

The Interconnected Science Ecosystem (INTERSECT) Architecture

Christian Engelmann, Swen Boehm, Michael Brim, Jack Lange, Thomas Naughton, Patrick Widener, and Ben Mintz

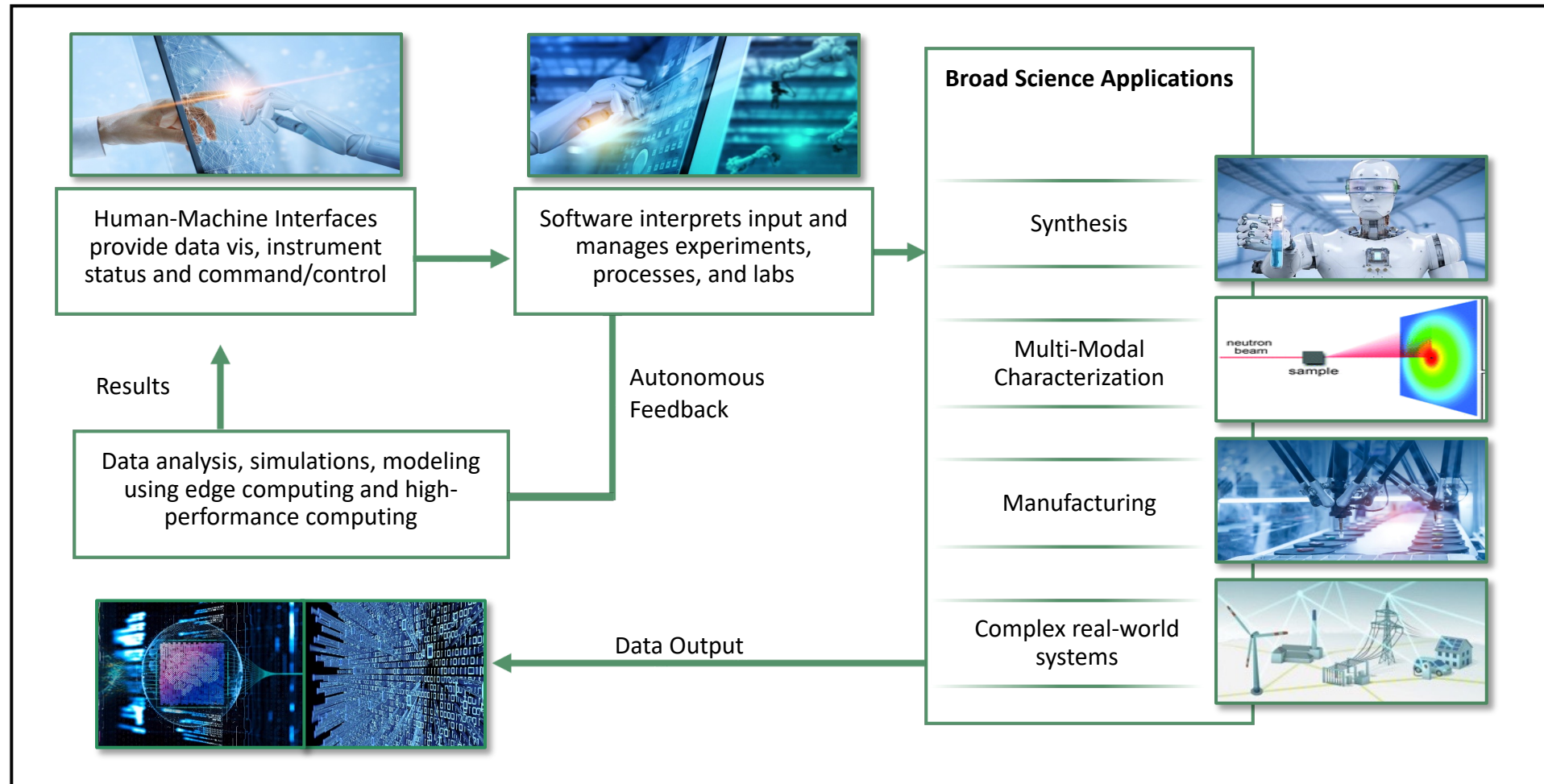
Oak Ridge National Laboratory

ORNL is managed by UT-Battelle LLC for the US Department of Energy



U.S. DEPARTMENT OF
ENERGY

Smart Autonomous Interoperable Laboratories



Common Ecosystem is Required for Interoperability

Guided by History

- Future Combat Systems (FCS) was the United States Army's principal modernization program from 2003 – 2009.
- The Boeing Company and Science Applications International Corporation (SAIC) worked together as the lead systems integrators, coordinating more than 550 contractors and subcontractors in 41 states
- Estimated program losses range from \$18-32B
- RAND Analysis of FCS (<https://www.rand.org/pubs/monographs/MG1206.html>)

- “an industry consortium led by Boeing and SAIC was effectively put in charge of overseeing its own performance”
- Entrenched communities were evident in the FCS program
- Overreliance that the acquisition community could develop and integrate items using both evolutionary and unknown revolutionary technologies
- An emphasis on the integration of technologies and advanced concepts allows the enforcement of system-of-systems discipline and curbs conflicting influences
- FCS involved the largest integrated set of requirements the Army had ever developed, and it was extremely difficult to analyze and understand precisely how all of them would interoperate
- Significant technology development should occur early in a program
- Alternative technology assessment metrics can supplement technical readiness levels, which may be inadequate for some aspect of system-of-systems acquisitions
- Having too many connections to or being too highly dependent on outside programs can lead to significant risk
- Risk-mitigation strategies that incorporate system-of-systems engineering practices will facilitate risk mitigation across systems



No single team has all the answers!



Early user engagement/adoption is critical!



Wild-wild west integration does not work!



Early technologies should focus integration!



Interoperability must be a primary goal!



Reuse software! Continuous Dev/Integration!



Technologies must be interchangeable!

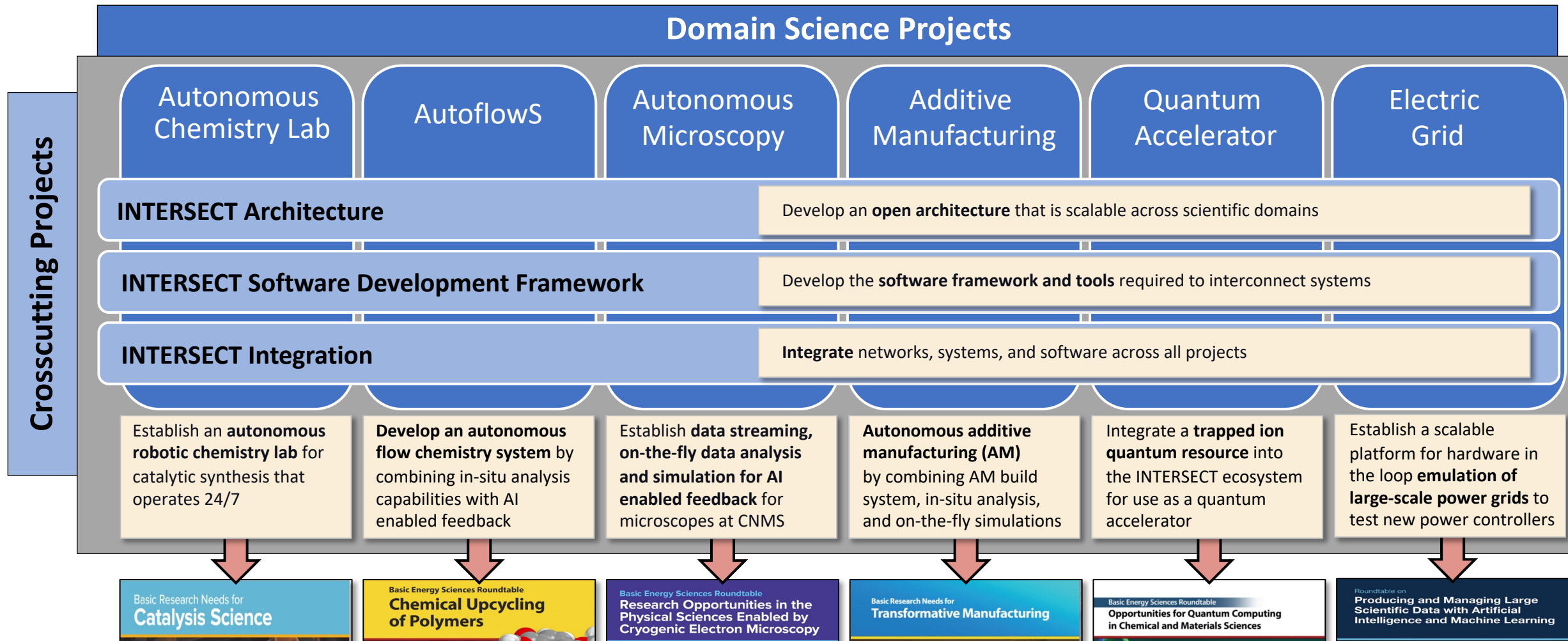


Define compatibility or compliance!



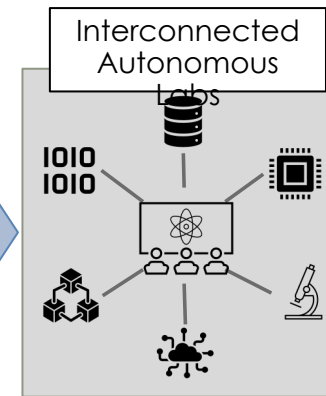
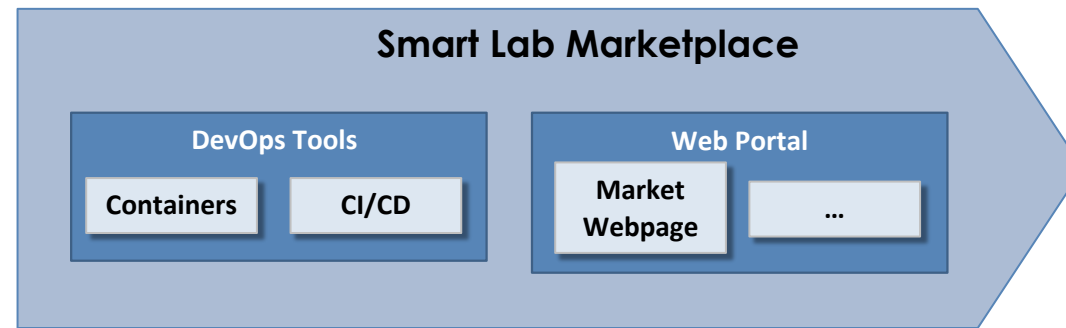
Good engineering practices are key!

