OAK RIDGE National Laboratory

Scientific Ecosystems: A complex network of interconnected scientific systems

Rob Moore

Co-Director, Interconnected Science Ecosystem (INTERSECT)

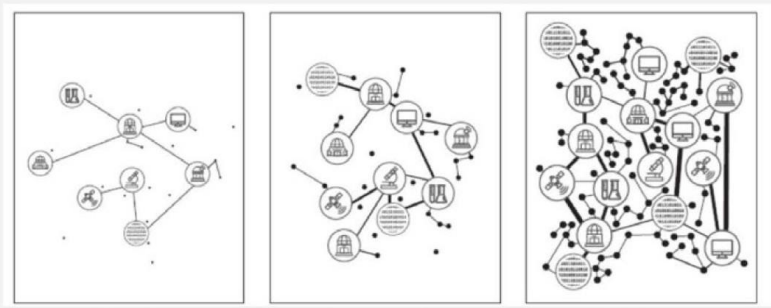
December 6, 2023 (Last Day of Class!!!!)

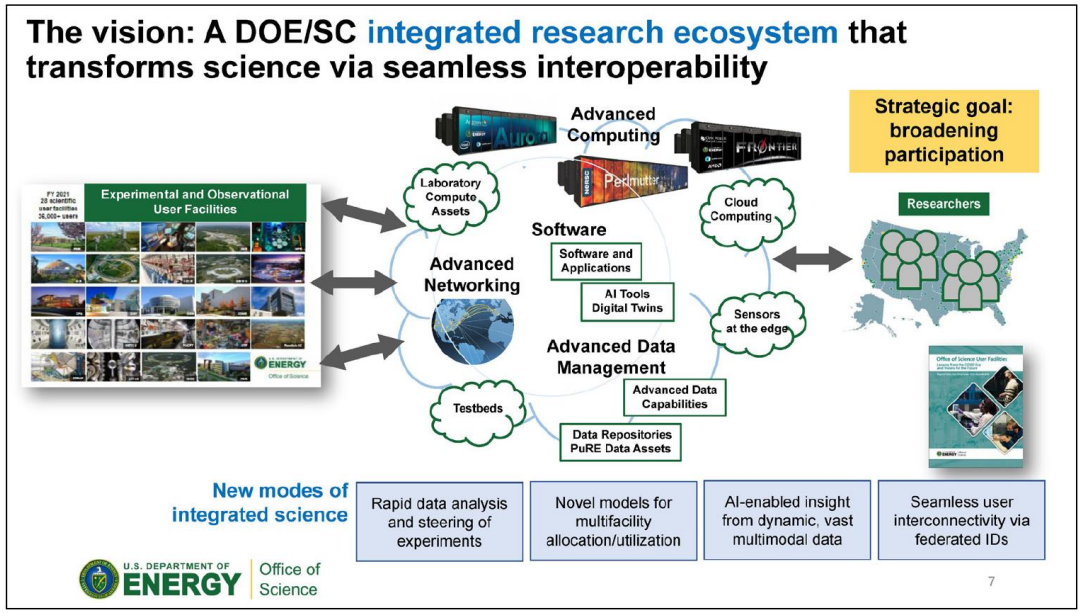
ORNL is managed by UT-Battelle, LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

Looking for a Model of the Future

"R&D continues to shift from smaller to bigger science, driven in large part by advances in computing and other research cyberinfrastructure, which interlink[s] research data, analytics, … and experimental instrumentation."

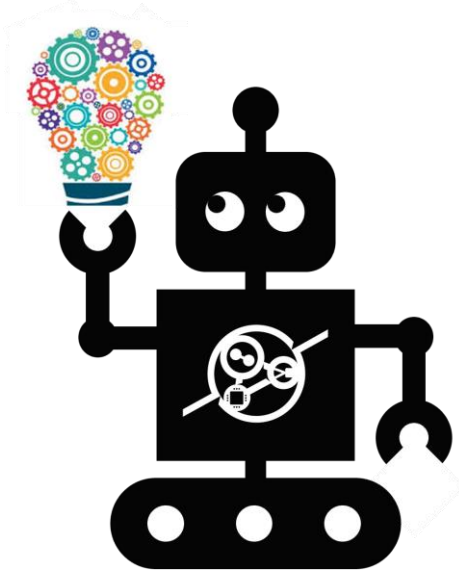


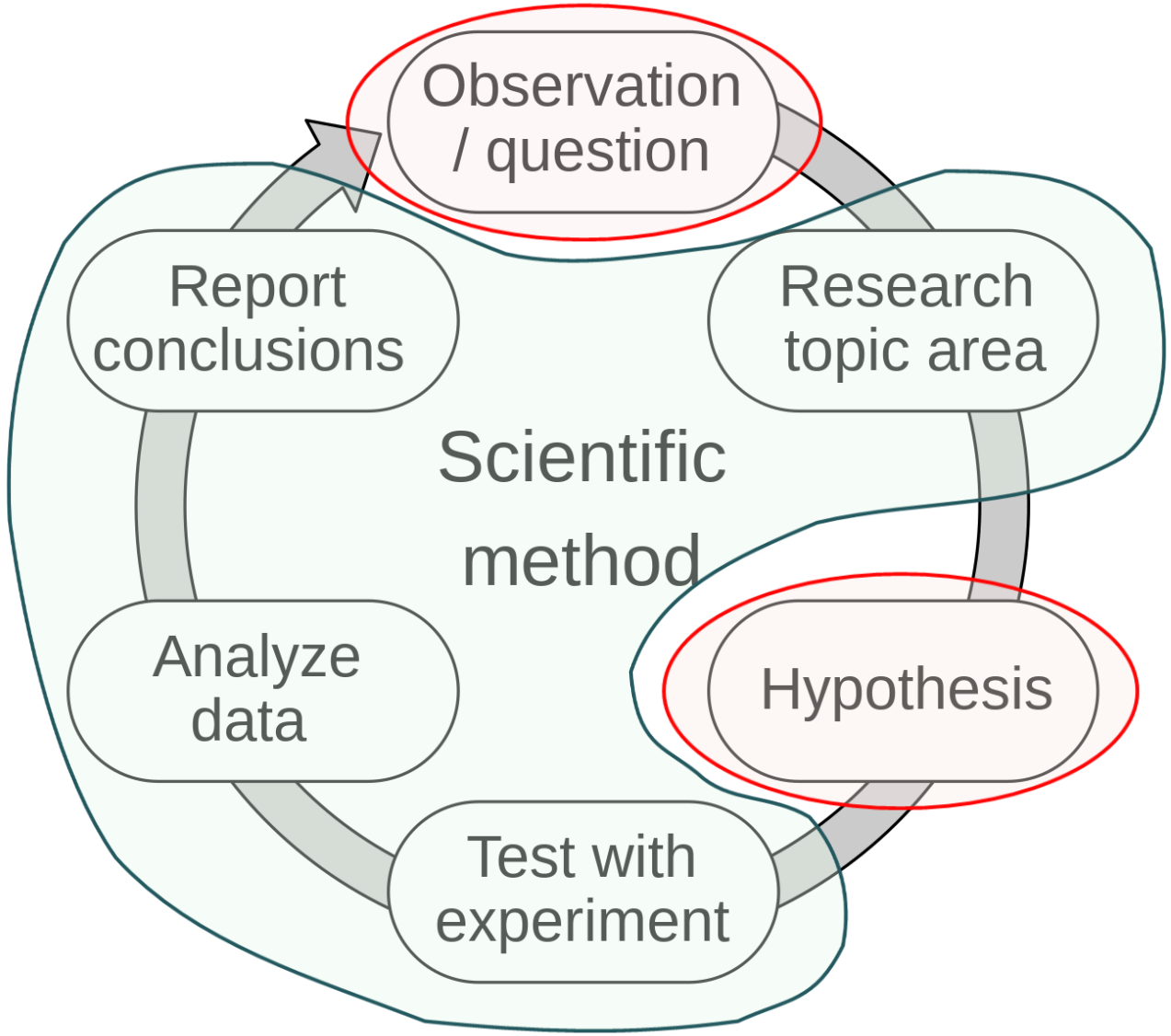


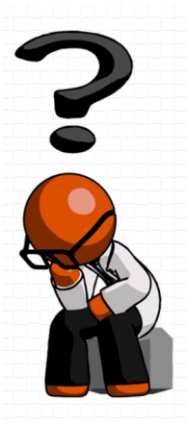
OAK RIDGE National Laboratory

INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

Let's start someplace that we can all understand







A Simple Example

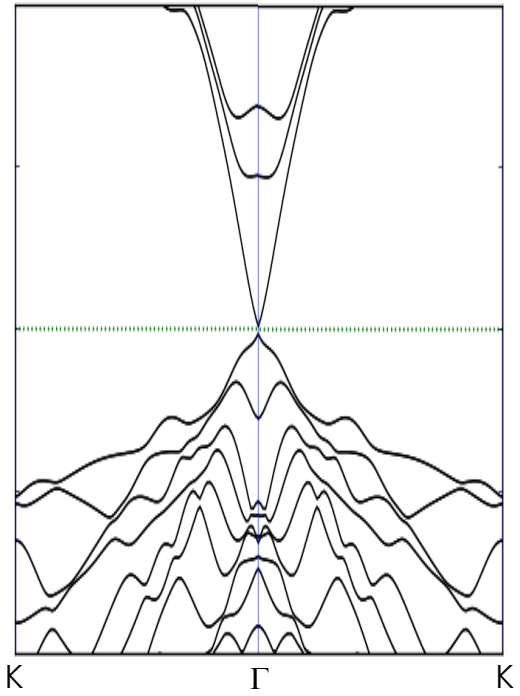
DFT – Density Functional Theory

- First principles calculations…(*with a few tunable parameters*)
- Calculates band dispersions in momentum space, i.e. spaghetti plots
- Calculates band orbital character for band dispersions

