Give me the data – current traffic, all roads, all the time
- Standardize it
- Support multiple modes – include Cyclists, Pedestrians, other vulnerable users
- Set driver's expectations: inform them when safety or mobility services are available
- Support targeted broadcasts to sets of vehicles by location, type, individual
- Support multiple uses of data sets via standardized interfaces, services
- Support roaming for users devices
- Provide authentication, ensuring users that messages are from legitimate sources

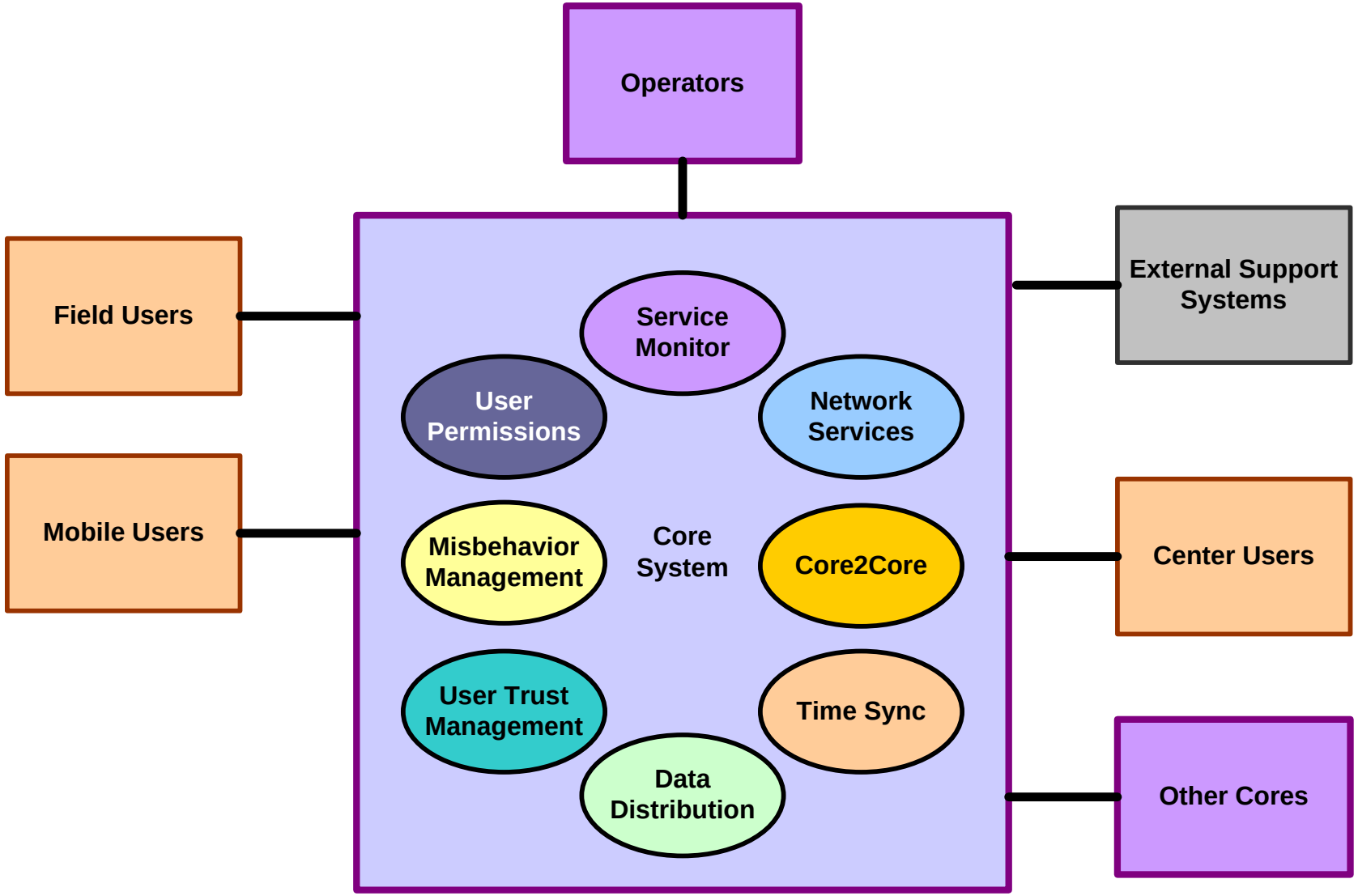
Core System ConOps

- Characterizes the Current System
- Identifies users' needs
- Defines concepts of the proposed system
- Shows operational scenarios of the proposed system
- Summarizes impacts, provided supporting analysis

Core System's Operational Characteristics

- Collection of services
- Heterogeneous community of systems, agencies, locations
- (Don't think control center) – not a physical plant, it's a collection of services
- Different Deployment Considerations
  - Standalone
  - Collocated
  - Distributed

Core System's 8 Subsystems



From ConOps to Req/Arch

Concept of Operations Document

- User Needs, Expectations, Constraints
- High Level System Description
- Operational Scenarios

System Requirements Specification

- *What* – functional requirements, interface definitions
- *How Well* - performance
- *Under What Conditions* – environmental, non-functional

System Architecture Document

- *Framework*
- *Address Stakeholder Concerns*
- *High-Level System Definition*

Review Plan

- Review the Needs, how they've evolved
- Introduce System Architecture
  - Viewpoints/views
- Introduce System Requirements
- Discuss Architecture Views
  - Include requirements supporting each area, alternatives that were explored
- Discuss Deployment Options, Overall Risks

Core System Needs

- What's driving the Core System

Core System Needs

- 2 Types of Needs Addressed:
  - User needs – capability required for that user to accomplish their goal
  - System needs – capability required in order to meet the operational goals
- In a nutshell:
  - Provide trust/security
  - Enable data exchanges
  - Take care of itself
  - Work with other cores

Things the Core System Needs to Do

- Data protection
  - Facilitate secure exchange of data
- Facilitate trust
  - With and between System Users
  - Revoke trust credentials when necessary
- Authorization
  - Manage who can do what
  - Verify
  - Identify misbehavior and allow System Users to provide misbehavior input

Listed in ConOps and SyRS Table 5-1

Things the Core System Needs to Do, continued

- Time
  - Operate on a common time base (for components of the Core System)
- Network Services
  - Support users accessing Core System over variety of communications mechanisms
  - Support connections to a private network (in addition to the Internet)
  - Route communications between Cores and System Users if using a private network