INTERSECT Programmatic Structure



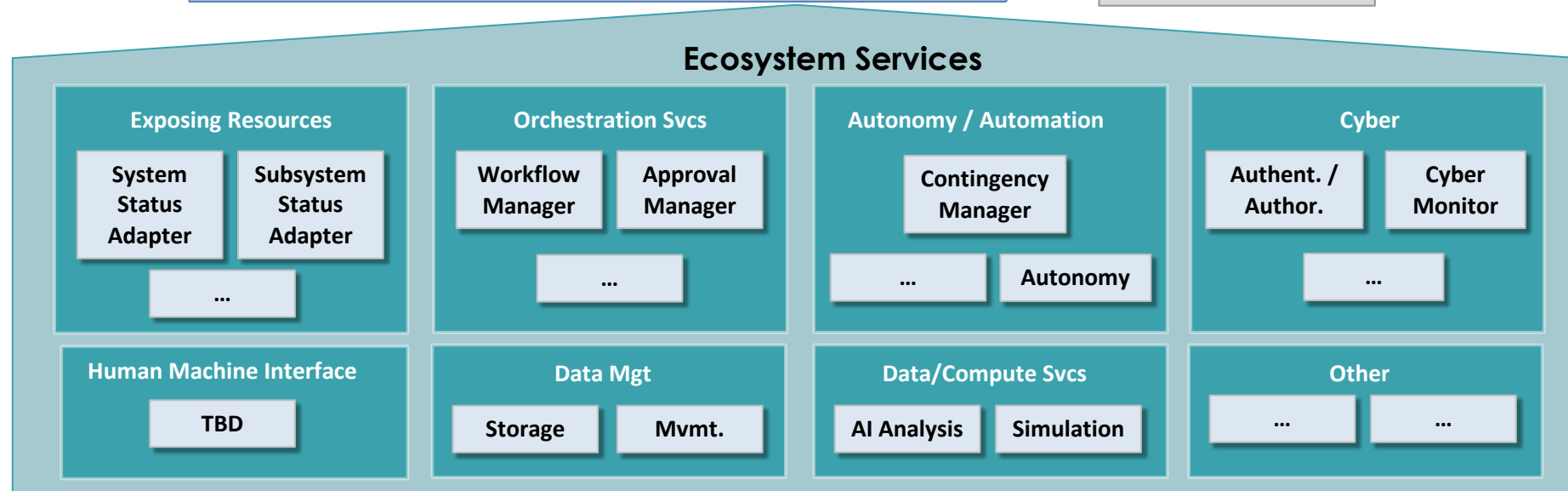
Interconnected Science Ecosystem

4) Create
Autonomous
Lab Software
Marketplace

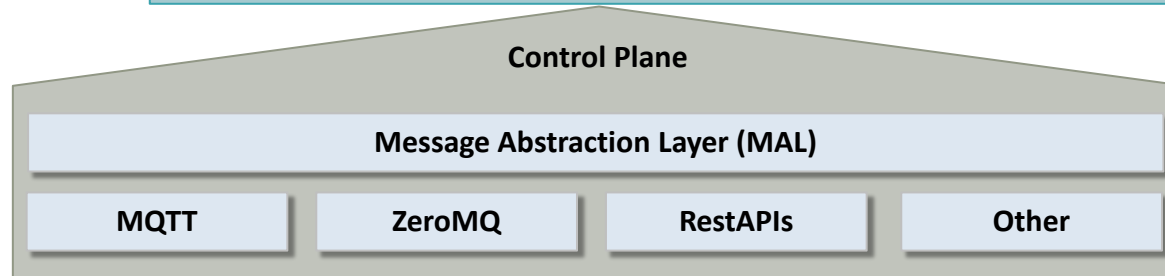


5) Demonstrate
autonomous lab
use case

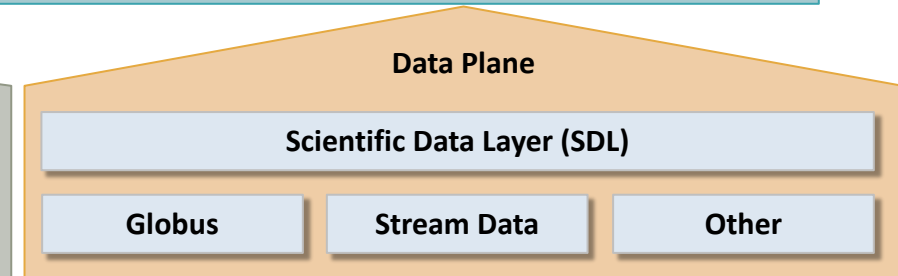
3) Build and
demonstrate
ecosystem
services



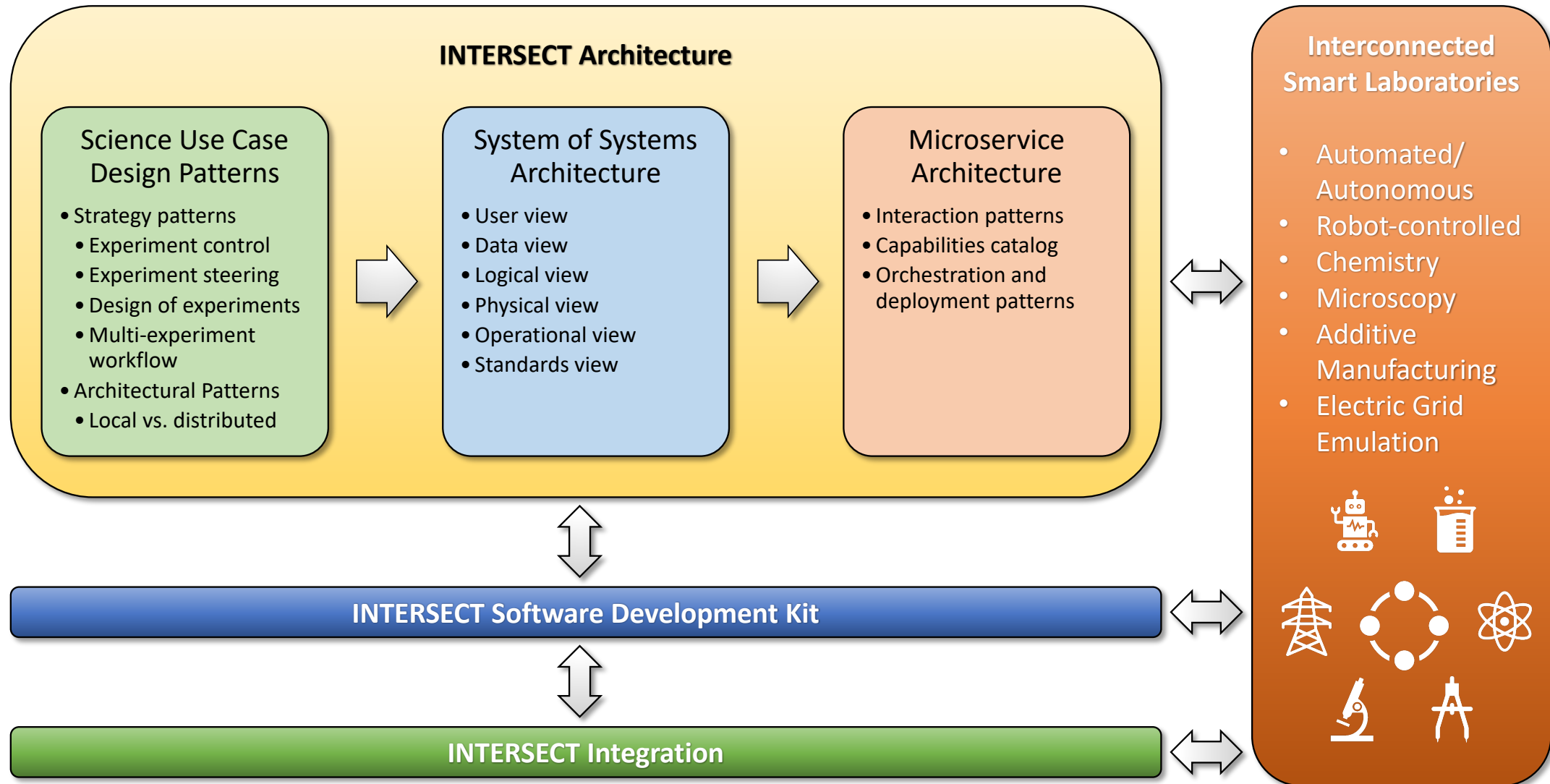
1) Prototype
and Build
Req for a
Common
MAL



2) Integrate
existing
data mgt.
tools

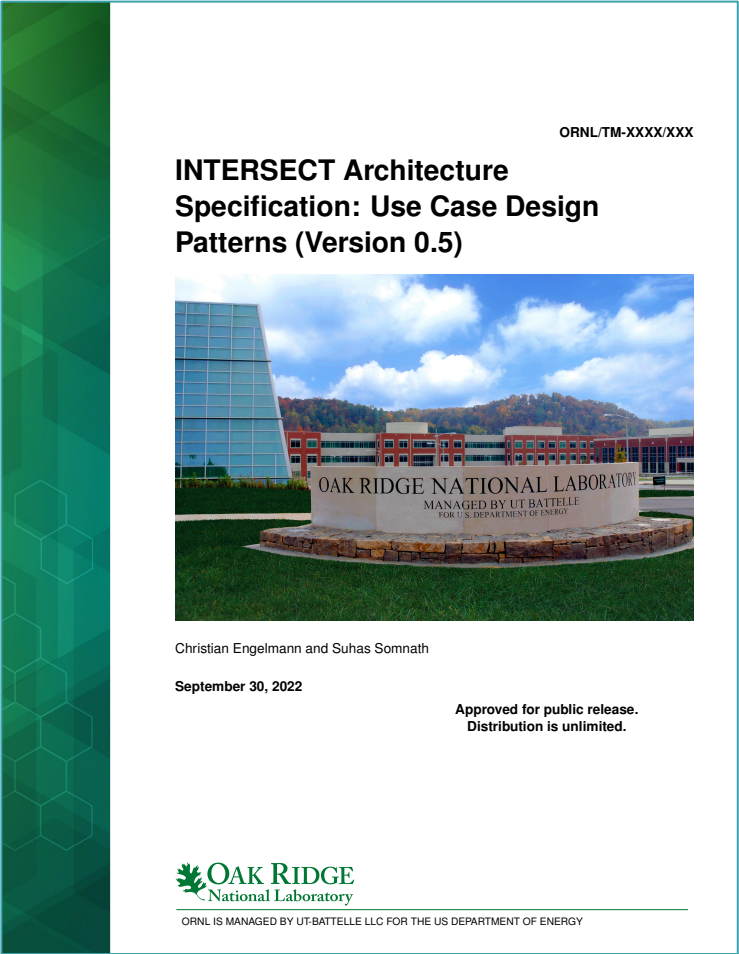


INTERSECT Architecture Overview



Science Use Case Design Pattern Specification

- Abstract descriptions of the involved hardware and software components and their work, data and control flows.