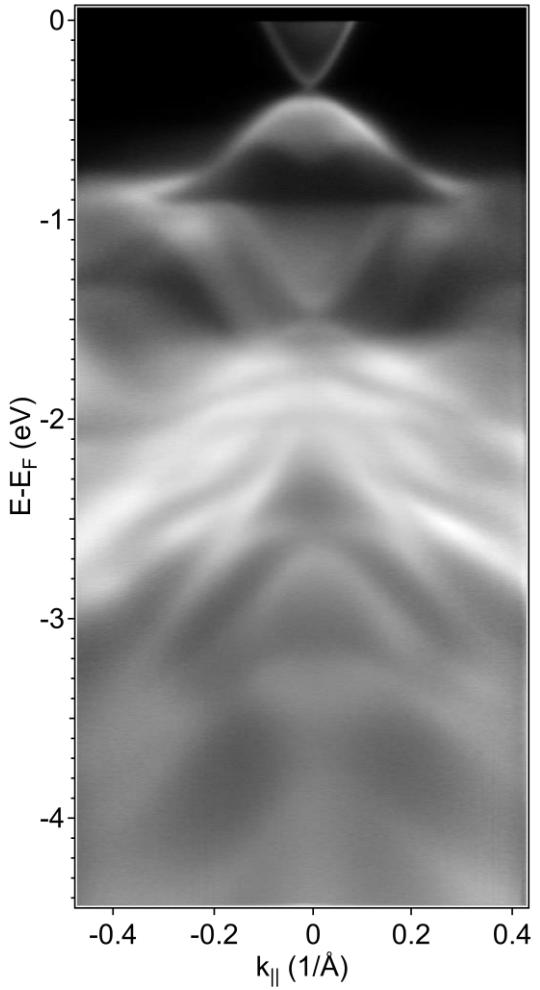
ARPES – Angle Resolved Photoemission Spectroscopy

- Measures the energy and momentum of electrons in materials
- Yields dispersion plots directly comparable to DFT
- Has polarization dependent matrix elements for orbital sensitivity

DFT of Bi_2Se_3

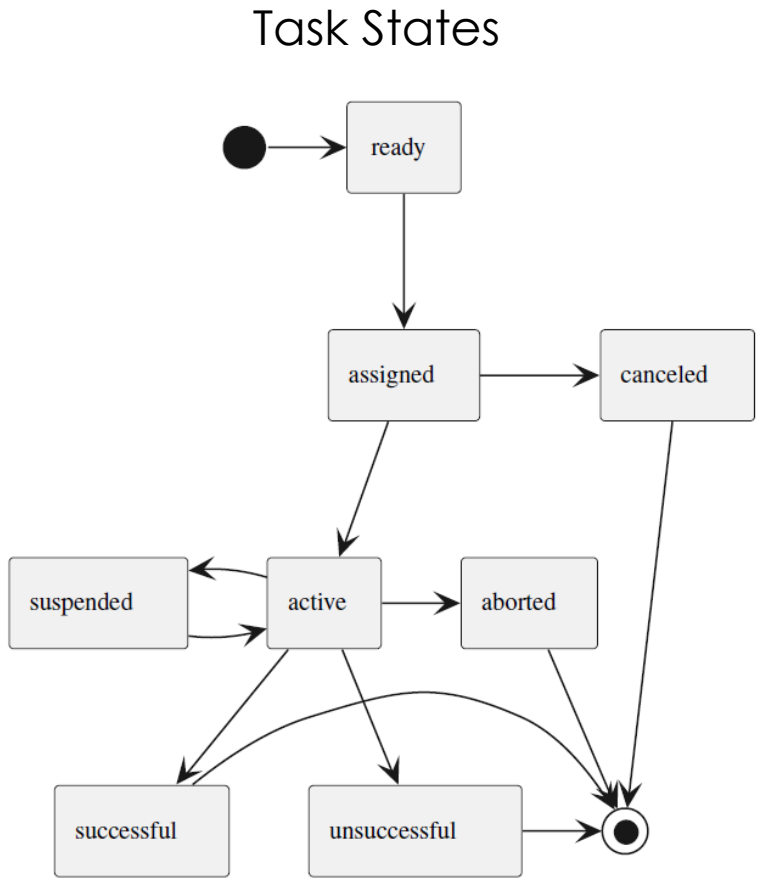


ARPES of Bi_2Se_3



A Simple Example

```
# This script is a job server.
JobServerVersion="v1.1"

# It checks the JOBLIST every 30 minutes
# and runs the next job on the list.
#
# To stop, have 'stop' as the next job
# To clean the temp directory have 'clean' as the next job

# Set Environment Variables
#CMDPrefix="mpirun -np 1 env OMP_NUM_THREADS=1 /home/lg2/Documents/DFT_Codes/qe-6.7/bin"
CMDPrefix="mpirun --use-hwthread-cpus /home/lg2/Documents/DFT_Codes/qe-6.7/bin"
TMP_DIR="/home/lg2/Documents/temp"

# Reset the status log
rm -f JOBSTATUS.LOG
echo "PWSCF Job Server $JobServerVersion" >JOBSTATUS.LOG
echo "Starting job search." >>JOBSTATUS.LOG
echo "" >>JOBSTATUS.LOG

# Reset screen
clear
echo "PWSCF Job Server $JobServerVersion"
echo
echo "Starting DFT calculations:"
echo "Please do not interrupt:"
echo
echo "Current status also in JOBSTATUS.LOG"
echo "Add jobs to JOBLIST"
echo
echo "Starting job search."
echo

# Read in jobs from JOBLIST file
Jobs=( $(cat "JOBLIST") )

# Count number of elements in JOBLIST
ElementCount=${#Jobs[@]}

# We have to skip the first 40 fields in JOBLIST
# because they are the header
Index=49
JobIndex=1

# Set up while loop to continue until 'stop'
# is read in the file
while [ ${Jobs[$Index]} != stop ]
do

# Run Job or wait to check JOBLIST again
if [ $Index -lt $ElementCount ] ; then

# Read Command and check to see if it is 'clean' command
CleanFlag=${Jobs[$Index]}
if [ $CleanFlag == 'clean' ] ; then
echo "Cleaning $TMP_DIR..."
echo
echo "Cleaning $TMP_DIR..." >>JOBSTATUS.LOG
echo >>JOBSTATUS.LOG
rm -rf $TMP_DIR/*
NextIndex=`expr $Index + 1`
Index=$NextIndex
fi

if [ ${Jobs[$Index]} == 'stop' ] ; then
break
fi

# Run Job LOOP
A=$Index
B=`expr $Index + 1`
C=`expr $Index + 2`
NextIndex=`expr $Index + 3`

echo "Starting Job $JobIndex: ${Jobs[$A]} < ${Jobs[$B]} > ${Jobs[$C]}"
echo "Starting Job $JobIndex: ${Jobs[$A]} < ${Jobs[$B]} > ${Jobs[$C]}" >>JOBSTATUS.LOG
echo -n "Start:"; date
echo -n "Start:" >>JOBSTATUS.LOG
date >>JOBSTATUS.LOG
echo "working......"
```



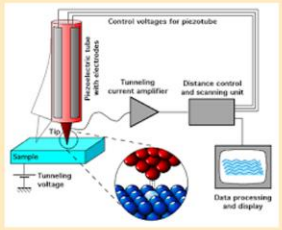
- By nature, science involves repetitive tasks
- By nature, we typically don't like repetitive tasks

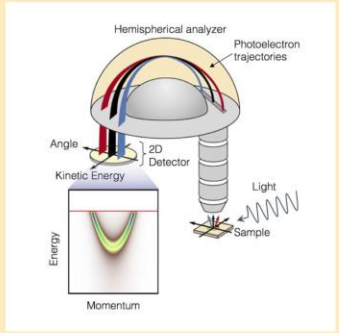
A Simple Example

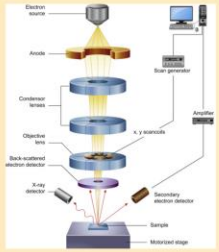
Micro-Ecosystem Demonstration

Connecting the Pieces

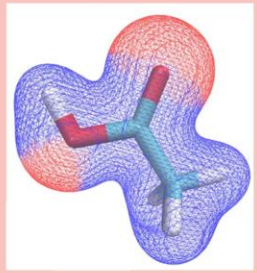
Characterization

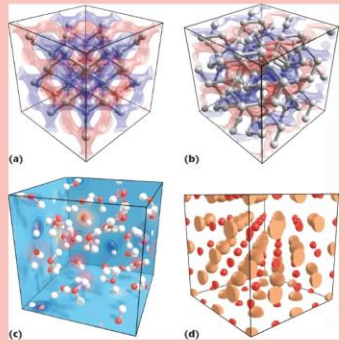






Theory





Simulation

AI Algorithms

nD Data

ML Descriptor

Edge Computing

INTERSECT

Edge Computing

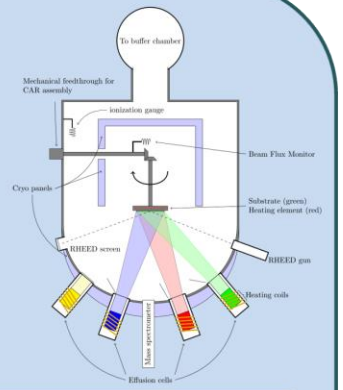
Growth Parameter

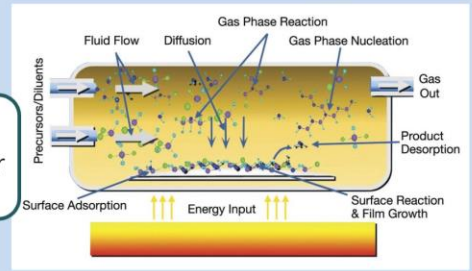
Orchestration, Workflow

Communication

Fabrication

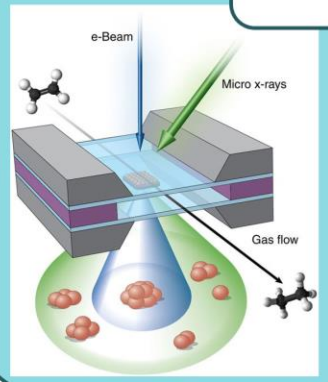
Synthesis

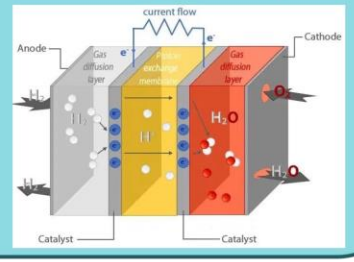




Automation

in operando, proof of principle





OAK RIDGE National Laboratory

INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

What We All Really Want

Hey Google

alexa

Siri

"Hey Oakley,
Help me solve a science problem!"