Things the Core System Needs to Do, continued

- Facilitate the provision of data
  - Match data providers with data requesters/consumers
  - Forward (or redistribute) data (publish-subscribe, aggregation, anonymization, etc.)
  - Facilitate situational-relevant distribution
    - Geography or Time

Things the Core System Needs to Do, continued

- Take care of itself
  - Service status
  - Integrity protection
  - System availability
  - Performance monitoring
  - System data security
- Preserve System User's anonymity

Things the Core System Needs to Do, continued

- Coordinate activities with other Core Systems
  - Support multiple, independent deployments
  - While providing interoperable services across Cores
  - And coordinating activities together to deliver information consistently

Constraints / Assumptions

- VII Privacy Policies Framework still applies
- IEEE 1609.x family used for DSRC
- X.509 based certificates except for DSRC-specific apps
- SAE J2735 basis for mobile user messages
- Standards may need to be developed or modified for Core System interfaces
- Core System interfaces based on IPv6

Constraints / Assumptions

- Comm will be provided by System Users
- Data (probe, basic safety) provided anonymously by mobile users
- Some vehicle based safety applications may be mandatory (beyond scope of this effort)
- Other mobility applications will be opt-in
- Deployment of Cores may be evolutionary, regional

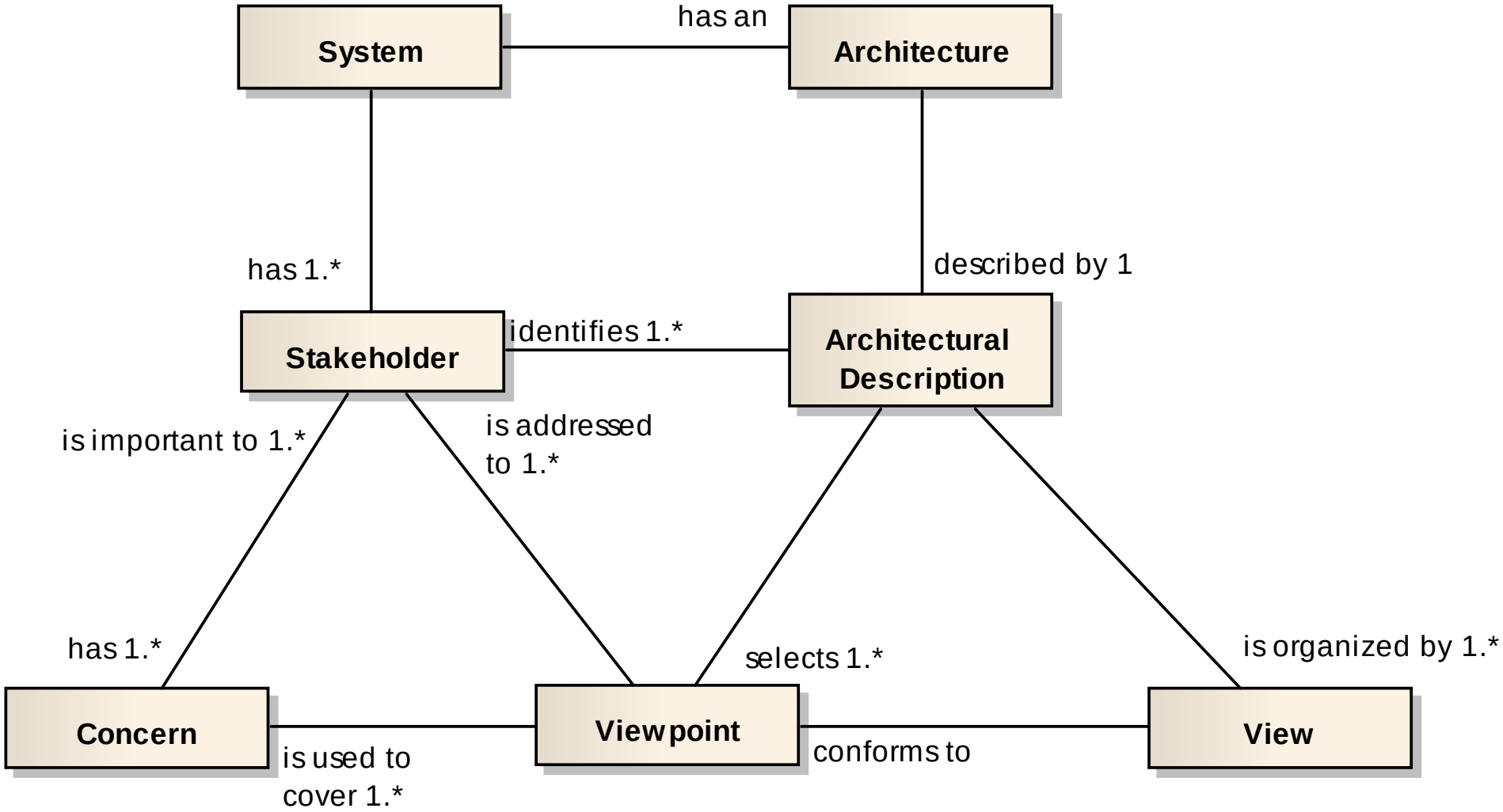
Core System Architecture, Intro

- 5 viewpoints and how they tell the story

System Architecture Terms

- Architecture: The fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution. (IEEE 1471)
- Viewpoint: a framework of rules for developing architecture views based on a related set of concerns
- View: a representation of the system from the perspective of a set of concerns

System Architecture Relationships



What does the Architecture include?

- Stakeholders and their Concerns
- Viewpoint Specifications
- Views defined in accordance with those Viewpoints, each of which addresses specific stakeholder concerns
- Traceability between objects in architecture to requirements in the SyRS

Who are the Stakeholders?