CONTENTS	
LIST OF FIGURES	v
LIST OF TABLES	vii
GLOSSARY	ix
ACKNOWLEDGEMENTS	xi
ABSTRACT	xiii
REVISION RECORD	xv
1. INTRODUCTION	1
2. TERMINOLOGY AND CONCEPTS	2
3. DESIGN PATTERNS FOR SCIENCE USE CASES	4
3.1 INTRODUCTION TO DESIGN PATTERNS	4
3.2 ANATOMY OF A SCIENCE USE CASE DESIGN PATTERN	5
3.3 FORMAT OF A SCIENCE USE CASE DESIGN PATTERN	5
4. CLASSIFICATION OF SCIENCE USE CASE DESIGN PATTERNS	7
4.1 STRATEGY PATTERNS	7
4.2 ARCHITECTURAL PATTERNS	7
5. CATALOG OF SCIENCE USE CASE DESIGN PATTERNS	9
5.1 STRATEGY PATTERNS	9
5.1.1 Experiment Control	9
5.1.2 Experiment Steering	10
5.1.3 Design of Experiments	12
5.1.4 Multi-Experiment Workflow	14
5.2 ARCHITECTURAL PATTERNS	17
5.2.1 Local Experiment Control	17
5.2.2 Remote Experiment Control	19
5.2.3 Local Experiment Steering	21
5.2.4 Remote Experiment Steering	23
5.2.5 Local Design of Experiments	25
5.2.6 Remote Design of Experiments	28
6. BUILDING SOLUTIONS USING SCIENCE USE CASE DESIGN PATTERNS	31
6.1 A STEP-BY-STEP GUIDE	31
6.2 PATTERN COMPOSITIONS	32
REFERENCES	35

Science Use Case Design Patterns: Anatomy

- **Approach: Focus on the control problem**
 - Open vs. closed loop control
 - Single vs. multiple experiment control
 - Steering vs. designing experiments
 - Local vs. remote compute in the loop
- Universal patterns that describe solutions free of implementation details
- Patterns may exclude each other or may be combined with each other
- Described pattern properties:
 - Name, Problem, Context, Forces, Solution, Capabilities, Resulting Context, Related Patterns, Examples, and Known Uses

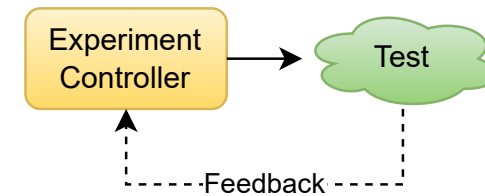


Figure: Single experiment control

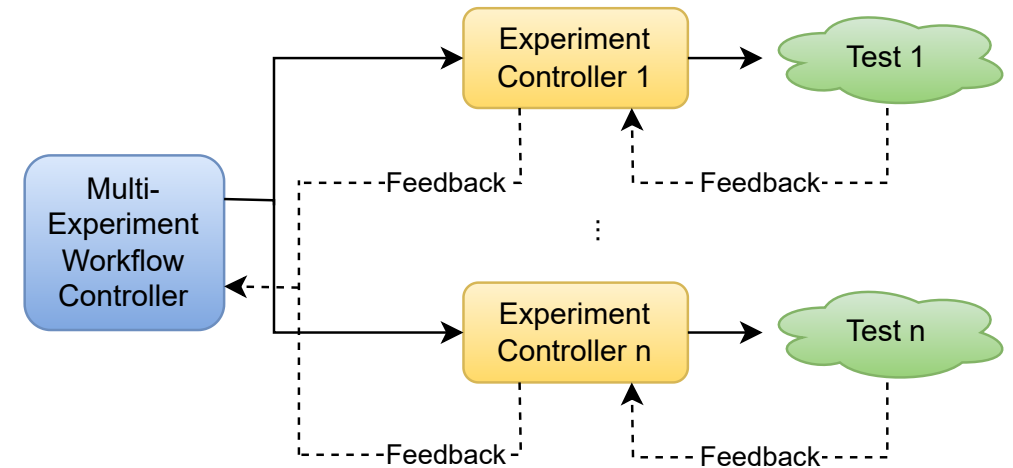


Figure: Multi-experiment control

Science Use Case Design Patterns: Classification

- **Strategy patterns:** High-level solutions with different control features
- **Architectural patterns:** More specific solutions using different hardware/software architectural features

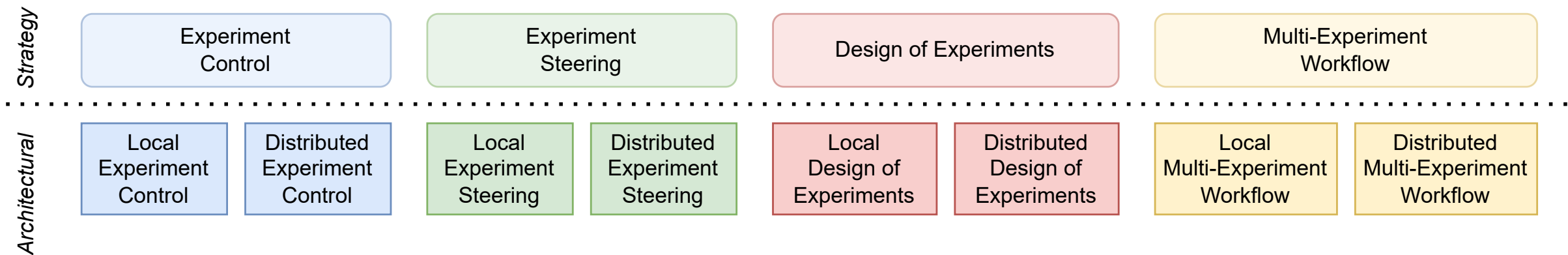
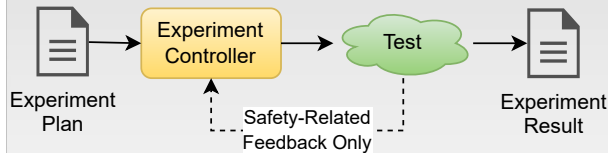


Figure: Pattern classification scheme

Science Use Case Design Patterns: Strategy Patterns

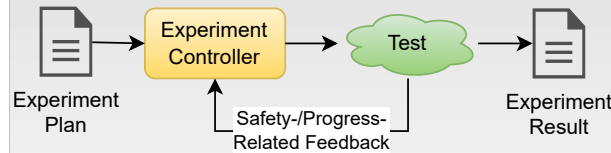
Experiment Control



Executes an existing plan

- Open loop control
- Automated operation

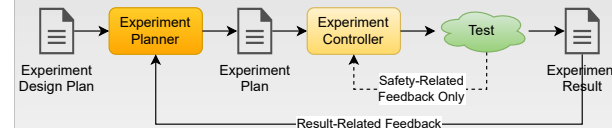
Experiment Steering



Executes an existing plan, depending on progress

- Closed loop control
- Autonomous operation
- Extends patterns:
 - *Experiment Control*

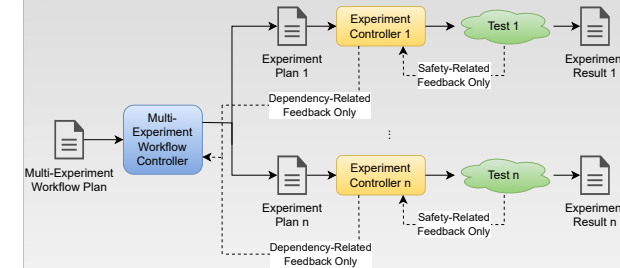
Design of Experiments



Creates/executes a plan, based on prior result

- Closed loop control
- Autonomous operation
- Uses patterns:
 - *Experiment Control*
- May use patterns:
 - *Experiment Steering*

Multi-Experiment Workflow



Executes existing plans (workflow of experiments)

- Open loop control
- Automated operation
- Uses patterns:
 - *Experiment Control*
- May use patterns:
 - *Experiment Steering*
 - *Design of Experiments*