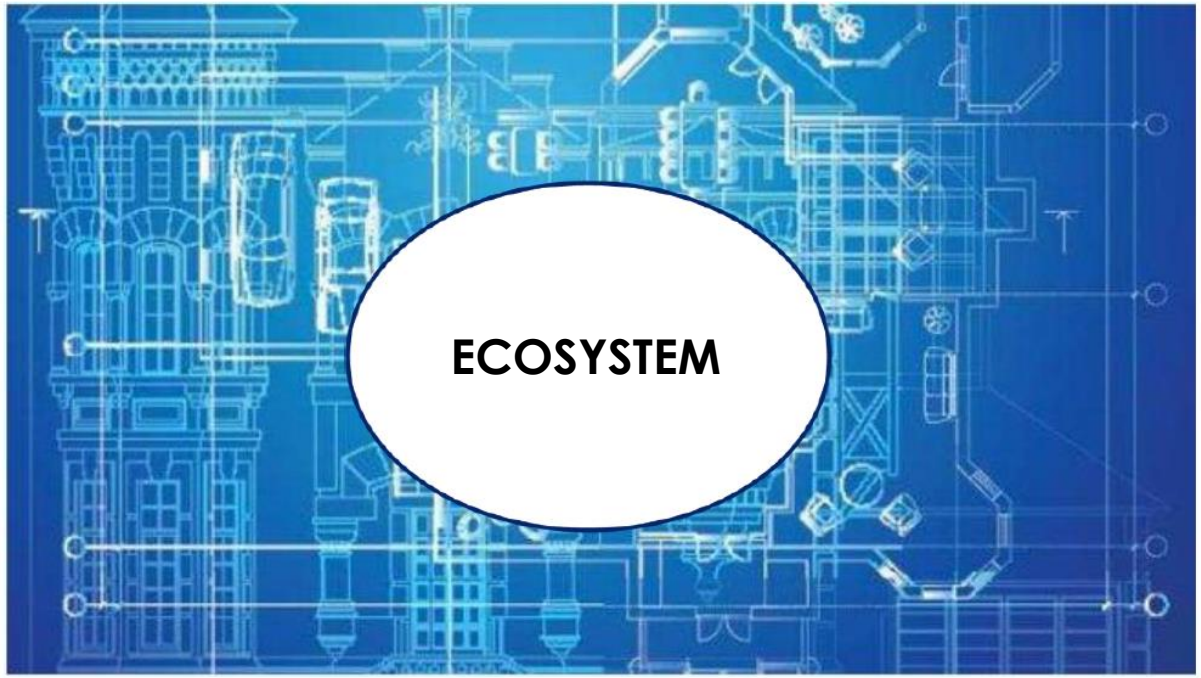
Bixby

How to View the Problem

User View

Data View

Operational View



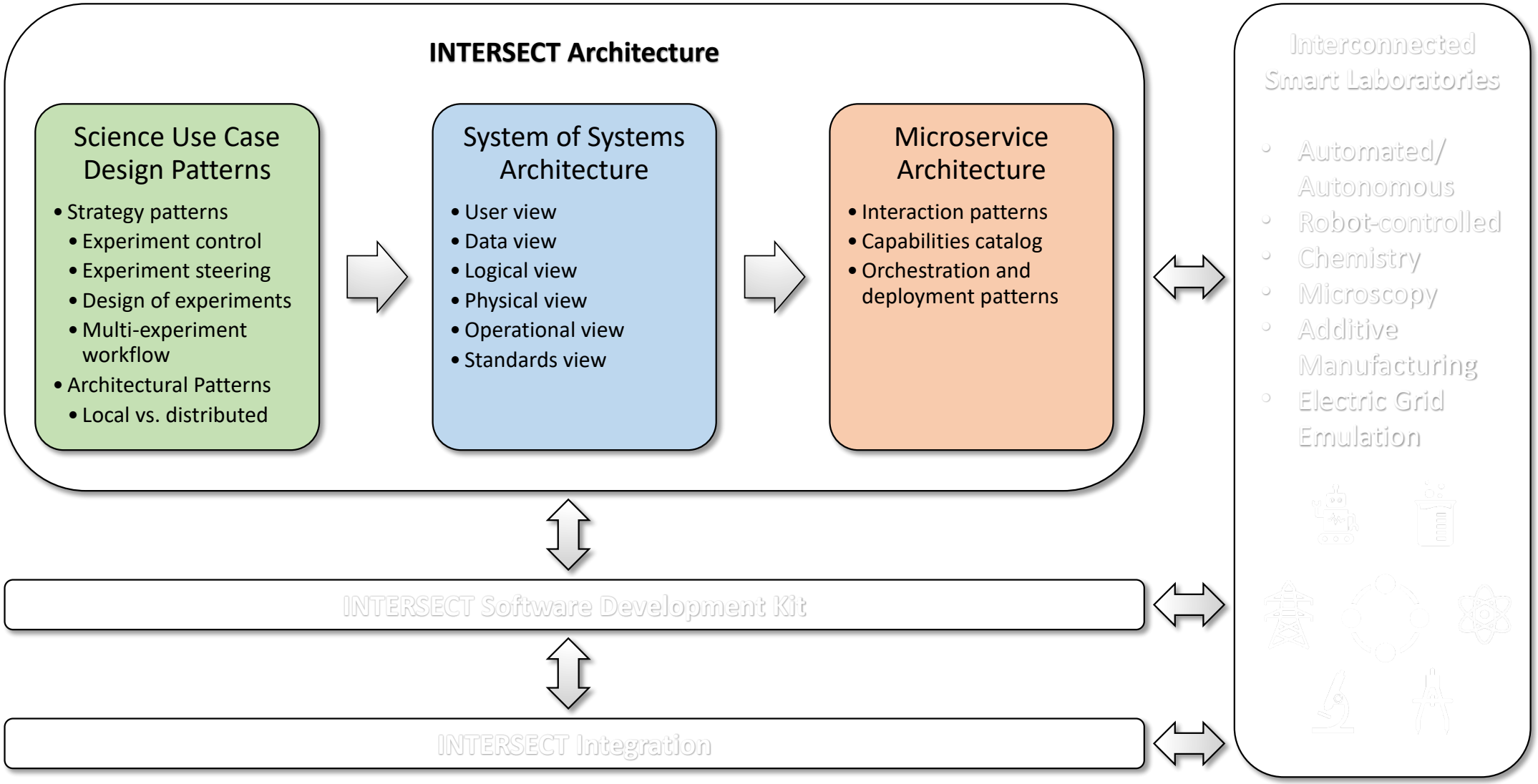
Logical View

Physical View

Standards View

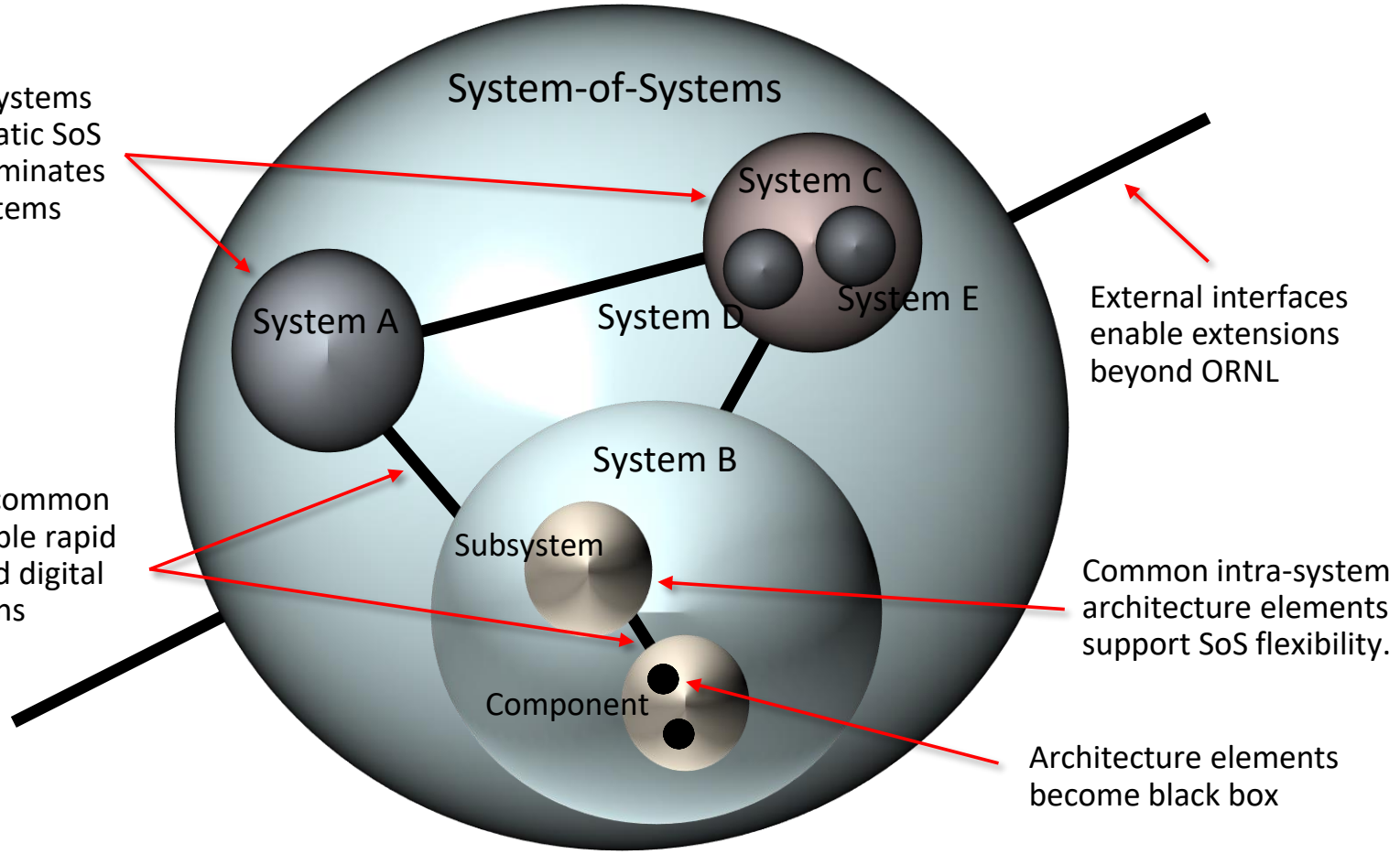
There are many stakeholders and we need to make sure everyone is satisfied