- Operators/Users of the Core System
  - Transportation users
  - Transportation system operators
  - Core System admin personnel
- Along with
  - Developers, Maintainers, Testers
  - Managers, Acquirers
  - Application and Device Developers
  - Service Providers
  - Policy Setters

Stakeholders' Concerns

- Performance
  - System performance
  - Reliability
  - Availability
- Interfaces
- Functionality
- Security
- Organization/ Resources
- Appropriateness
- Feasibility
- Risks
- Evolvability
- Deployability
- Maintainability

Architecture Viewpoints

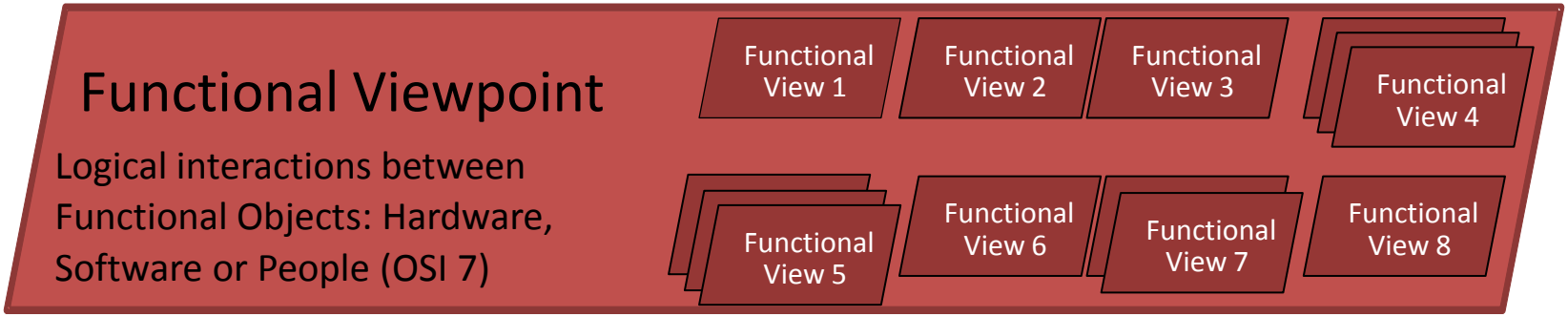
- Provide framework to address all concerns
- Describe the system from different perspectives
- Describe the Core System as a set of Objects and interactions among them
- Expose a different set of design concerns and issues
- Provide the means for reasoning about aspects of the system
- Trace to requirements (in SyRS)

Architecture Viewpoints

- Enterprise:
  - Organizational entities and their relationships. Focuses on scope and policy

Architecture Viewpoints

- Functional:
  - System as a collection of abstract objects that interact at interfaces



Architecture Viewpoints

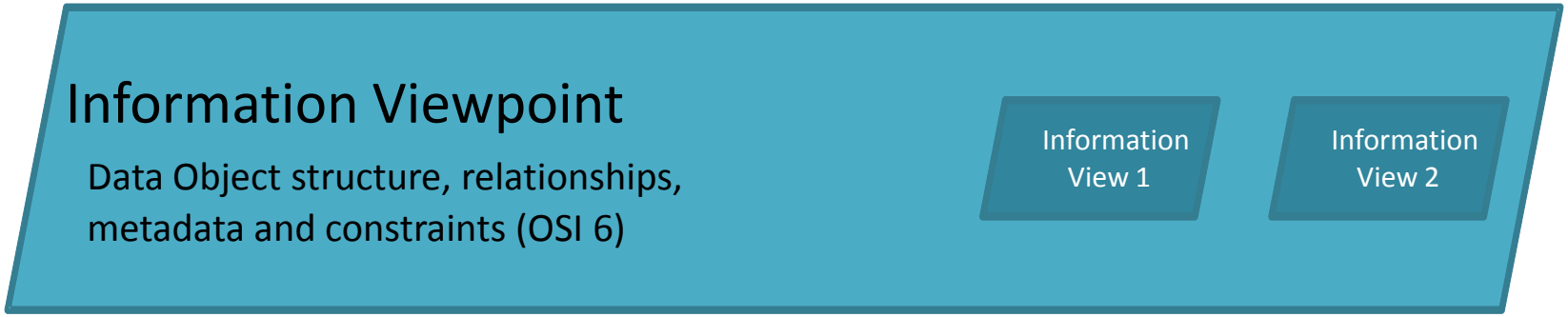
- Connectivity:
  - System as a set of components that interact across links

Architecture Viewpoints

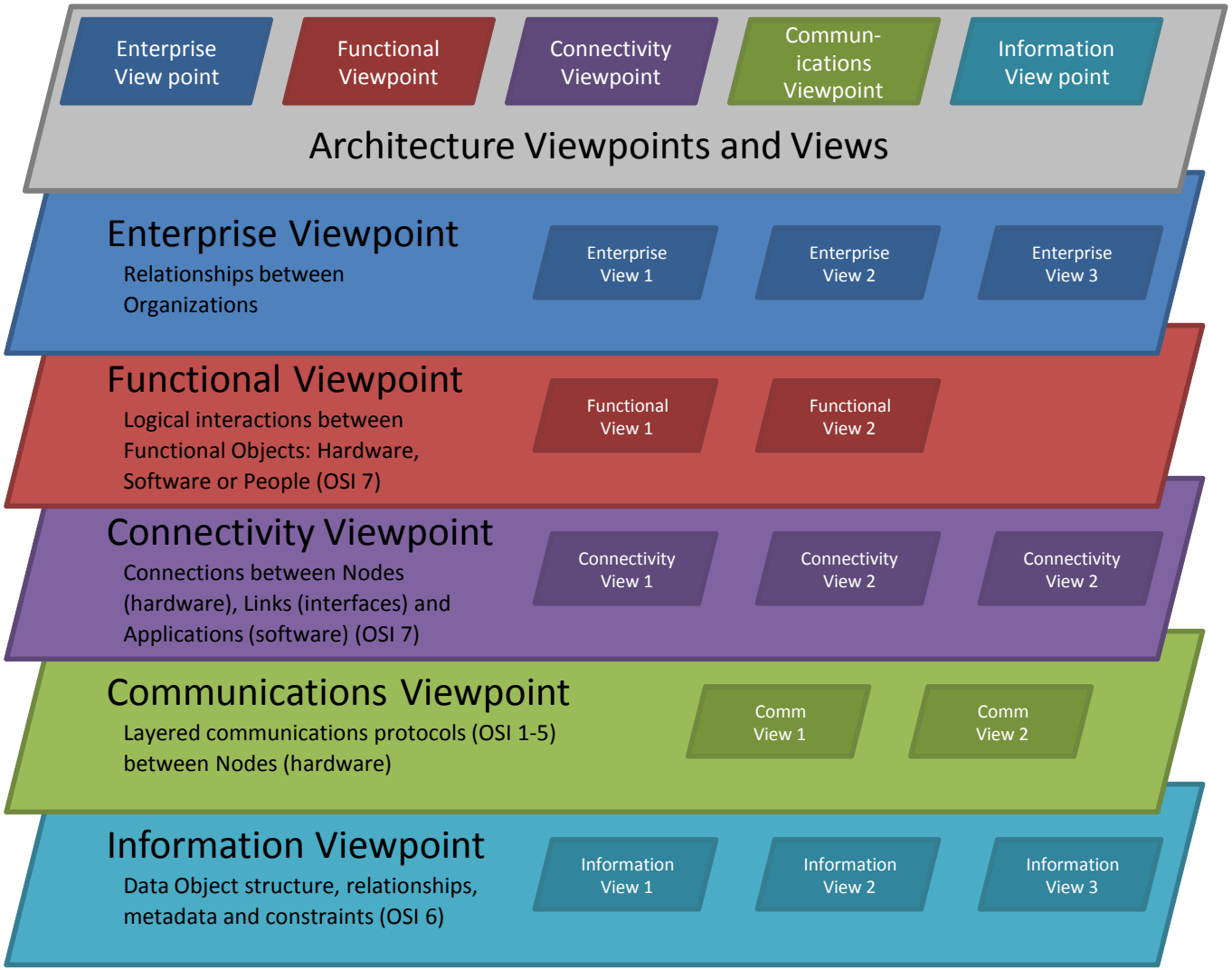
- Communications:
  - Mechanisms required to communicate between system components