Science Use Case Design Patterns: Architectural Patterns

Local vs. Distributed Experiment Steering

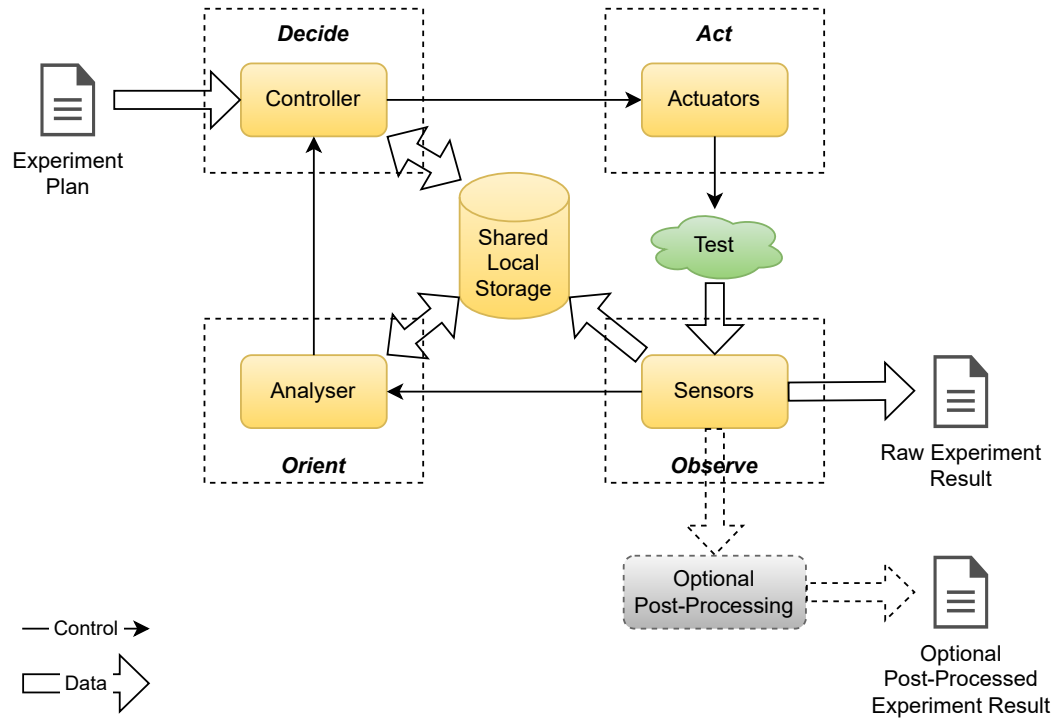


Figure: Local Experiment Steering

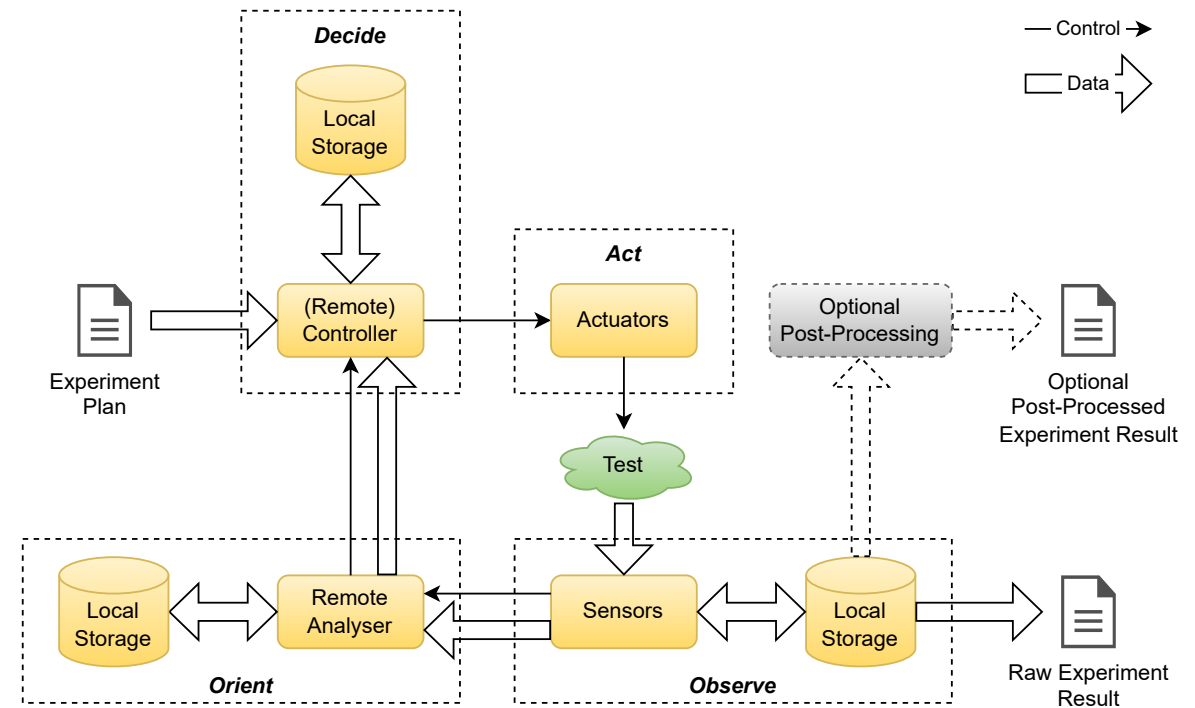


Figure: Distributed Experiment Steering

Science Use Case Design Patterns: Compositions

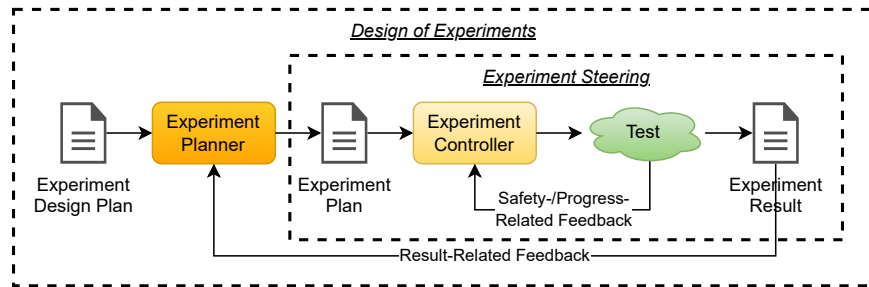


Figure: Strategy pattern composition

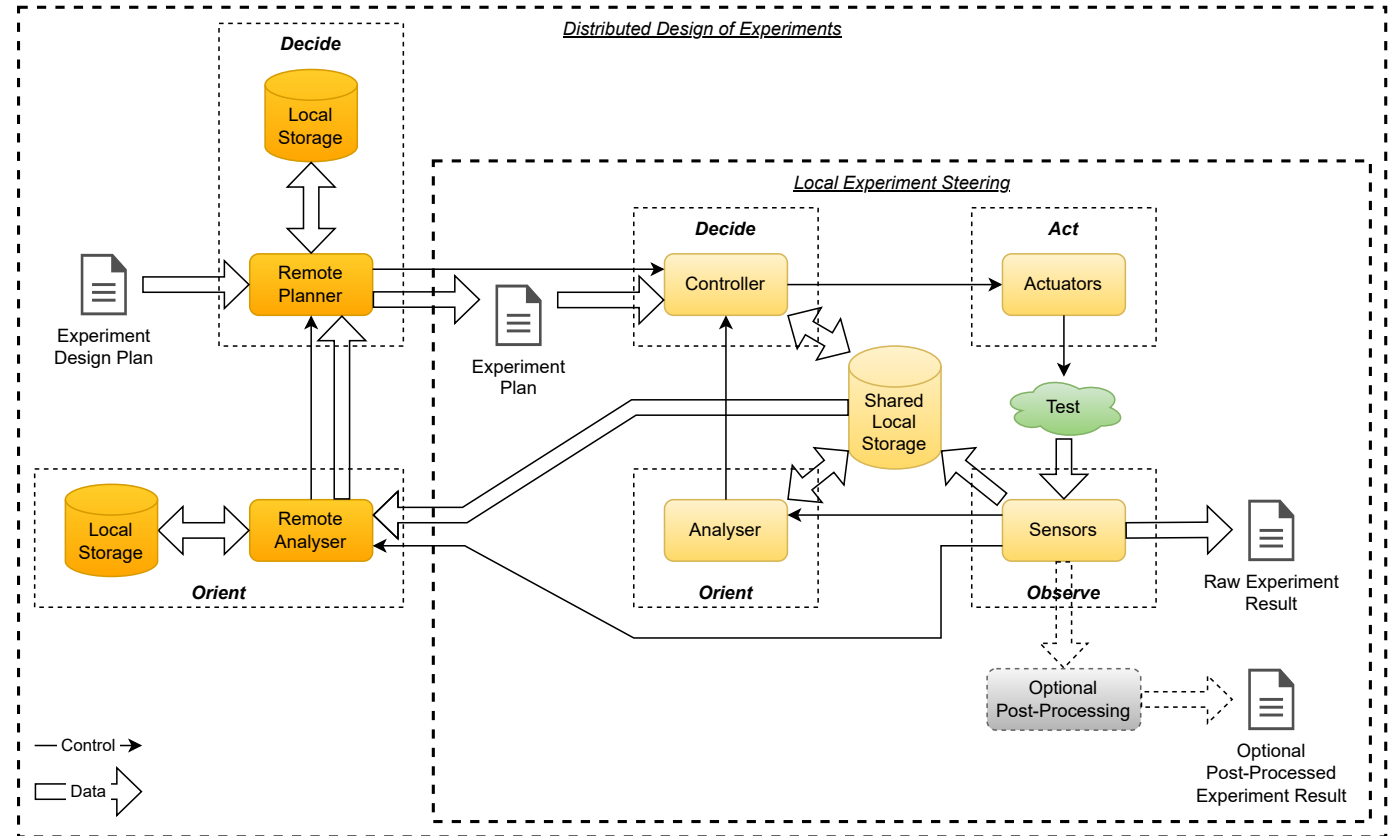


Figure: Architectural pattern composition


System of Systems Architecture Specification

- Detailed design decisions about the involved hardware and software components from different points of view.

ORNL/TM-XXXX/XXX

INTERSECT Architecture


Specification: System of System Architecture (Version 0.5)



Olga A. Kuchar, Swen Boehm, Thomas Naughton, Suhas Somnath, Ben Mintz, Jack Lange, Scott Atchley

September 16, 2022

Approved for public release.
Distribution is unlimited.



OAK RIDGE
National Laboratory

ORNL IS MANAGED BY UT-BATTELLE LLC FOR THE US DEPARTMENT OF ENERGY

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS AND ABBREVIATIONS

INTERSECT TERMINOLOGY

ACKNOWLEDGEMENTS

ABSTRACT

REVISION RECORD

1. INTRODUCTION

1.1 INTRODUCTION

1.2 PURPOSE OF THIS DOCUMENT

1.3 STAKEHOLDER REPRESENTATION

1.4 DOCUMENT SCOPE

1.5 DOCUMENT OVERVIEW

1.6 DOCUMENT MANAGEMENT AND CONFIGURATION CONTROL INFORMATION

2. LOGICAL VIEW

2.1 INTRODUCTION

2.2 SYSTEM CONCEPTS

2.3 SERVICE DESCRIPTION

2.4 SYSTEM OVERVIEW

2.5 SYSTEM OPTIONS

2.6 SYSTEM RESOURCE FLOW REQUIREMENTS

2.7 CAPABILITY INTEGRATION PLANNING

2.8 SYSTEM INTEGRATION MANAGEMENT

2.9 OPERATIONAL PLANNING

3. OPERATIONAL VIEW

3.1 INTRODUCTION

3.2 HIGH-LEVEL OPERATIONAL DIAGRAM

3.3 OPERATIONAL ACTIVITIES

4. USER VIEW

4.1 INTRODUCTION

4.2 USER PERSON TYPE AND ASSOCIATED VIEWS

4.3 OWNER

4.4 OPERATOR / MAINTAINER

4.5 ADMINISTRATOR

5. DATA VIEW

5.1 INTRODUCTION

iii

5.2 CONCEPTUAL DATA MODEL 83

5.3 SEQUENCE DIAGRAMS 88

5.4 ENTITY RELATIONSHIP DATA MODEL 90

5.5 INTERSECT DATA MESSAGING SCHEMA 91

5.6 DESCRIPTIONS FOR THE INTERSECT DATA MESSAGING SCHEMA 91

6. STANDARDS VIEW 97

7. PHYSICAL VIEW 99

7.1 INTRODUCTION 99

7.2 CONCEPTUAL PHYSICAL VIEW 100

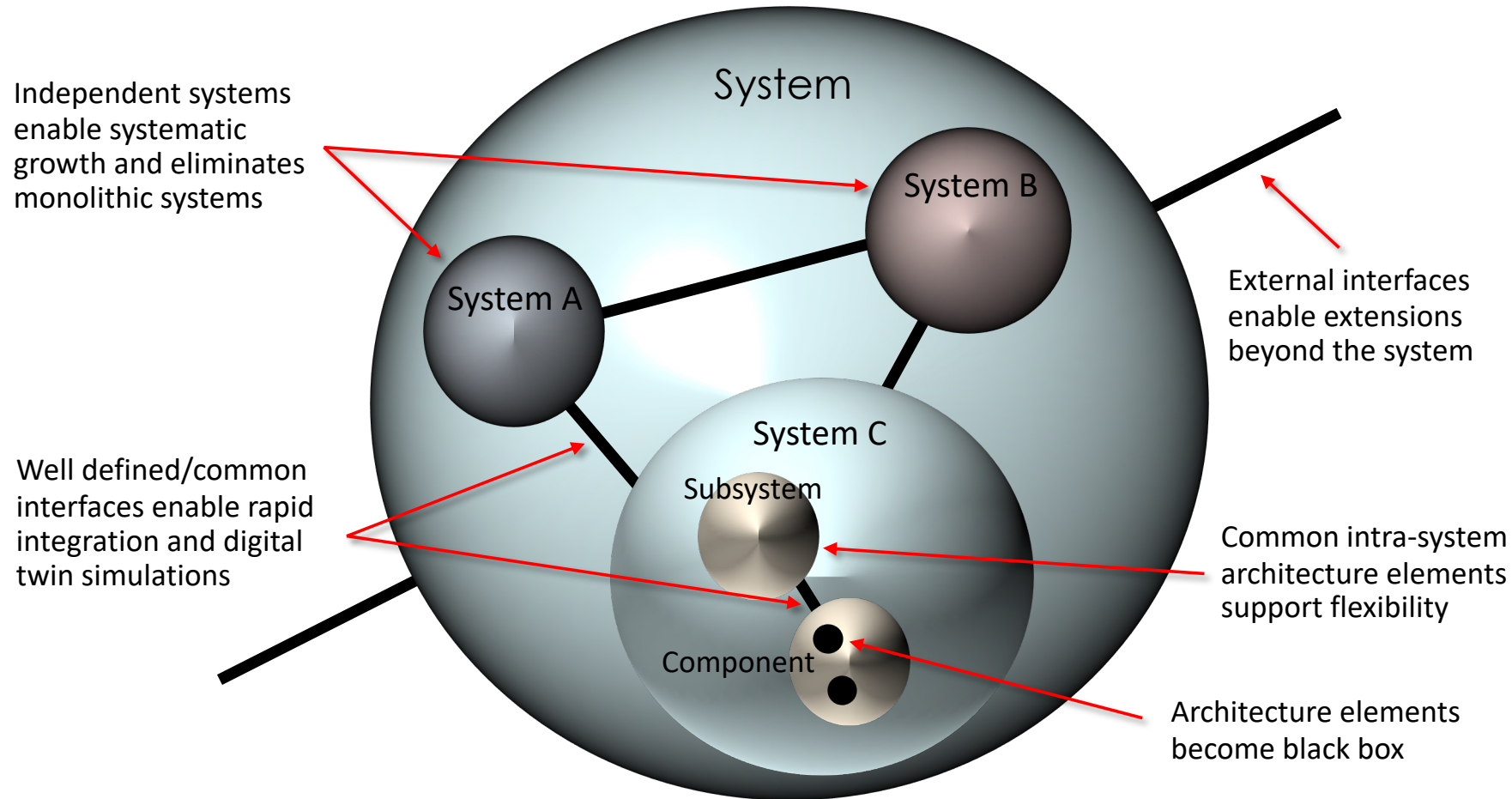
REFERENCES 106

Appendices

A INTERSECT MESSAGE SCHEMA 109

Why System of Systems?