Build from the Ground Up with Uniform Standards



System of Systems Architecture

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

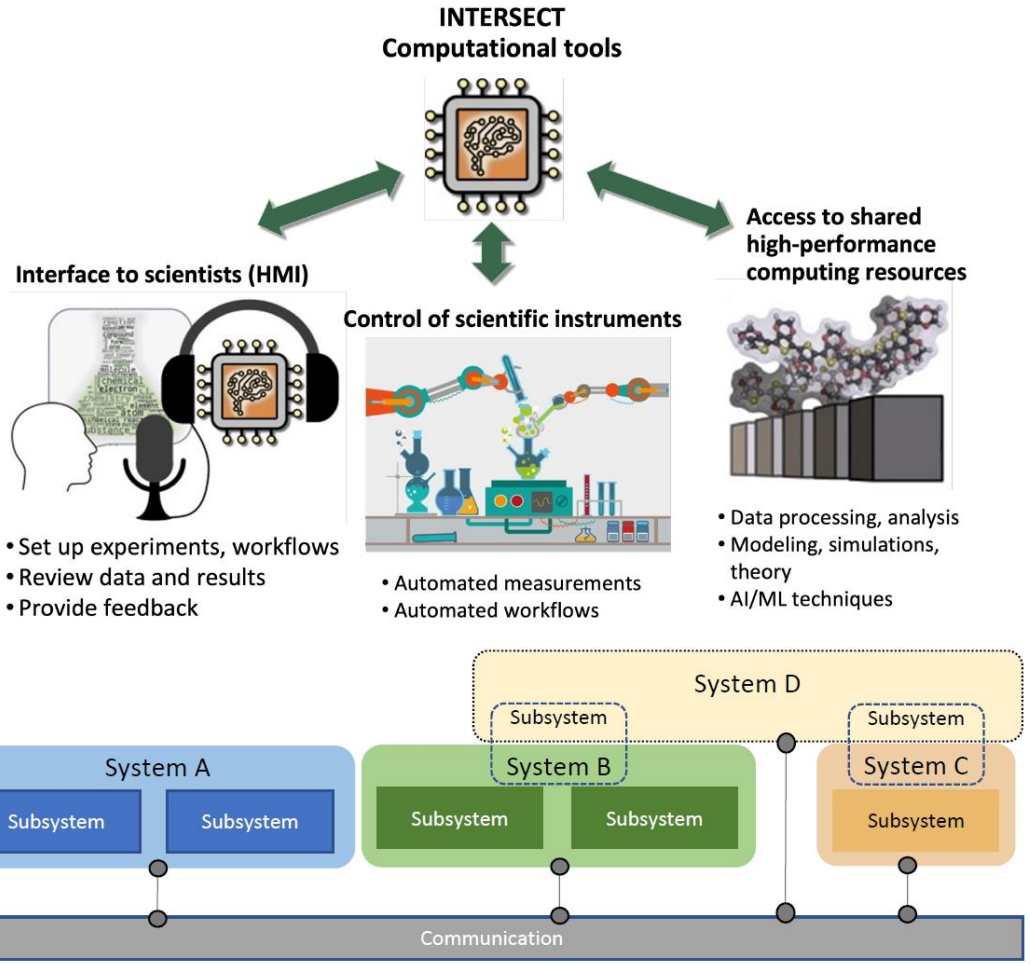
Component

ComponentControlStatus
ComponentCommand
ComponentCommandStatus

SoS Enables Scalable, Flexible, and Interoperable Autonomous Experiments

INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

Systems of Systems Architecture

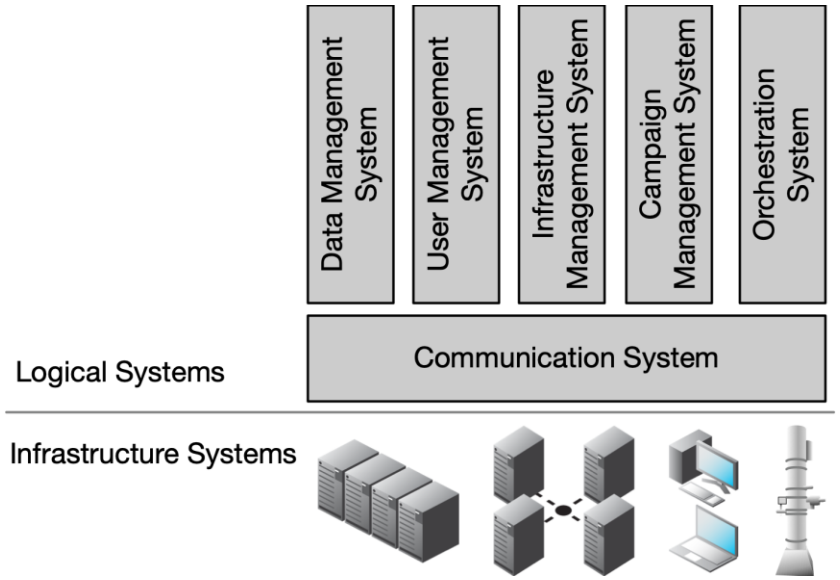


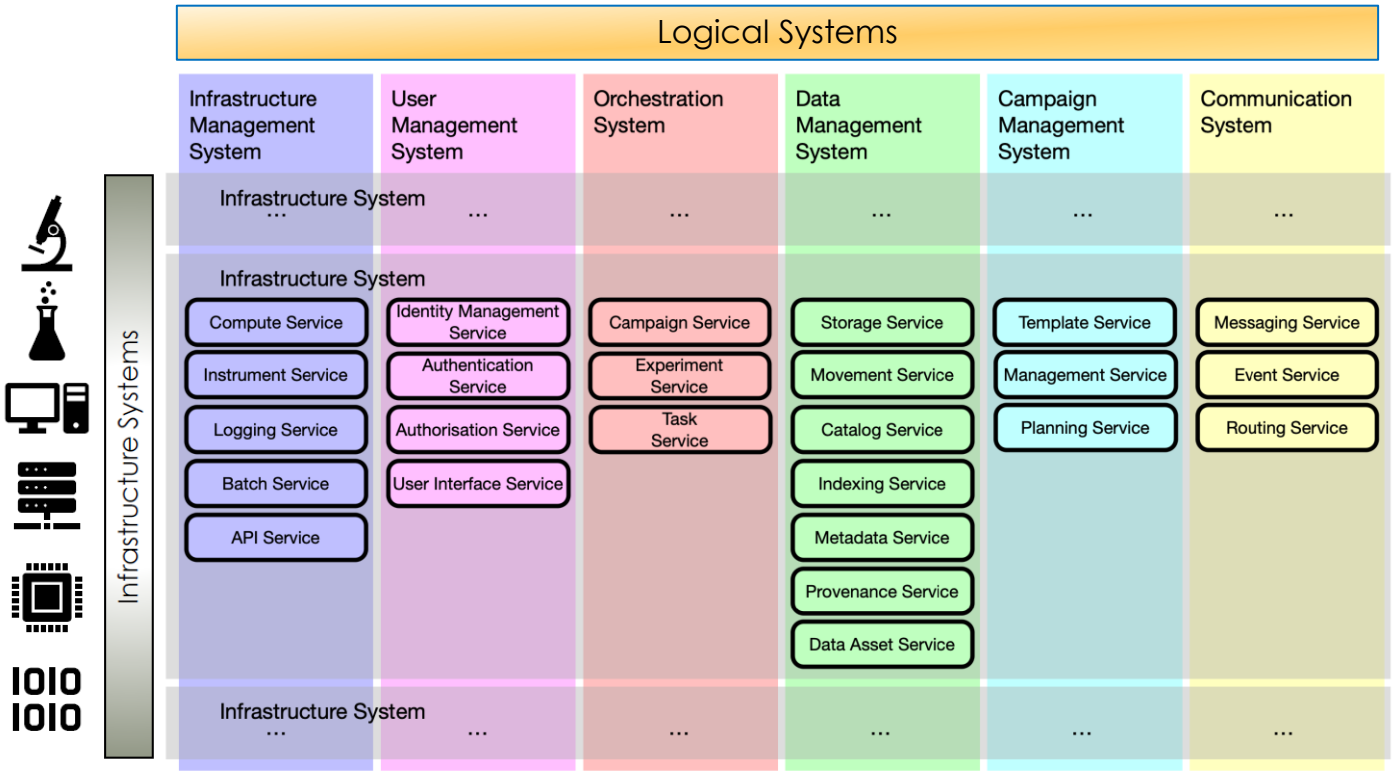
Stakeholder roles

	View Chapters					
Role	**User**	**Data**	**Operational**	**Logical**	**Physical**	**Standards**
Application software developers	X	X		X		X
Infrastructure software developers		X	X		X	X
End users	X			X		
Application and platform hardware engineers						
Security Engineers		X	X		X	
Communications engineers		X				X
System-of-system engineers		X	X	X	X	X
Chief engineer/scientists	X	X		X		X
Lead System Integrator		X	X		X	
System Integration and test engineers	X	X	X	X		
External test agencies	X	X	X	X	X	
Operational system managers	X	X	X			

Break the experimental tasks down into the smallest components. Treat the components as a system and figure out how to connect them in the simplest ways.

Connecting the Pieces





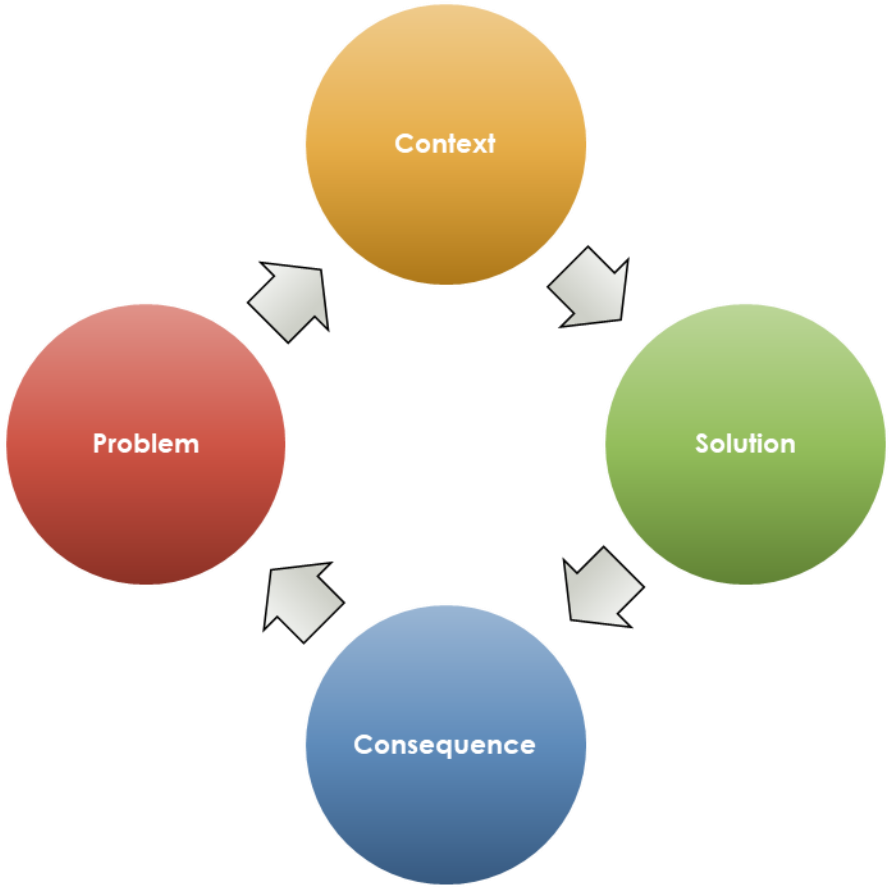
Simple minded systems:
- Infrastructure Systems
- Logical Systems

- Connections between Infrastructure and Logical systems is through communications
- Microservices provides the links which creates capabilities

With the systems and links, now we need to think about the patterns of connections