Architecture Viewpoints

- Information:
  - Kinds of information handled by the system



Architecture Viewpoints & Views



Architecture Guide

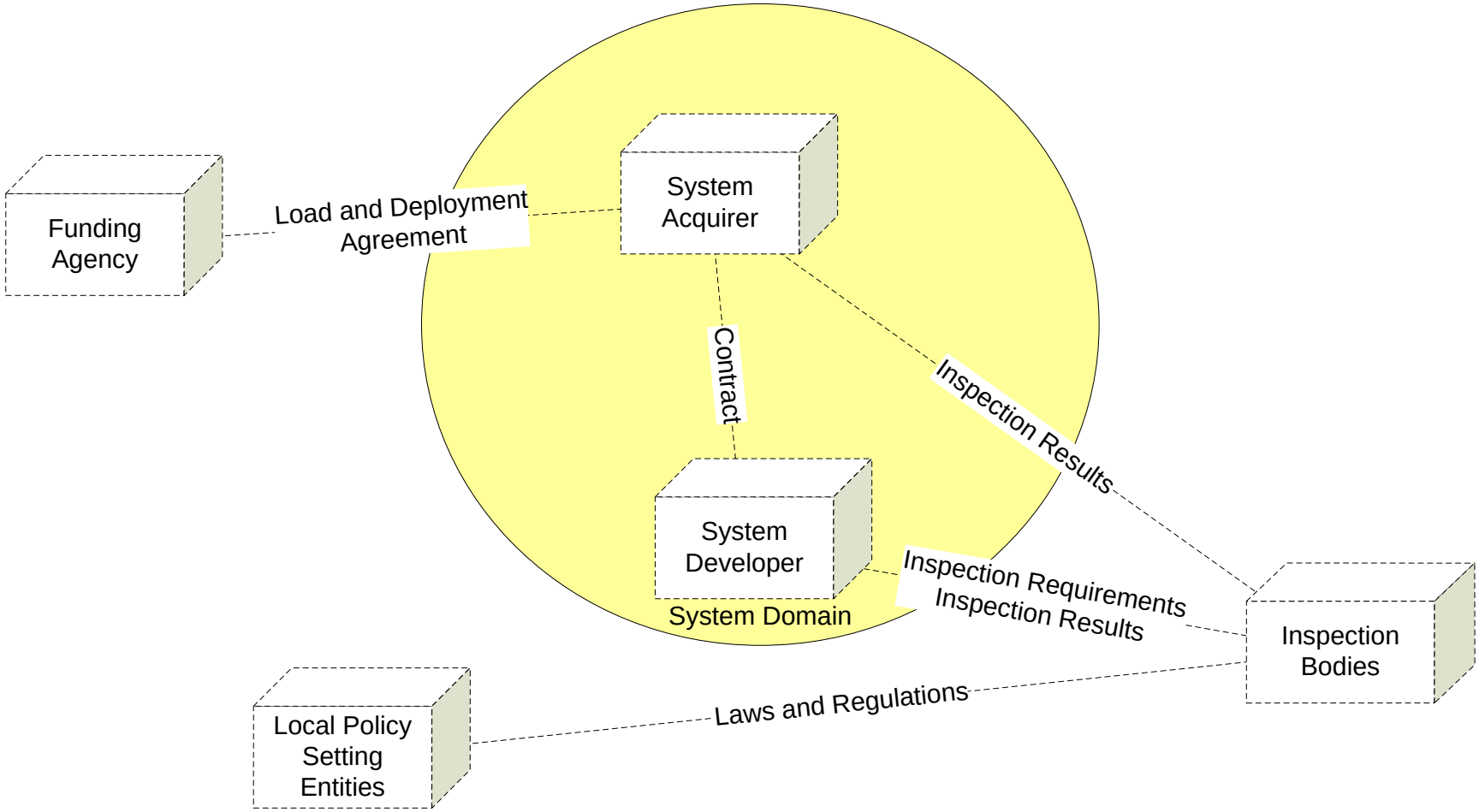
- What is in each view?
- How do I read a view?
  - Definitions
  - Description
  - Diagram
  - Alternatives

Enterprise

- Addresses the relationships between organizations
- Roles those organizations play that involve various resources

Local Policy Setting Entities	Enterprise Objects. (Duplicates will be shaded.)
Core System Facility	Facilities. (Duplicates will be shaded.)
Standards	Logical relationships between Enterprise Objects.
Core System	Domains.