Common Architecture Elements



Common Messages

System

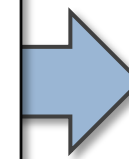
SystemStatus
SystemControlStatus
SystemControlRequest
SystemControlRequestStatus
SystemTask
SystemTaskStatus

Subsystem

SubsystemStatus
SubsystemControlRequest
SubsystemControlRequestStatus
X_Capability
X_CapabilityStatus
X_CapabilityCommand
X_CapabilityCommandStatus
X_CapabilityActivity

Component

ComponentControlStatus
ComponentCommand
ComponentCommandStatus



Enable Scalable, Flexible, and Interoperable Development, Deployment and Operation

System of Systems Architecture Views



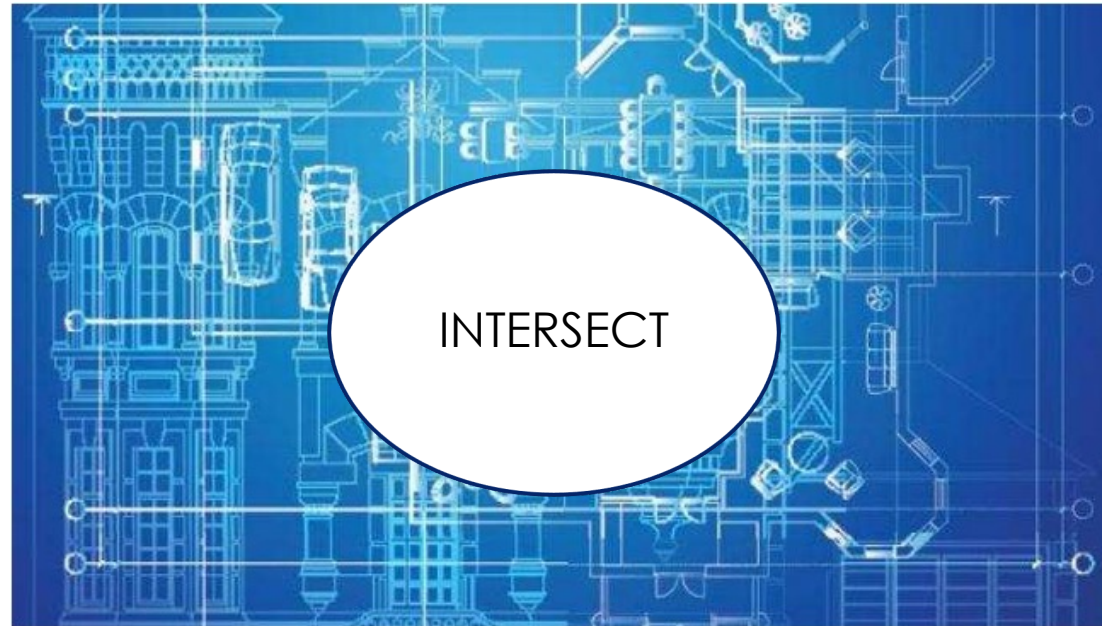
User View



Data View



Operational View



Logical View



Physical View



Standards View

System of Systems Architecture: Logical View



- **Captures the logical composition of systems and their relationships and interactions**
- Includes:
 - Definition of system concepts
 - Definition of system options
 - System resource flow requirements capture
 - Capability integration planning
 - System integration management
 - Operational planning

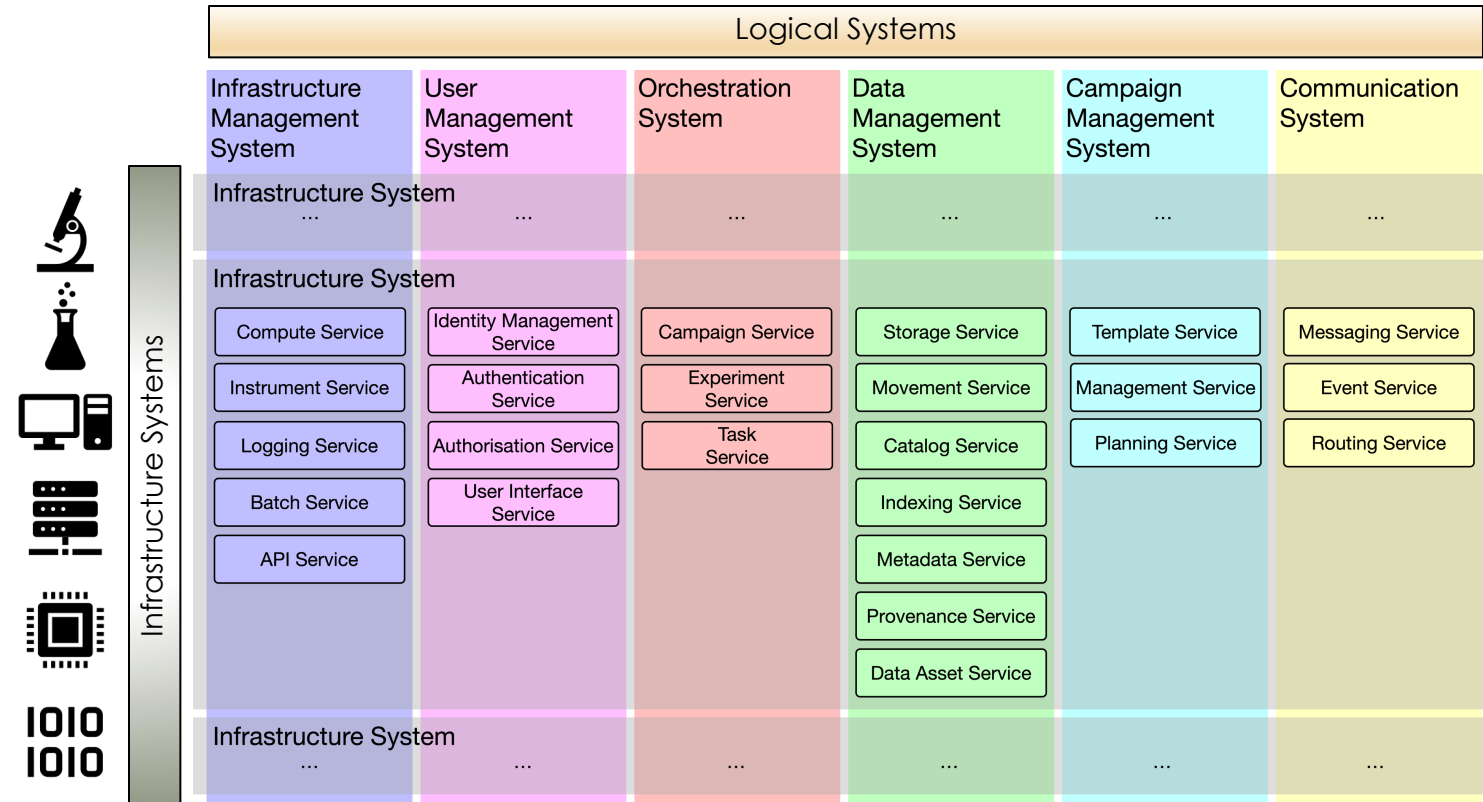


Figure: Relationships between infrastructure and logical systems and their services

System of Systems Architecture: User View



- ***Captures user-facing functionality***
- Does not include system-internal interactions
- Described activities:
 - Logging into dashboard
 - Experiment creation
 - Start experiment
 - Steer experiment
 - Experiment end
- Includes examples for graphical user interfaces

Register for INTERSECT account

First Name	John	Required
Middle Name	C	Optional
Last Name	Doe	Required
Title	Computer Scientist	Recommended
Division	National Center for Computational Sciences	Recommended
Organization	Oak Ridge National Lab	Required
Email address	jcdoe@ornl.gov	Required
Phone number	8651234567	Required
Profile image	Button to upload image	Optional
Interests	Materials; microscopy; energy	Separate tokens by separator like “,”

Note: The default role is “User”. INTERSECT administrators, Owners and operators of Resources are recommended to request change in your role in the User Profile after registering in INTERSECT.

[Register](#)

Microscope

Title: Automated microscopy to identify material compositions

Intent: The intent is to automate a process to determine microscopic material found in samples

Background: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam imperdiet est quis eros rhoncus porta.

Description: Praesent leo felis, gravida vitae dolor eu, elementum mattis odio. Pellentesque finibus, odio cursus cursus facilisis, libero mi placerat ligula, et rutrum dolor nisl quis ante.

Workflow:

```
next_locations = dgx2.user.generate_random_positions()
dgx2.send_data(next_locations, microscope)
next_locs = microscope.recv_data(dgx2)
while next_locs is not None:
    data = microscope.measure(next_locs, configs={...})
    microscope.send_data(data, dgx2)
    last_data = dgx2.recv_data(microscope)
    next_locations = dgx2.user.get_next_posns(last_data, params)
    next_locs = microscope.recv_data(dgx2)
microscope.withdraw_probe()
data manager.save(all data, campaign id, )
```

Recommended Resources: Microscope, dgx2...

Past Campaigns:

Date	User	Title
6/13/21	srivas1	Automated...
7/1/22	kuchar02	Mini Cells...

[Use Template](#) [Cancel](#)


Figure: Examples of graphical user interfaces for different user interactions

Microservice Architecture Specification

- Detailed design decisions about software microservices, including their functionalities, capabilities, compositions, with control, work, and data flows.

ORNL/TM-XXXX/XXX


INTERSECT Architecture
Specification: Microservice
Architecture (Version 0.5)



Michael J. Brim
Christian Engelmann

June 2022

Approved for public release.
Distribution is unlimited.