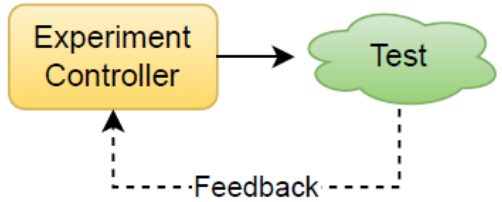
Design Patterns

- Systematize recurring problems by describing generalized solutions based on best practices
- Offer solution templates to solve specific problems that may apply to different situations
- Provide different solution alternatives to specific problems
- Identify the key aspects of solutions and create abstract descriptions to develop reusable design elements
- Communicate problems and solutions with clear terms and abstract concepts

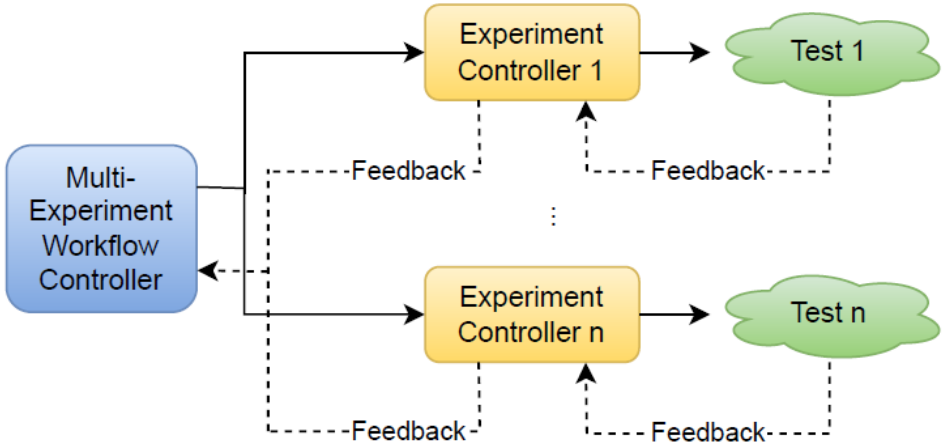


Design Pattern Anatomy

- ***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



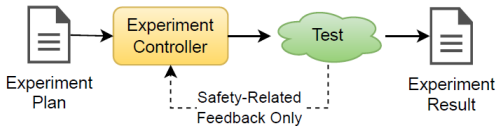
Single experiment control



Multi-experiment control

Use Case Design Pattern Strategy

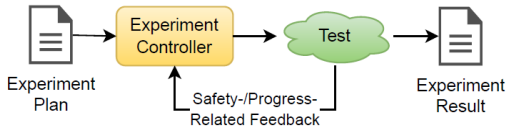
Experiment Control



Executes an existing plan

- Open loop control
- Automated operation

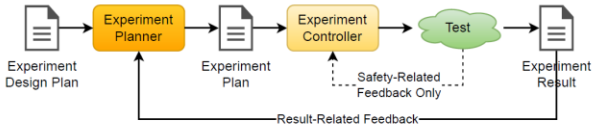
Experiment Steering



Executes an existing plan, depending on progress

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

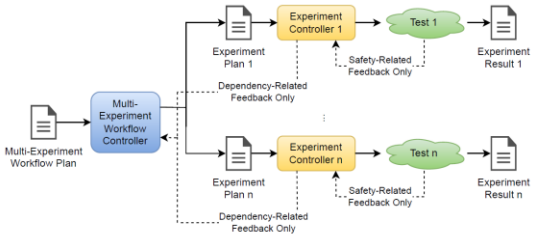
Design of Experiments



Creates / executes a plan, based on prior result

- Closed loop control
- Autonomous operation
- Use patterns:
  - Experimental Control
- May use patterns:
  - Experimental Steering

Multi-Experiment Workflow



Executes existing plans (workflows of experiments)

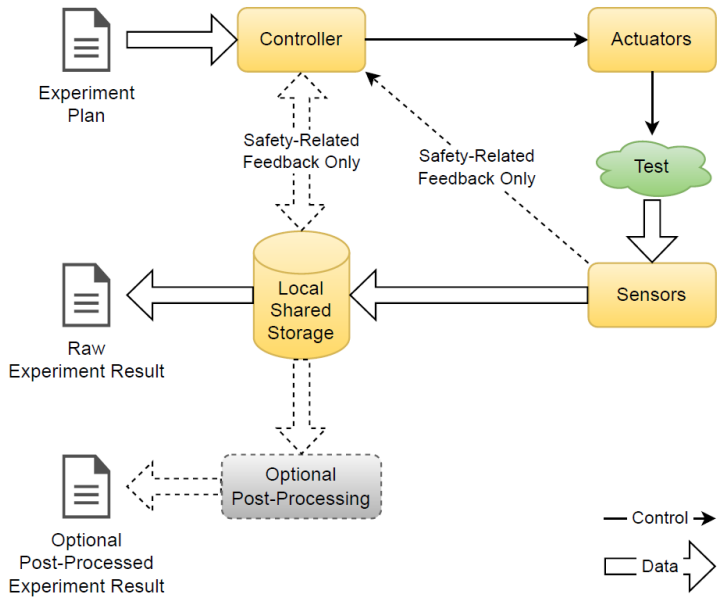
- Open loop control
- Automated operation
- User patterns:
  - Experimental Control
- May use patterns:
  - Experimental Steering
  - Design of Experiments