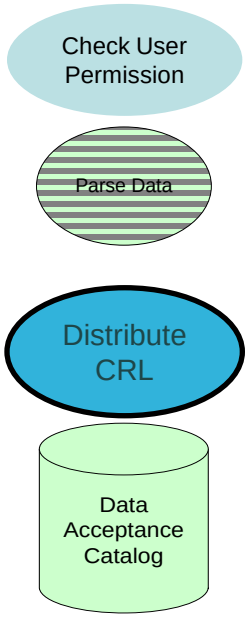
Example Enterprise View



Functional

- Focuses on the behavior, structure, and interaction of the functions performed by the system
- Shows functions for each subsystem
- Traceable to *functional* requirements
- Color coding:
  - Subsystems each represented by a different color
  - Information Objects are the same color as the source Function object

Functional

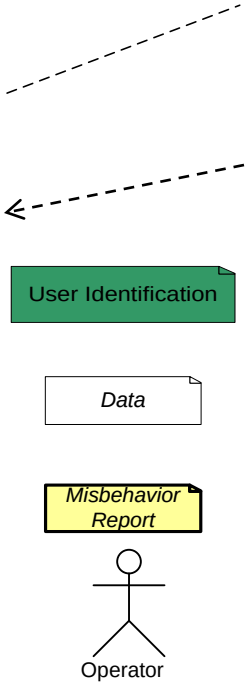


Functional Objects

Functional Objects that represent optional functions

Functional Objects that store encrypted data

Data stores



Logical relationships between Functional Objects

Data flow between a Functional Object and an external Object

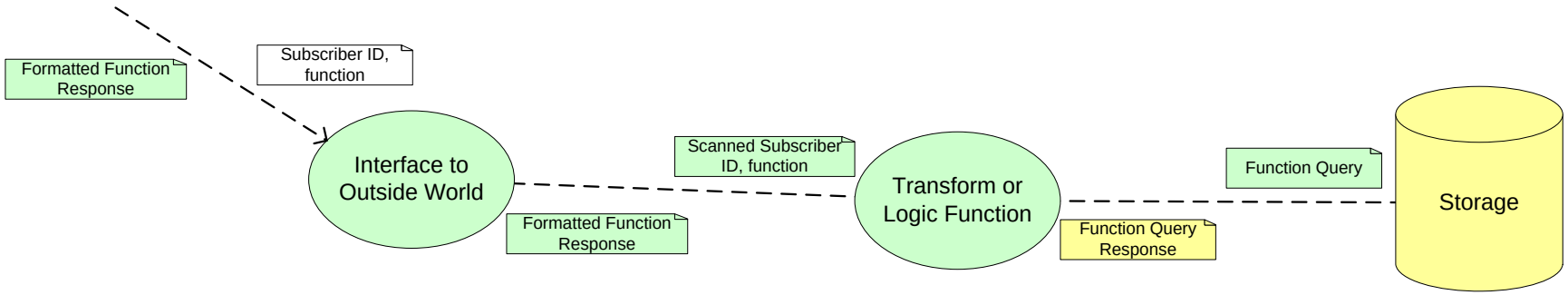
Information Object

An Information Object whose sender expects an acknowledgement

An Information Object that is secure

An external actor

Example Functional View

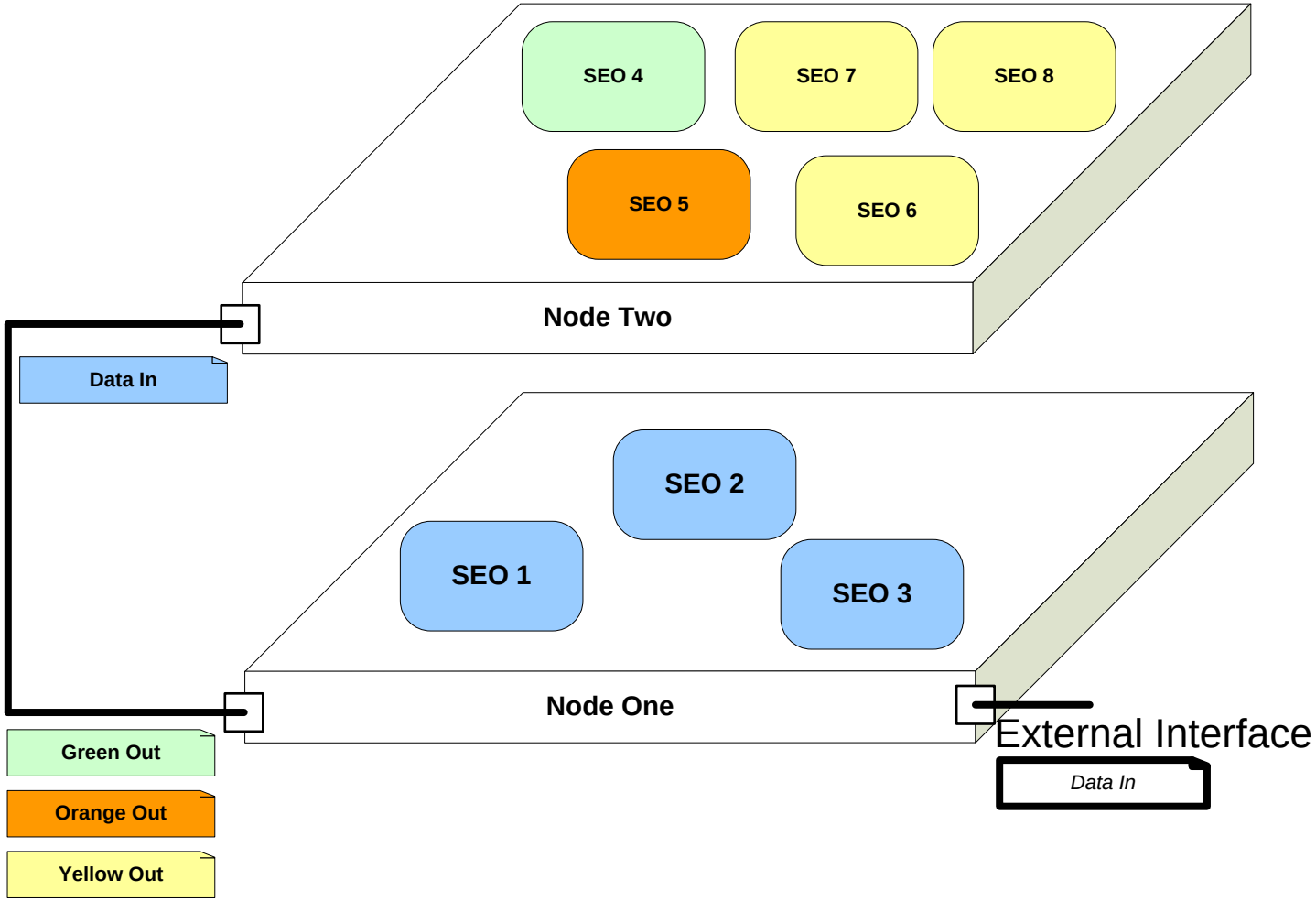


Connectivity

- Composition of the physical elements (nodes) and their connections and interactions
- Links are traceable to *interface* requirements