OAK RIDGE
National Laboratory

ORNL IS MANAGED BY UT-BATTELLE LLC FOR THE US DEPARTMENT OF ENERGY

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS AND ABBREVIATIONS

INTERSECT TERMINOLOGY

ACKNOWLEDGEMENTS

ABSTRACT

REVISION RECORD

1 INTRODUCTION

2 INTERSECT MICROSERVICE ARCHITECTURE

2.1 INTRODUCTION TO MICROSERVICES ARCHITECTURE

2.2 MICROSERVICES ARCHITECTURE IN INTERSECT

3 CLASSIFICATION OF INTERSECT MICROSERVICES

3.1 COMMONALITIES OF INTERSECT MICROSERVICES

3.2 INTERSECT MICROSERVICE CAPABILITIES

3.3 INTERSECT INFRASTRUCTURE MICROSERVICES

3.3.1 General Utility

3.3.2 Communication and Messaging

3.3.3 Computing

3.3.4 Cybersecurity and Identity Management

3.3.5 Data and Information Management

3.3.6 Human-Computer and Human-Machine Interfaces

3.3.7 System Management

3.4 EXPERIMENT-SPECIFIC MICROSERVICES

3.4.1 Experiment Control Microservices

3.4.2 Experiment Data Microservices

3.4.3 Experiment Design Microservices

3.4.4 Experiment Planning Microservices

3.4.5 Experiment Steering Microservices

4 CATALOG OF INTERSECT MICROSERVICES

4.1 INTERSECT INFRASTRUCTURE MICROSERVICES

4.1.1 Communication and Messaging Microservices

4.1.2 Computing Microservices

4.1.3 Cybersecurity Microservices

4.1.4 Data and Information Management Microservices

4.1.5 Human-Computer Interface Microservices

4.1.6 System Management Microservices

4.2 EXPERIMENT-SPECIFIC MICROSERVICES

4.2.1 Experiment Control Microservices

4.2.2 Experiment Data Microservices

4.2.3 Experiment Design Microservices

4.2.4 Experiment Planning Microservices

4.2.5 Experiment Steering Microservices

5 ORCHESTRATION AND DEPLOYMENT OF INTERSECT MICROSERVICES

5.1 MICROSERVICE ORCHESTRATION DESIGN PATTERNS

iii

5.1.1 Asynchronous Messaging vs. RESTful Services

5.1.2 Conductor vs. Choreography

5.2 MICROSERVICE DEPLOYMENT DESIGN PATTERNS

5.2.1 Sidecar Pattern

5.2.2 Ambassador Proxy Pattern

5.2.3 Service Mesh Pattern

REFERENCES

33

34

35

36

36

37

40

Microservice Architecture: Microservice Capabilities

- System consists of
 - Subsystems, resources, and services
- Subsystem consists of
 - Services and resources
- Service consists of
 - Microservice capabilities

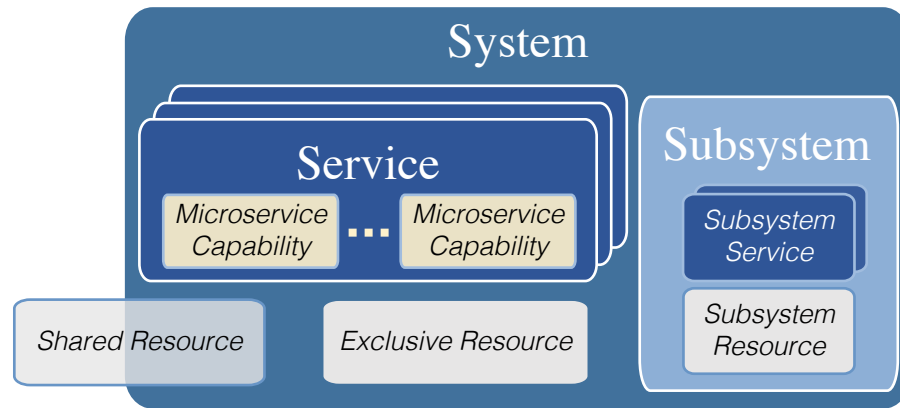


Figure: Systems, subsystems, services, and microservices

Capability: *Unique Capability Name*

Description: A short summary description of the domain of interest for this capability and the provided functionality.

Related Capabilities: Where applicable, provides references to related capabilities.

- **Extends:** A list of base capabilities that the functionality of this capability extends. A service implementing this capability must also implement the base capabilities.
- **Requires:** A list of required capabilities that are necessary to implement the functionality of this capability. The required capabilities are most often provided by other services, but may be implemented in the same service.

Custom Data Type: Where applicable, provides definitions of new data types or structures.

Interactions: Command

- **MethodName()**

Purpose: A short description of the purpose of the current command method.

Command Data: A list of input data for the current method formatted as:

- **dataName (DataType)** : A description of the data, including any format or value constraints.

Interactions: Request-Reply

- **MethodName()**

Purpose: A short description of the purpose of the current request method.

Request Data: A list of input data for the current method formatted as:

- **dataName (DataType)** : A description of the data, including any format or value constraints.

Reply Data: A list of output data for the current method formatted as:

- **dataName (DataType)** : A description of the data, including any format or value constraints.

Interactions: Asynchronous Event

- **EventName**

Purpose: A description of the activity or state change that generates this event.

Event Data: A list of data for the current event formatted as:

- **dataName (DataType)** : A description of the data, including any format or value constraints.

Figure 3-1. Microservice Capability Definition Format

Microservice Architecture: Interaction Patterns

- Command / Acknowledgement
 - Responds immediately
- Request / Reply
 - Responds after fulfilling the request
- Asynchronous Event
 - Status update or event information
- Can be mapped to asynchronous and RESTful client-server communication
 - Microservice architecture does not force a specific implementation

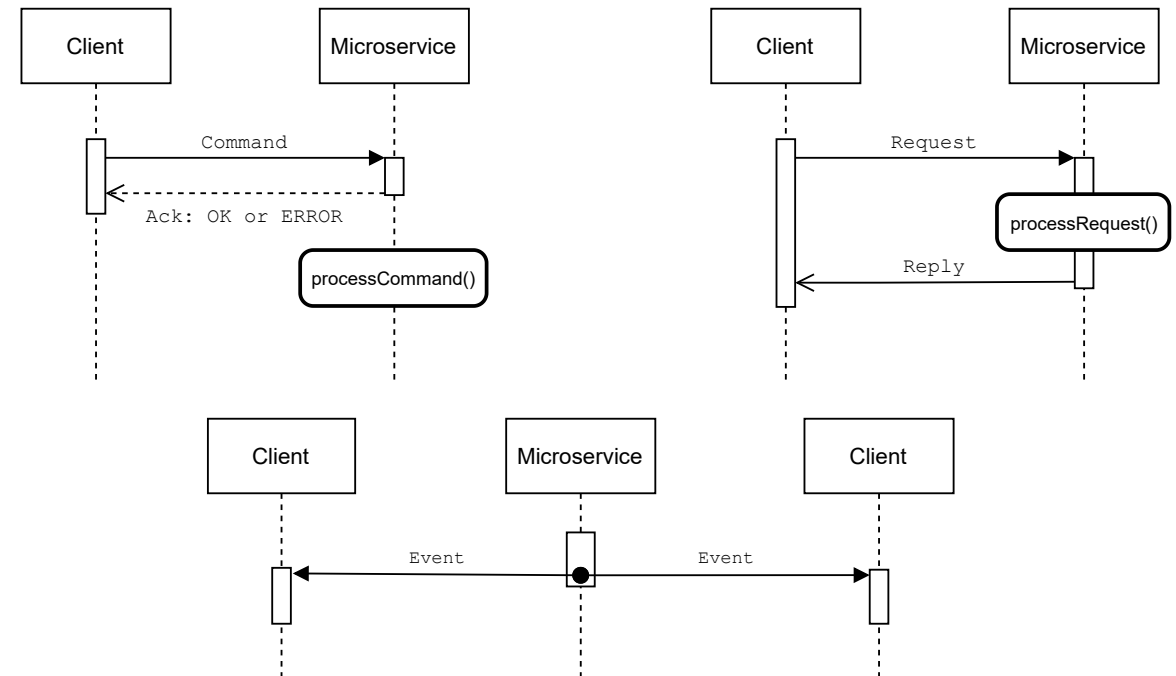


Figure: Command/acknowledgement, request/reply and asynchronous event interaction patterns for microservices

Microservice Architecture: Capabilities Catalog

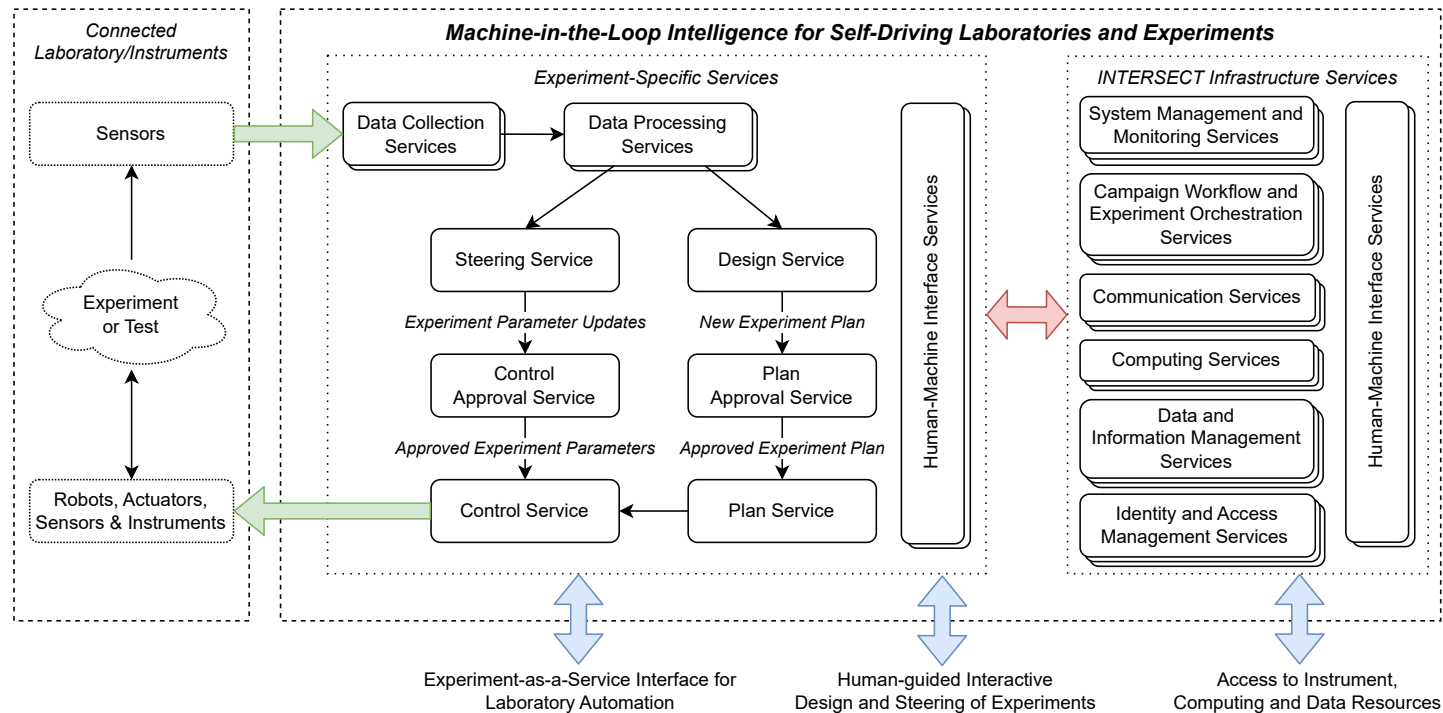
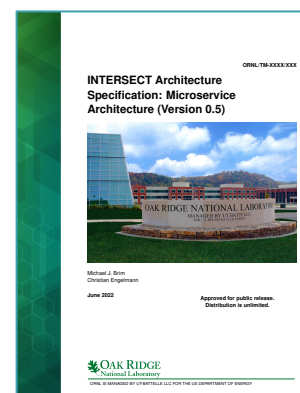


Figure: Experiment-specific and infrastructure services in the context of autonomous experiments and self-driving laboratories

- Example: Data Management
 - Data Transfer
 - File Transfer
 - Block Data Transfer
 - Streaming Data Transfer
 - Multi-party Data Transfer
 - Data Storage
 - File System Storage
 - Key-value Storage
 - Object Storage
 - Relational Database
 - Non-relational Database
 - ...