Nesting patterns helps ensure universality and interoperability (think Lego blocks)

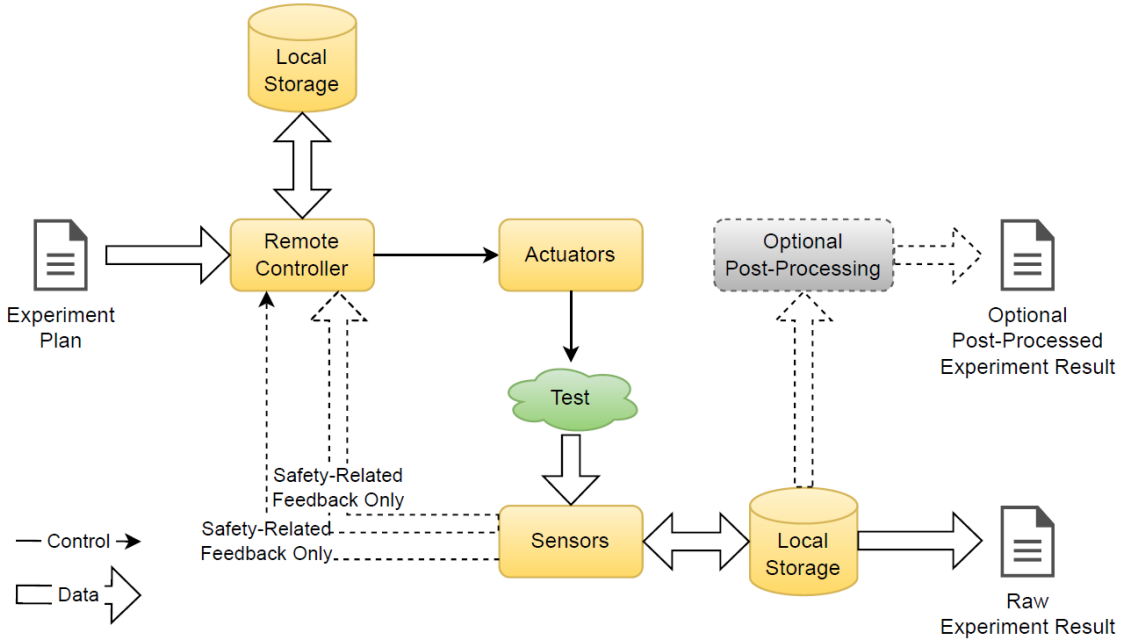
OAK RIDGE National Laboratory

INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

Mapping Use Case Patterns on to Architectural Patterns



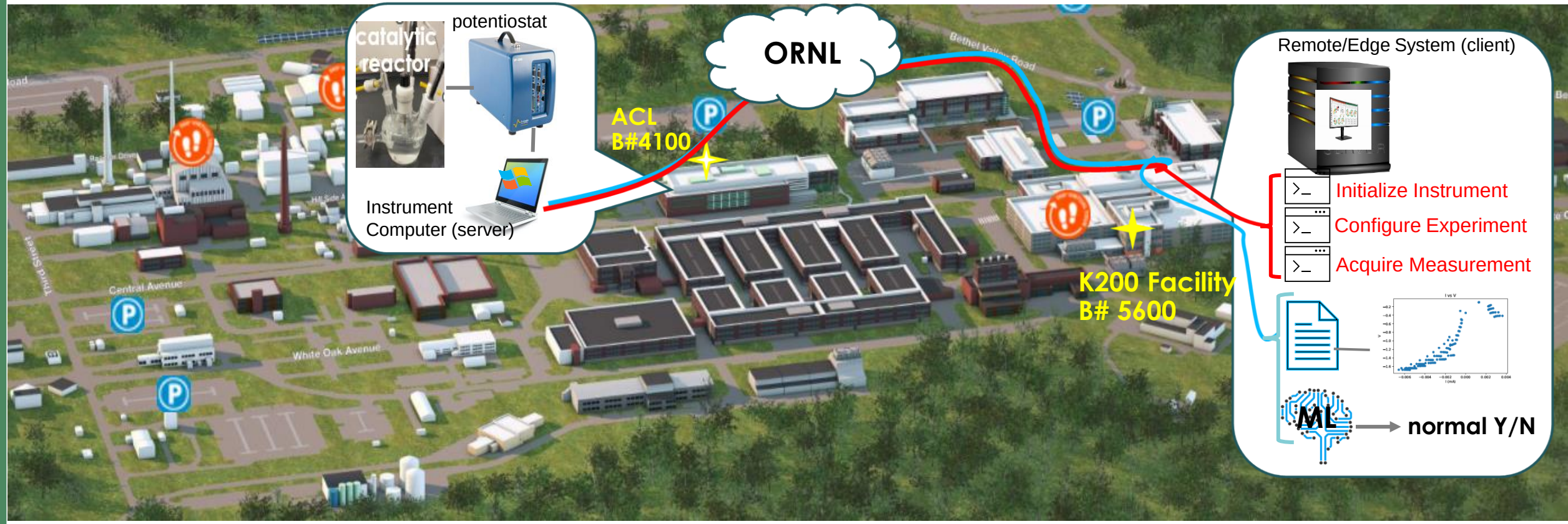
Local experiment control



Distributed experiment control

We want to make sure that our ecosystem doesn't care whether the systems and architectural components are next to each other or across the world.