DSRC Device	Nodes
	link between Nodes, likely a wired connection
	link between Nodes, likely a wireless connection
Core Router	Applications external to the Core System and Core Functional Objects
User Identification	An Information Object
	Ports

Example Connectivity View

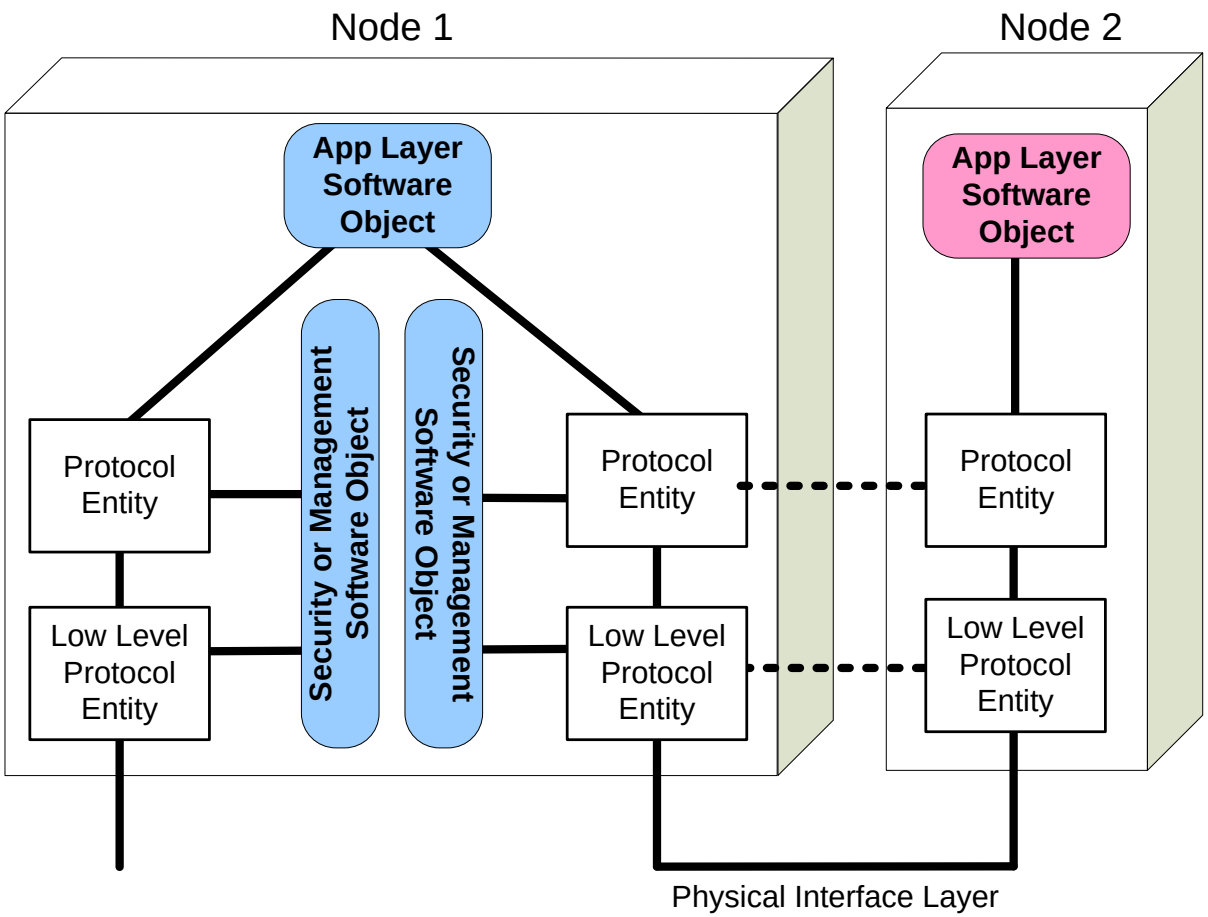


Communications

- Layered communications protocols between nodes
- Links are traceable to interface requirements

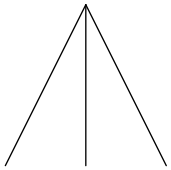
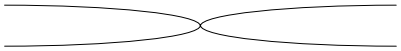
Internet Protocols	Protocol entities
	Node
	Link between nodes
	Logical link between protocol entities
Core Router	Software engineering object

Example Communications View

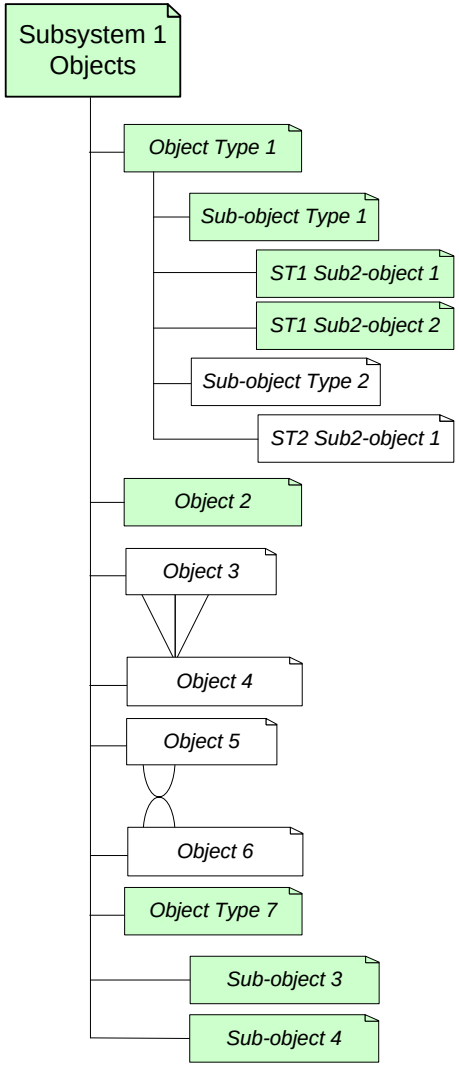


Information

- Defines the Data objects: structure, relationships, metadata, and constraints
- Traceable to *interface, data* requirements

User Identification	Information Objects
	“part-of” relationship
	“aggregation” relationship
	“Transformation” relationship