Current Status

- ***INTERSECT Open Architecture Specification***
 - Design pattern catalog that covers the science use cases in the INTERSECT Initiative
 - System-of-systems architecture specification with elements, communication and interfaces and some command and control and resource triad specifications
 - Initial microservice architecture that covers some INTERSECT science use cases
- ***v0.5 released as 3 ORNL reports in Sept. 2022 (v0.7 latest internal version)***
 - INTERSECT Architecture: Use Case Design Patterns
 - INTERSECT Architecture: System of Systems Architecture
 - INTERSECT Architecture: Microservices Architecture



INTERSECT Architecture Demonstration

ORNL/TM-XXXX/XXX

INTERSECT Architecture Specification: Use Case Design Patterns (Version 0.5)

CONTENTS

LIST OF FIGURES	v
LIST OF TABLES	vii
GLOSSARY	ix
ACKNOWLEDGEMENTS	xi
ABSTRACT	xiii
REVISION RECORD	xv
1. INTRODUCTION	1
2. TERMINOLOGY AND CONCEPTS	2
3. DESIGN PATTERNS FOR SCIENCE USE CASES	4
3.1 INTRODUCTION TO DESIGN PATTERNS	4
3.2 ANATOMY OF A SCIENCE USE CASE DESIGN PATTERN	5
3.3 FORMAT OF A SCIENCE USE CASE DESIGN PATTERN	5
4. CLASSIFICATION OF SCIENCE USE CASE DESIGN PATTERNS	7
4.1 STRATEGY PATTERNS	7
4.2 ARCHITECTURAL PATTERNS	7
5. CATALOG OF SCIENCE USE CASE DESIGN PATTERNS	9
5.1 STRATEGY PATTERNS	9
5.1.1 Experiment Control	9
5.1.2 Experiment Steering	10
5.1.3 Design of Experiments	12
5.1.4 Multi-Experiment Workflow	14
5.2 ARCHITECTURAL PATTERNS	17
5.2.1 Local Experiment Control	17
5.2.2 Remote Experiment Control	19
5.2.3 Local Experiment Steering	21
5.2.4 Remote Experiment Steering	23
5.2.5 Local Design of Experiments	25
5.2.6 Remote Design of Experiments	28
6. BUILDING SOLUTIONS USING SCIENCE USE CASE DESIGN PATTERNS	31
6.1 A STEP-BY-STEP GUIDE	31
6.2 PATTERN COMPOSITIONS	32
REFERENCES	35

ORNL/TM-XXXX/XXX

INTERSECT Architecture Specification: System of System Architecture (Version 0.5)

CONTENTS

LIST OF FIGURES	v
LIST OF TABLES	vii
ACRONYMS AND ABBREVIATIONS	ix
INTERSECT TERMINOLOGY	xi
ACKNOWLEDGEMENTS	xiii
ABSTRACT	xv
REVISION RECORD	xvii
1. INTRODUCTION	1
1.1 INTRODUCTION	1
1.2 PURPOSE OF THIS DOCUMENT	3
1.3 STAKEHOLDER REPRESENTATION	3
1.4 DOCUMENT SCOPE	5
1.5 DOCUMENT OVERVIEW	5
1.6 DOCUMENT MANAGEMENT AND CONFIGURATION CONTROL INFORMATION	6
2. LOGICAL VIEW	7
2.1 INTRODUCTION	7
2.2 SYSTEM CONCEPTS	7
2.3 SERVICE DESCRIPTION	10
2.4 SYSTEM OVERVIEW	12
2.5 SYSTEM OPTIONS	12
2.6 SYSTEM RESOURCE FLOW REQUIREMENTS	14
2.7 CAPABILITY INTEGRATION PLANNING	14
2.8 SYSTEM INTEGRATION MANAGEMENT	14
2.9 OPERATIONAL PLANNING	14
3. OPERATIONAL VIEW	23
3.1 INTRODUCTION	23
3.2 HIGH-LEVEL OPERATIONAL DIAGRAM	24
3.3 OPERATIONAL ACTIVITIES	26
4. USER VIEW	37
4.1 INTRODUCTION	37
4.2 USER PERSON TYPE AND ASSOCIATED VIEWS	39
4.3 OWNER	64
4.4 OPERATOR / MAINTAINER	71
4.5 ADMINISTRATOR	74
5. DATA VIEW	83
5.1 INTRODUCTION	83

5.2 CONCEPTUAL DATA MODEL	83
5.3 SEQUENCE DIAGRAMS	88
5.4 ENTITY RELATIONSHIP DATA MODEL	90
5.5 INTERSECT DATA MESSAGING SCHEMA	91
5.6 DESCRIPTIONS FOR THE INTERSECT DATA MESSAGING SCHEMA	91
6. STANDARDS VIEW	97
7. PHYSICAL VIEW	99
7.1 INTRODUCTION	99
7.2 CONCEPTUAL PHYSICAL VIEW	100
REFERENCES	106

Appendices

A INTERSECT MESSAGE SCHEMA	109
----------------------------	-----

ORNL/TM-XXXX/XXX

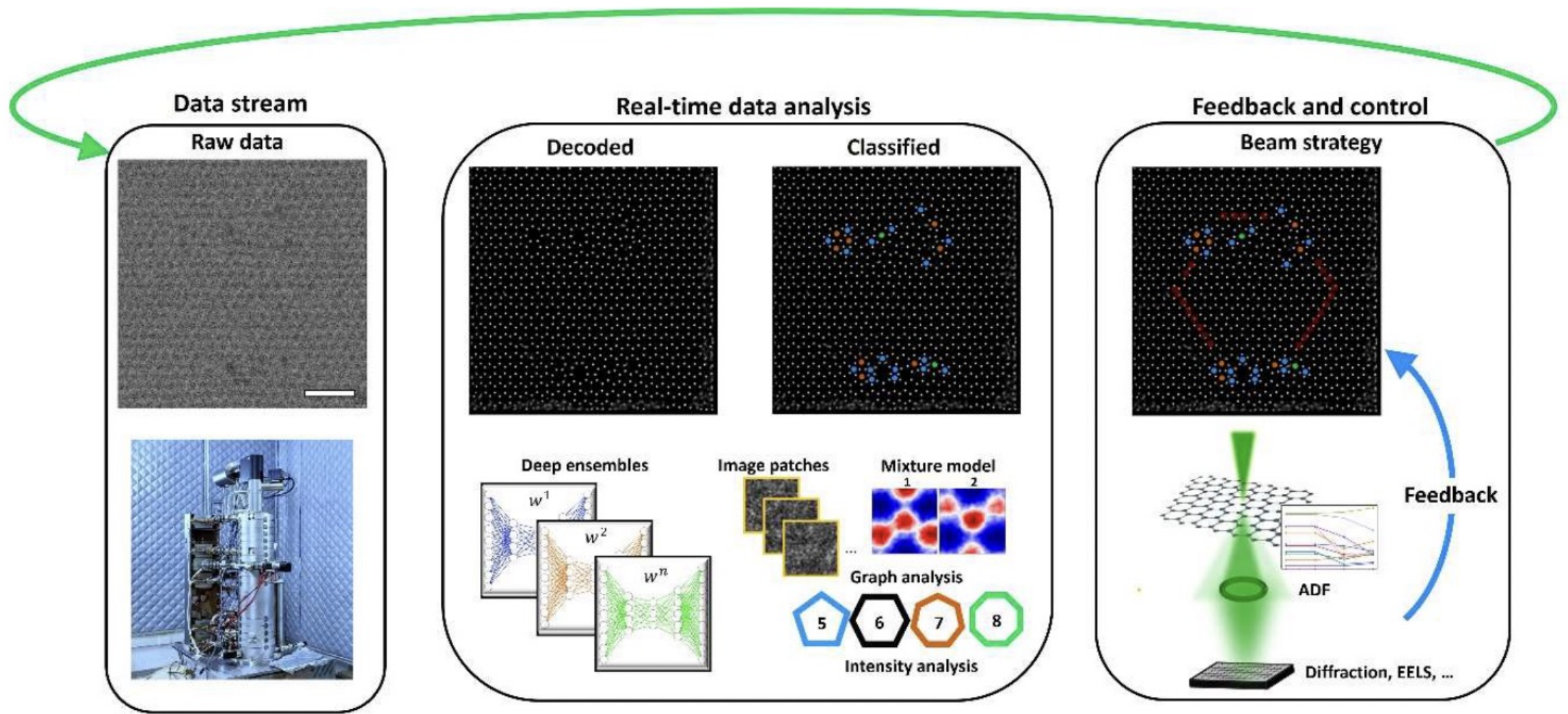
INTERSECT Architecture Specification: Microservice Architecture (Version 0.5)

CONTENTS

LIST OF FIGURES	v
LIST OF TABLES	vii
ACRONYMS AND ABBREVIATIONS	ix
INTERSECT TERMINOLOGY	xi
ACKNOWLEDGEMENTS	xiii
ABSTRACT	xv
REVISION RECORD	xvii
1. INTRODUCTION	1
2. INTERSECT MICROSERVICE ARCHITECTURE	2
2.1 INTRODUCTION TO MICROSERVICES ARCHITECTURE	2
2.2 MICROSERVICES ARCHITECTURE IN INTERSECT	3
3. CLASSIFICATION OF INTERSECT MICROSERVICES	5
3.1 COMMONALITIES OF INTERSECT MICROSERVICES	5
3.2 INTERSECT MICROSERVICE CAPABILITIES	6
3.3 INTERSECT INFRASTRUCTURE MICROSERVICES	7
3.3.1 General Utility	8
3.3.2 Communication and Messaging	11
3.3.3 Computing	11
3.3.4 Cybersecurity and Identity Management	19
3.3.5 Data and Information Management	20
3.3.6 Human-Computer and Human-Machine Interfaces	25
3.3.7 System Management	25
3.4 EXPERIMENT-SPECIFIC MICROSERVICES	31
3.4.1 Experiment Control Microservices	31
3.4.2 Experiment Data Microservices	31
3.4.3 Experiment Design Microservices	31
3.4.4 Experiment Planning Microservices	31
3.4.5 Experiment Steering Microservices	31
4. CATALOG OF INTERSECT MICROSERVICES	32
4.1 INTERSECT INFRASTRUCTURE MICROSERVICES	32
4.1.1 Communication and Messaging Microservices	32
4.1.2 Computing Microservices	32
4.1.3 Cybersecurity Microservices	32
4.1.4 Data and Information Management Microservices	32
4.1.5 Human-Computer Interface Microservices	32
4.1.6 System Management Microservices	32
4.2 EXPERIMENT-SPECIFIC MICROSERVICES	32
4.2.1 Experiment Control Microservices	32
4.2.2 Experiment Data Microservices	32
4.2.3 Experiment Design Microservices	32
4.2.4 Experiment Planning Microservices	32
4.2.5 Experiment Steering Microservices	32
5. ORCHESTRATION AND DEPLOYMENT OF INTERSECT MICROSERVICES	33
5.1 MICROSERVICE ORCHESTRATION DESIGN PATTERNS	33

5.1.1 Asynchronous Messaging vs. RESTful Services	33
5.1.2 Conductor vs. Choreography	34
5.2 MICROSERVICE DEPLOYMENT DESIGN PATTERNS	35
5.2.1 Sidecar Pattern	36
5.2.2 Ambassador Proxy Pattern	36
5.2.3 Service Mesh Pattern	37
REFERENCES	40