Federated Approach Across ORNL



Data Channel of OneDrive end-to-end setup

Control Channel: remote steering Potentiostat via Pyro API

Potentiostat and control computer communication via USB

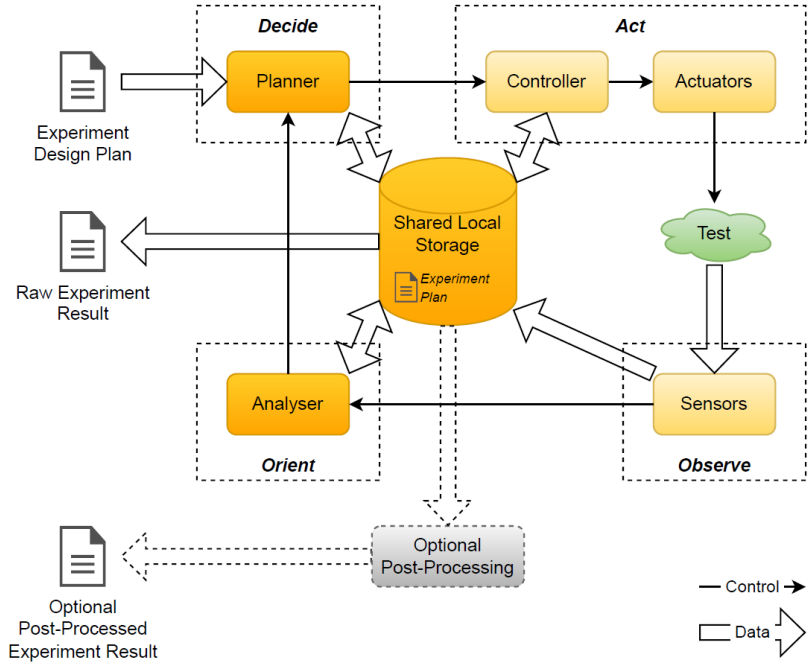
ML Method: detect deviation from normal

- based on I-V regressions

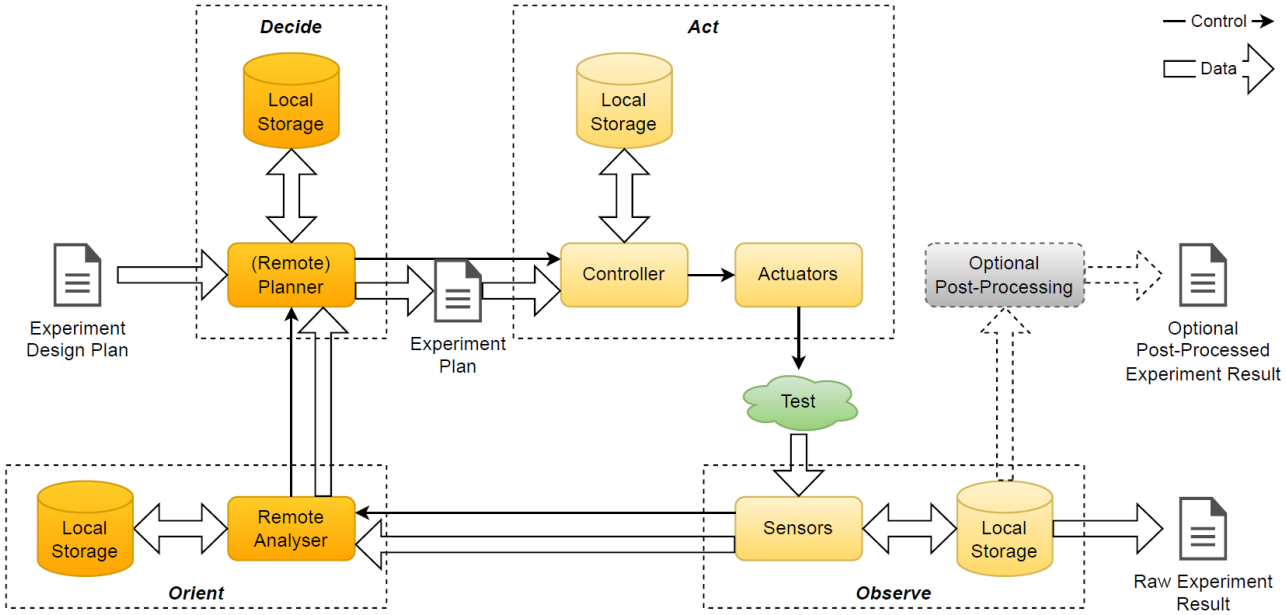
OAK RIDGE National Laboratory

INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

Mapping Use Case Patterns on to Architectural Patterns



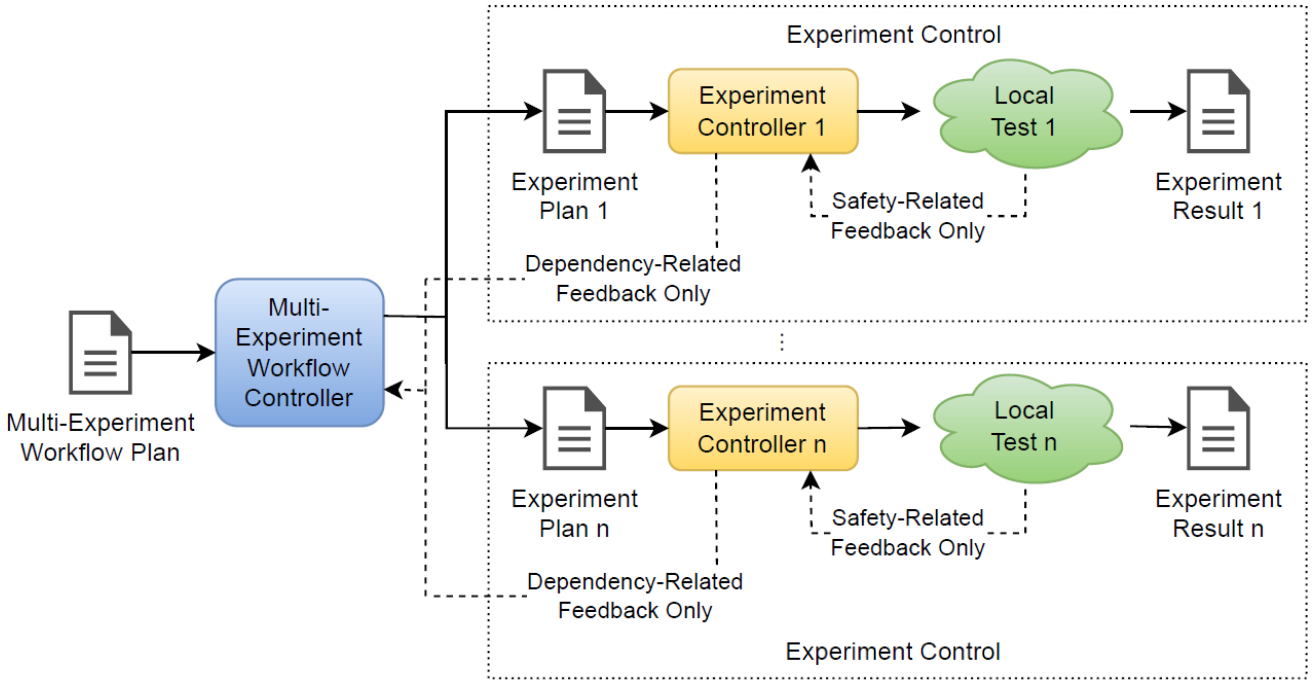
Local design of experiments

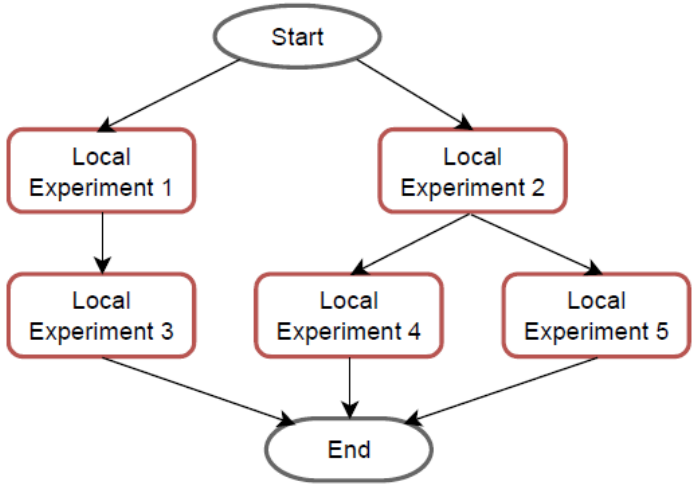


Distributed design of experiments

As with our nested strategic design patterns, we can map different strategies onto architectural patterns to understand how to move data and control experiments.

Multi-Experiment Workflows

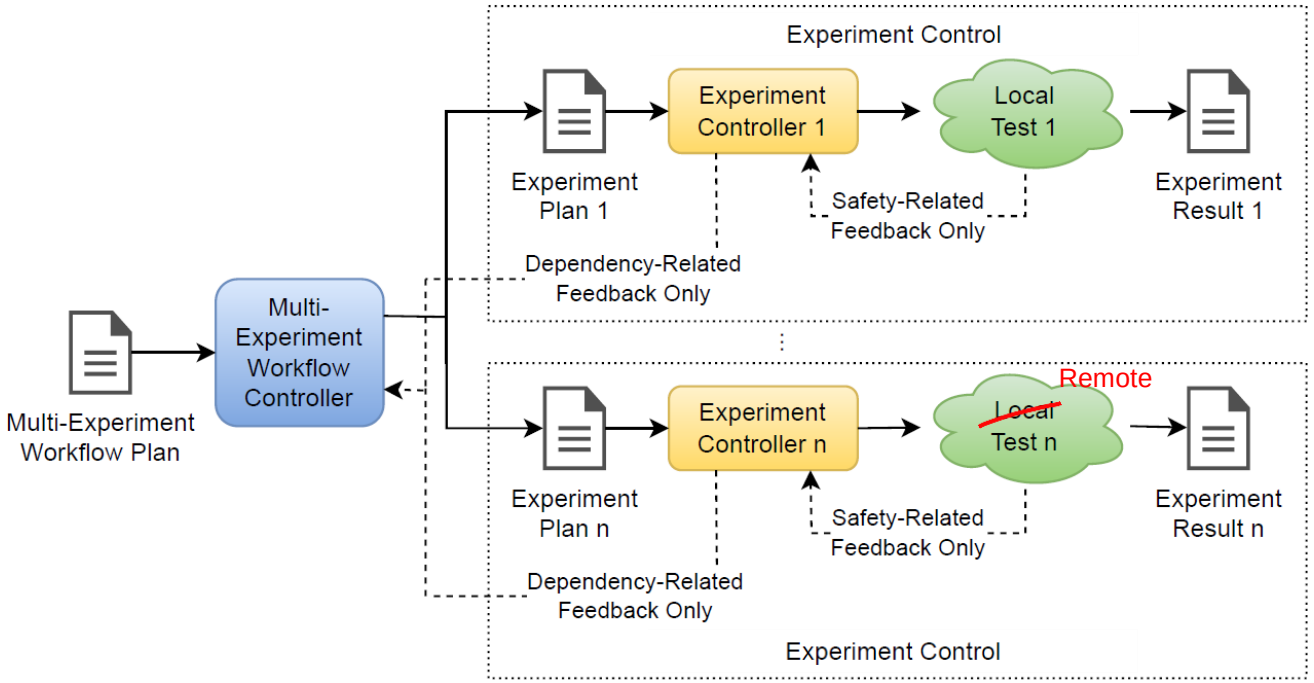


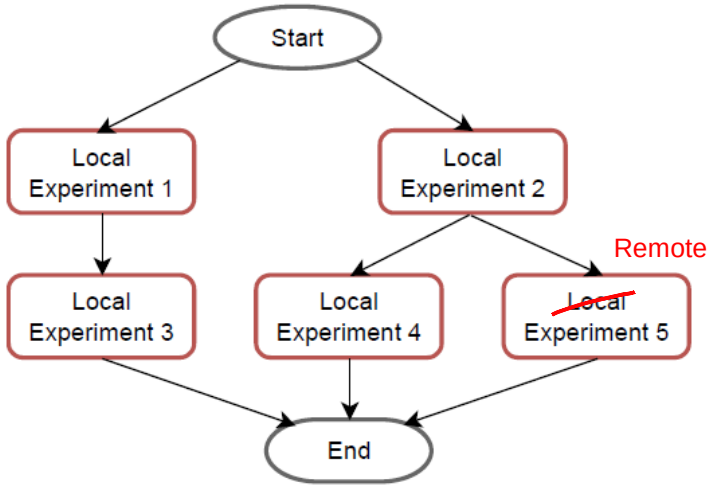


Local multi-experiment workflow

As we reach the multi-experiment workflow design patterns, you can start to see the benefits of this approach.