Example Information View



Core System Architecture Status

- June 13 draft, reviewed at DC Workshop
  - Views at varying levels of completeness
  - Alternatives for evaluation/discussion
- Sep 6, reviewed here
  - All views are complete, traced to requirements
  - Alternatives evaluated - resulting choices documented in section 4, rationale and dissenting alternatives in section 6

Architecture Views

- Enterprise Viewpoint:
  - Security Credentials Distribution
  - Operations
  - Core System and Application Development and Deployment
  - Configuration and Maintenance
  - Governance
  - Business Model Facilitation
- Functional Viewpoint:
  - Top Level
  - Data Distribution
  - System Configuration
  - User Configuration
  - System Monitor and Control
  - Credentials Distribution
  - Misbehavior Management
  - Core Decryption
  - Networking
  - Core Backup

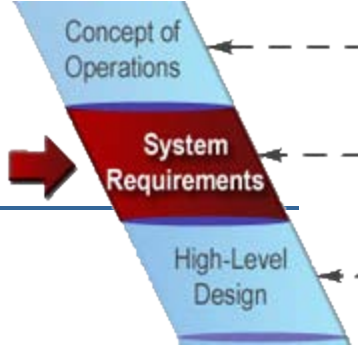
Architecture Views, continued

- Connectivity Viewpoint:
  - High Level
  - Core System Function Allocation
  - State and Mode Transitions
- Communications Viewpoint:
  - Mobile DSRC Device and Core
  - Mobile Wide-Area Wireless User and Core
  - Fixed Point Center/Field User and Core, Core2Core
  - Core Routing
- Information Viewpoint
  - Top Level External Objects
  - Top Level Internal Objects

Core System Requirements

- Introduction, organization, definitions

System Requirements

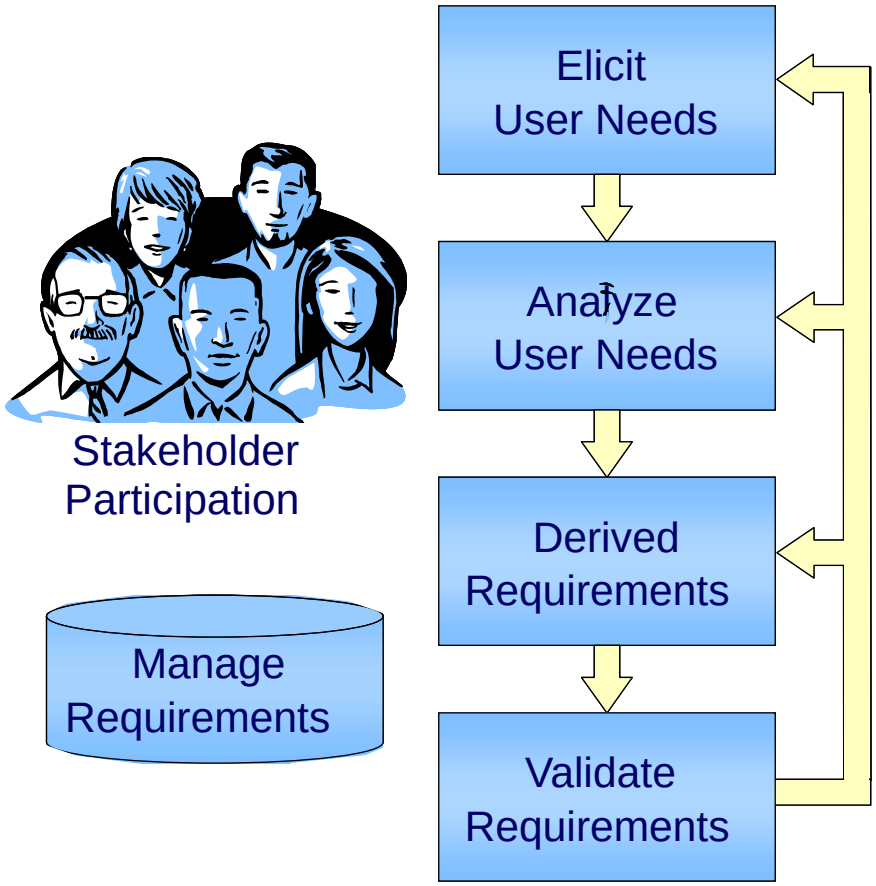


"Something that governs *what*, *how well*, and *under what conditions* a product will achieve a given purpose"

-- EIA-632, Electronics Industry Association Standard "Processes for Engineering a System"

System Requirements

- Key activities
  - Elicit User Needs
  - Analyze User Needs
  - Derived Requirements
  - Validate Requirements
  - Manage Requirements