Autonomous Microscopy: Science Goal



Autonomous Microscopy: Science Use Case Design Patterns

- Strategy Pattern

- Experiment Steering
- Control of an **ongoing** STEM experiment via analysis of periodic experimental data

- Architectural Pattern

- Distributed Experiment Steering
- Local control of an **ongoing** STEM experiment via **remote** analysis of periodic experimental data

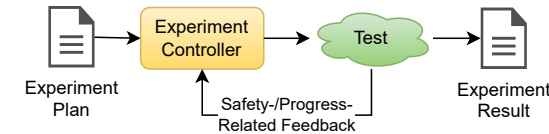


Figure: Strategy pattern: Experiment Steering

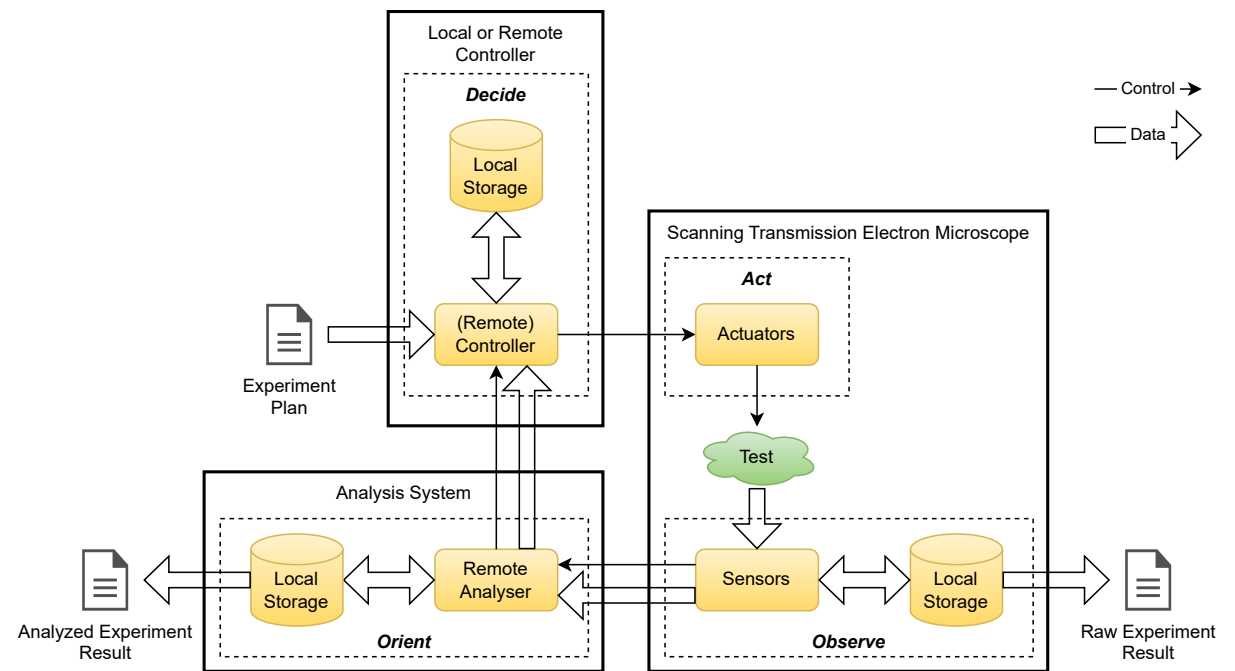
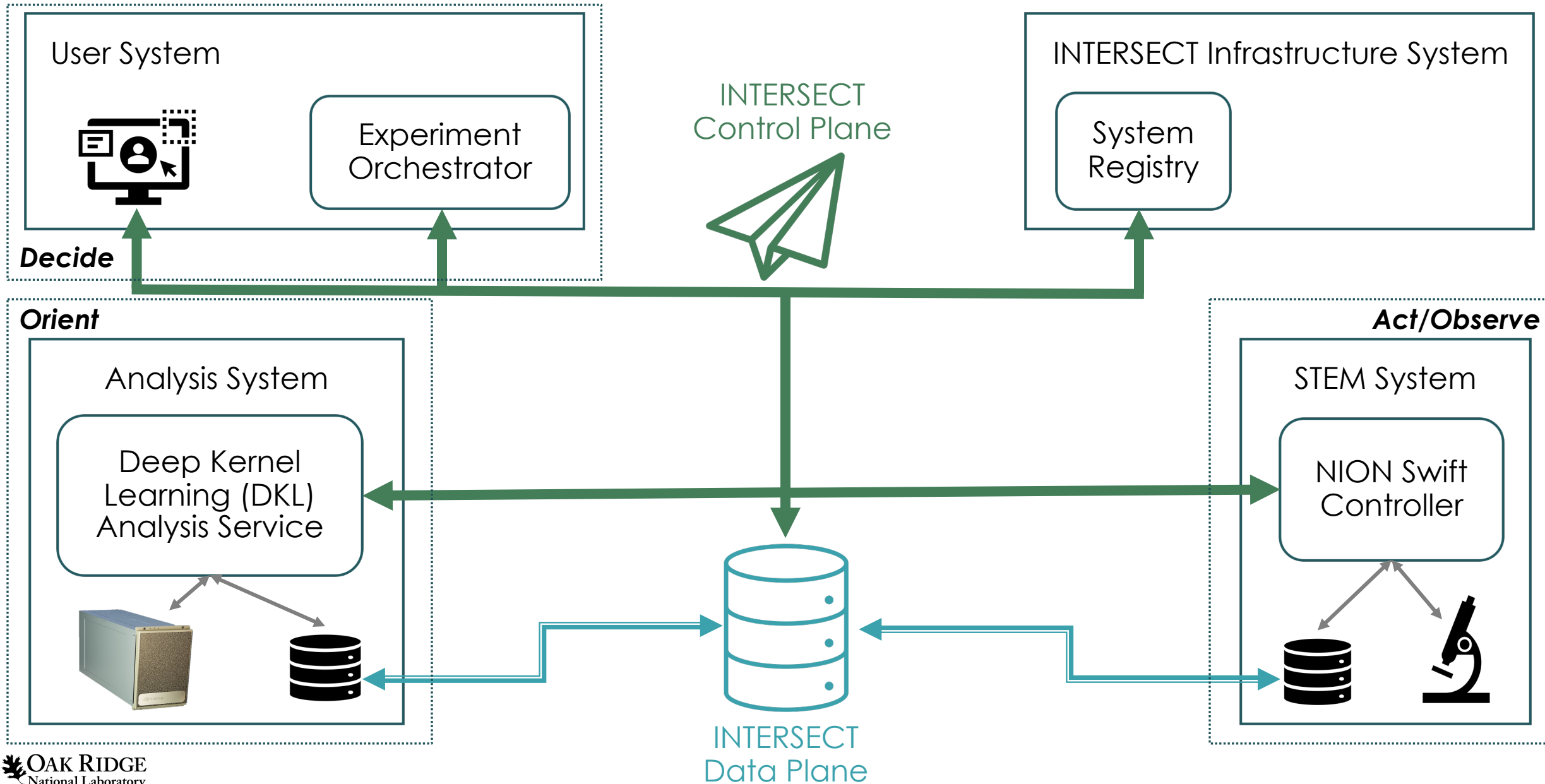


Figure: Architectural pattern: Remote Experiment Steering

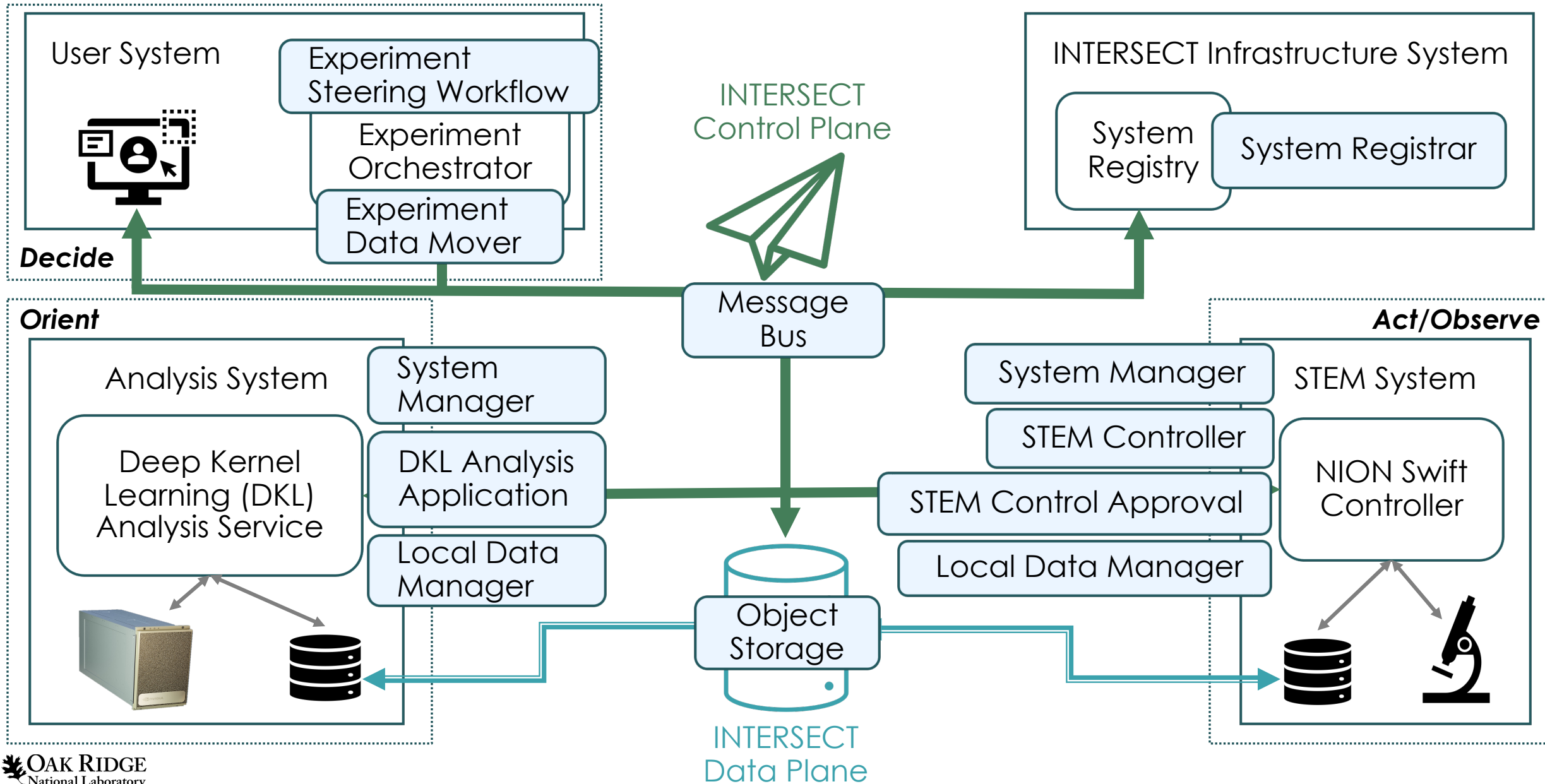
System of Systems Architecture

KEY: System Service



Microservice Architecture

KEY: Microservice Capability



Questions?