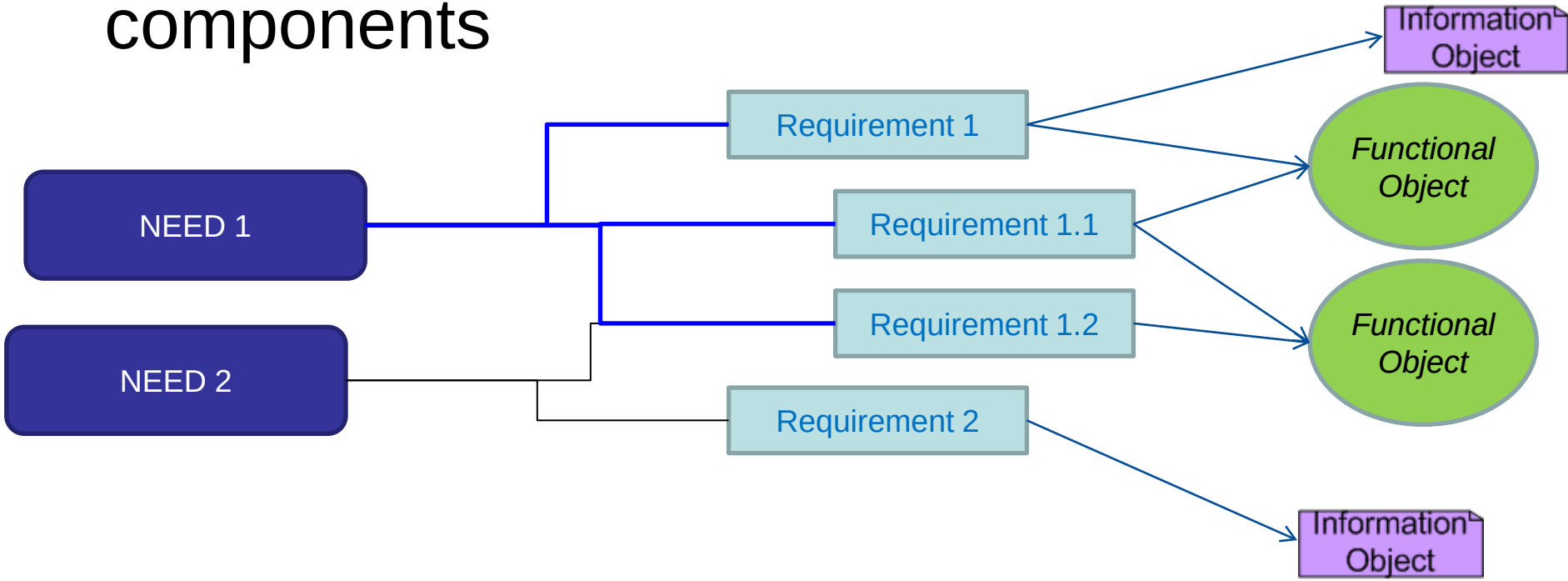


Core System Requirements

- Levels of Requirements
  - System
  - Subsystem
    - With sub-requirements as needed
- Types of Requirements
    - Functional Requirements
    - Performance Requirements
    - Interface Requirements
    - Non-Functional System Requirements
    - Constraints

Requirements "Readers Guide"

- Requirements specify what the system "shall" do to satisfy the needs of the users
- Traceable to both Needs and Architecture components



Requirements “Readers Guide”

- Look for structure/grammar of requirements

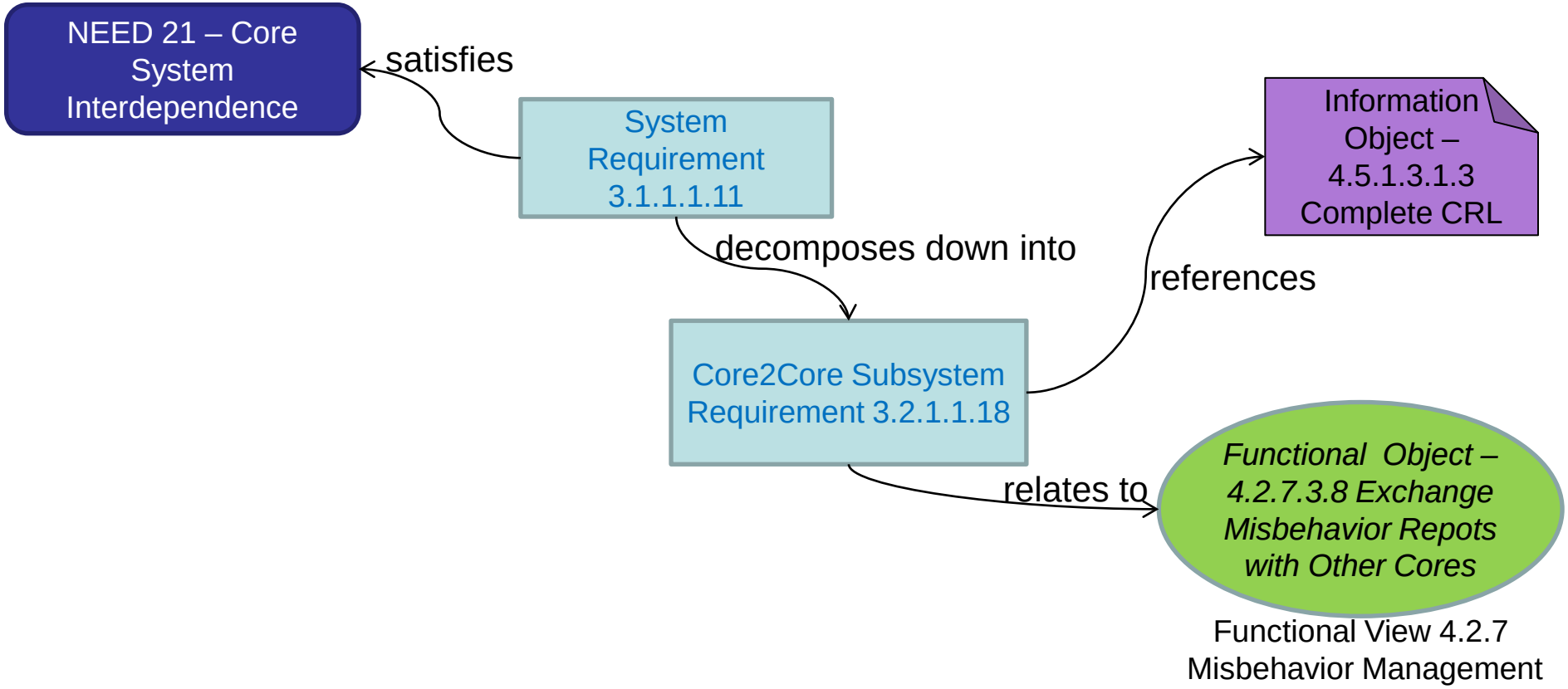
[ID] [Actor] [Action] [Target] [Constraint] [Localization]

- Identifier
- Actor/Subject
- Action Verb
- Target/Object
- *Constraint, Localization*

3.1.1.1.11 The Core System shall transmit the 4.5.1.3.1.3 Complete CRL to other Core Systems.

Requirements "Readers Guide"

3.1.1.1.11 The Core System shall transmit the 4.5.1.3.1.3 Complete CRL to other Core Systems.



Characteristics of Good Requirements

- ✓ Necessary
- ✓ Concise
- ✓ Attainable
- ✓ Standalone
- ✓ Consistent
- ✓ Unambiguous
- ✓ Verifiable

System Level Requirements

- Higher Level requirements, Related to Needs
- Not necessarily related to any one part of the system
- Includes types of requirements that will be decomposed to subsystem level later
  - Functional
  - Performance
  - Interface
- Includes some types of requirements not found in subsystem requirements
  - Non-functional, ‘ilities’
  - Constraints

System to Subsystem Requirements

- Core System Requirements go from the System level down to a subsystem level
  - What shall the ___ subsystem do?
  - To satisfy the overall system requirements and needs of the system
- Subsystem Requirements Divided by Type
  - Functional
  - Performance
  - Interface

Requirements Review

- Will review along with the Architecture Views
  - Data Distribution
  - Security Credentials Configuration, Distribution, Management, including Misbehavior Management
  - Core2Core interactions
  - Core Decryption, Networking