Multi-Experiment Workflows



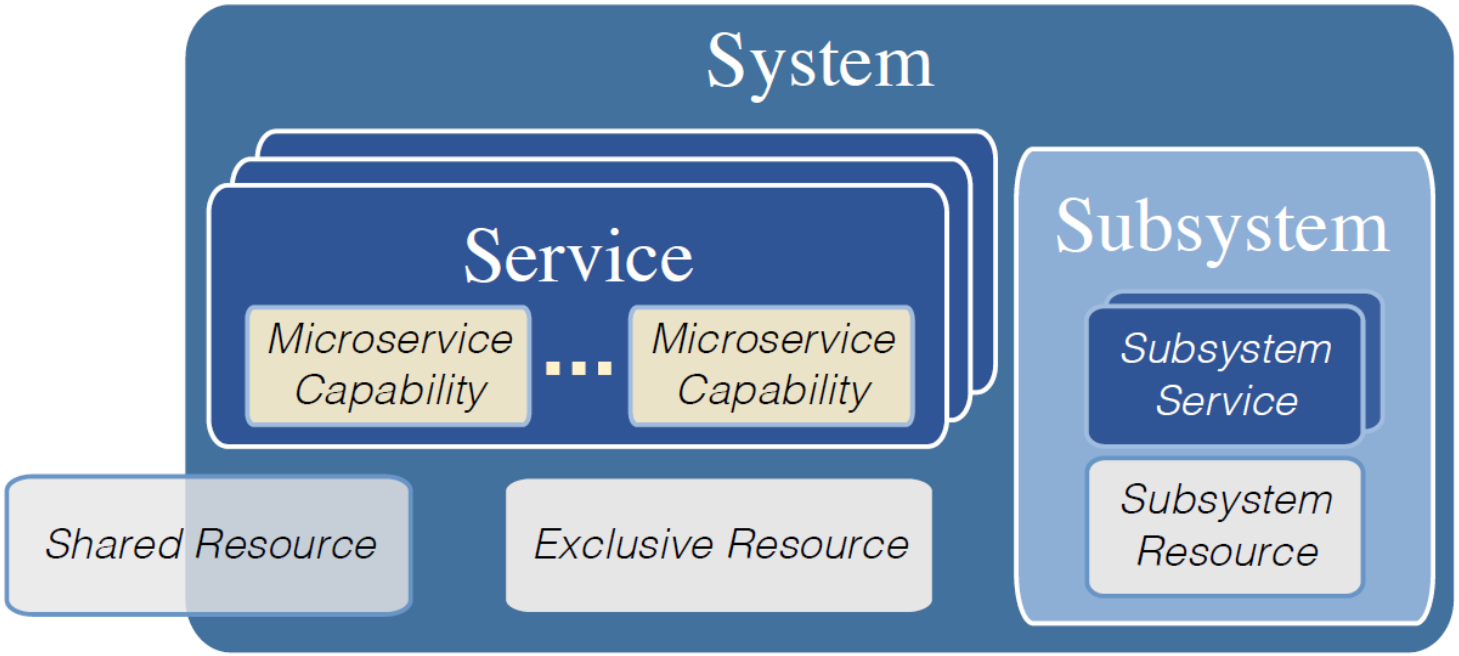


Distributed

~~Local~~ multi-experiment workflow

Simplifying the connections between abstract architectural components allows for interoperability and autonomous workflows regardless of science domain or locality of resources.

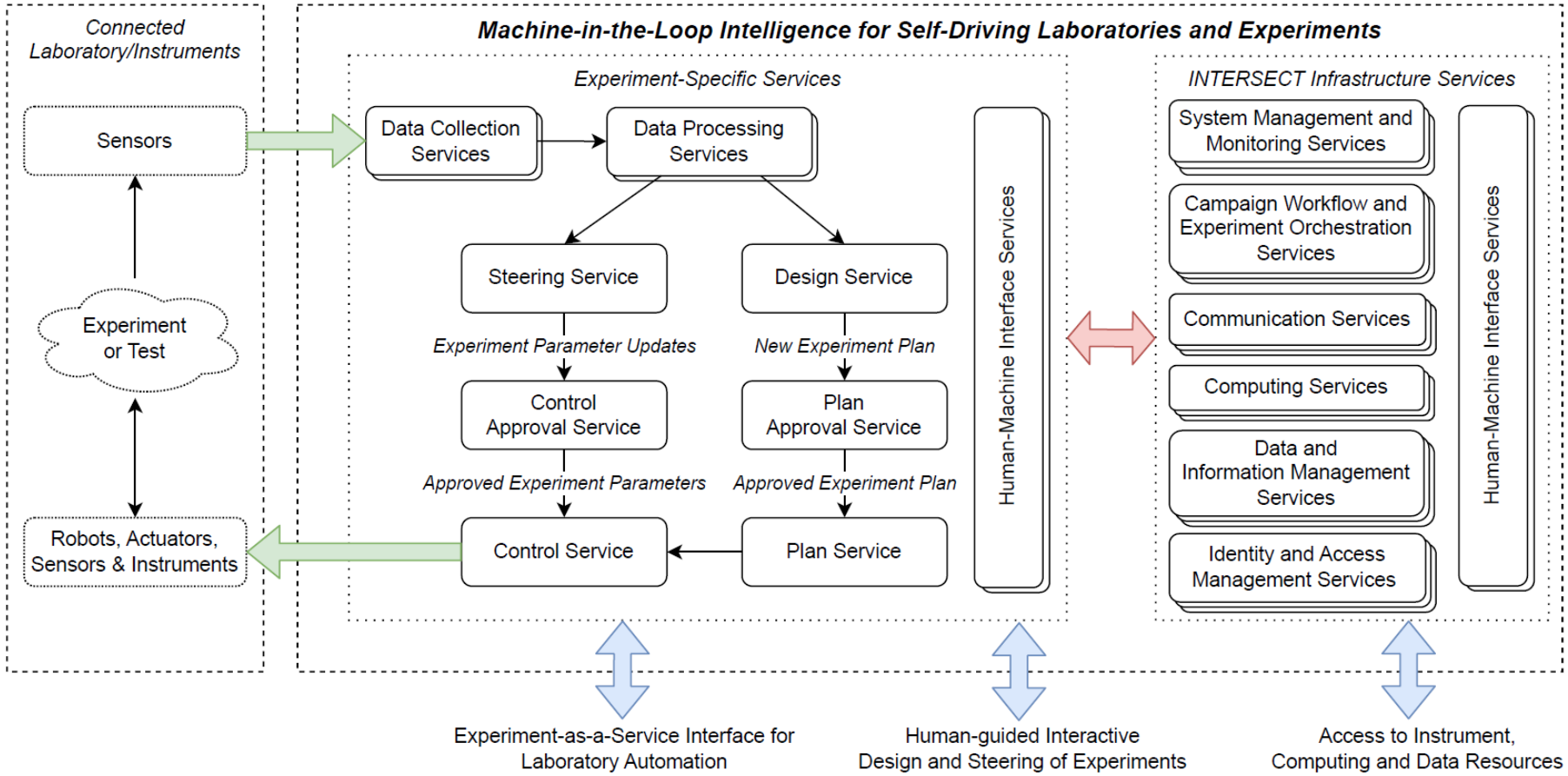
Microservices – From Components to Capabilities



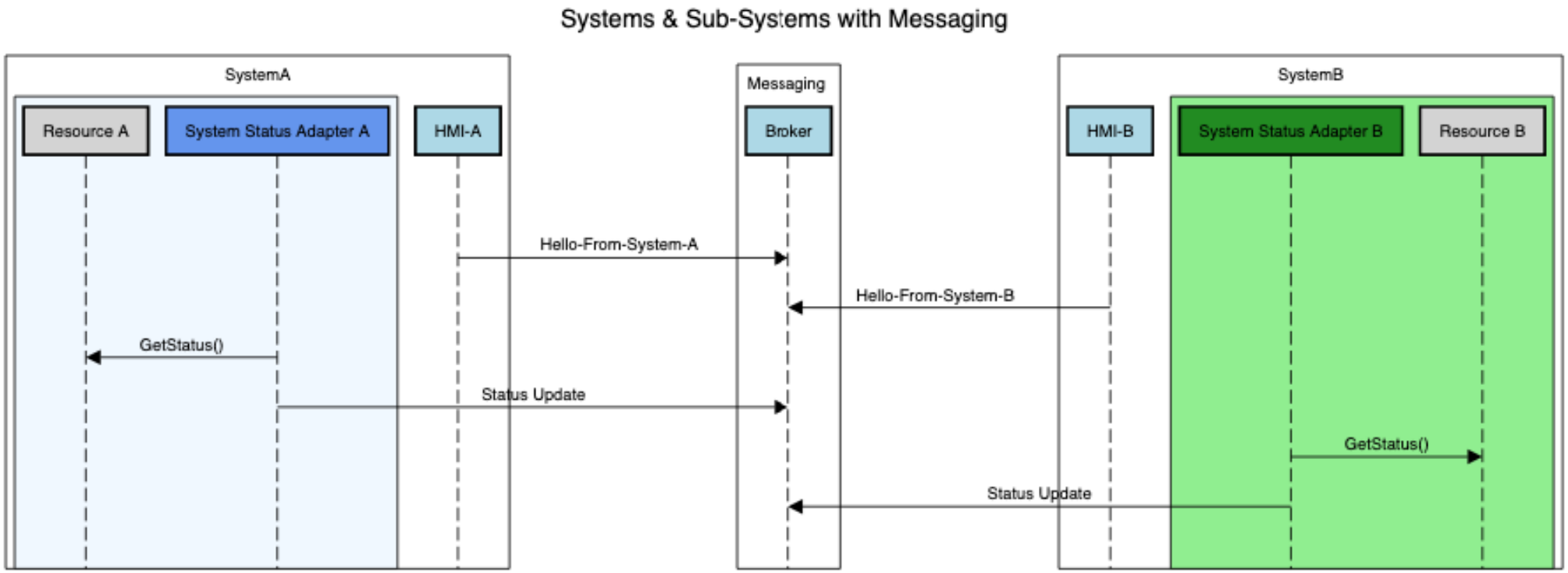
We have seen a systems of systems approach and how abstract patterns map architectural components onto workflows. So how do we turn architectural components into experimental capabilities?

Microservices – From Components to Capabilities

Classes of Microservices

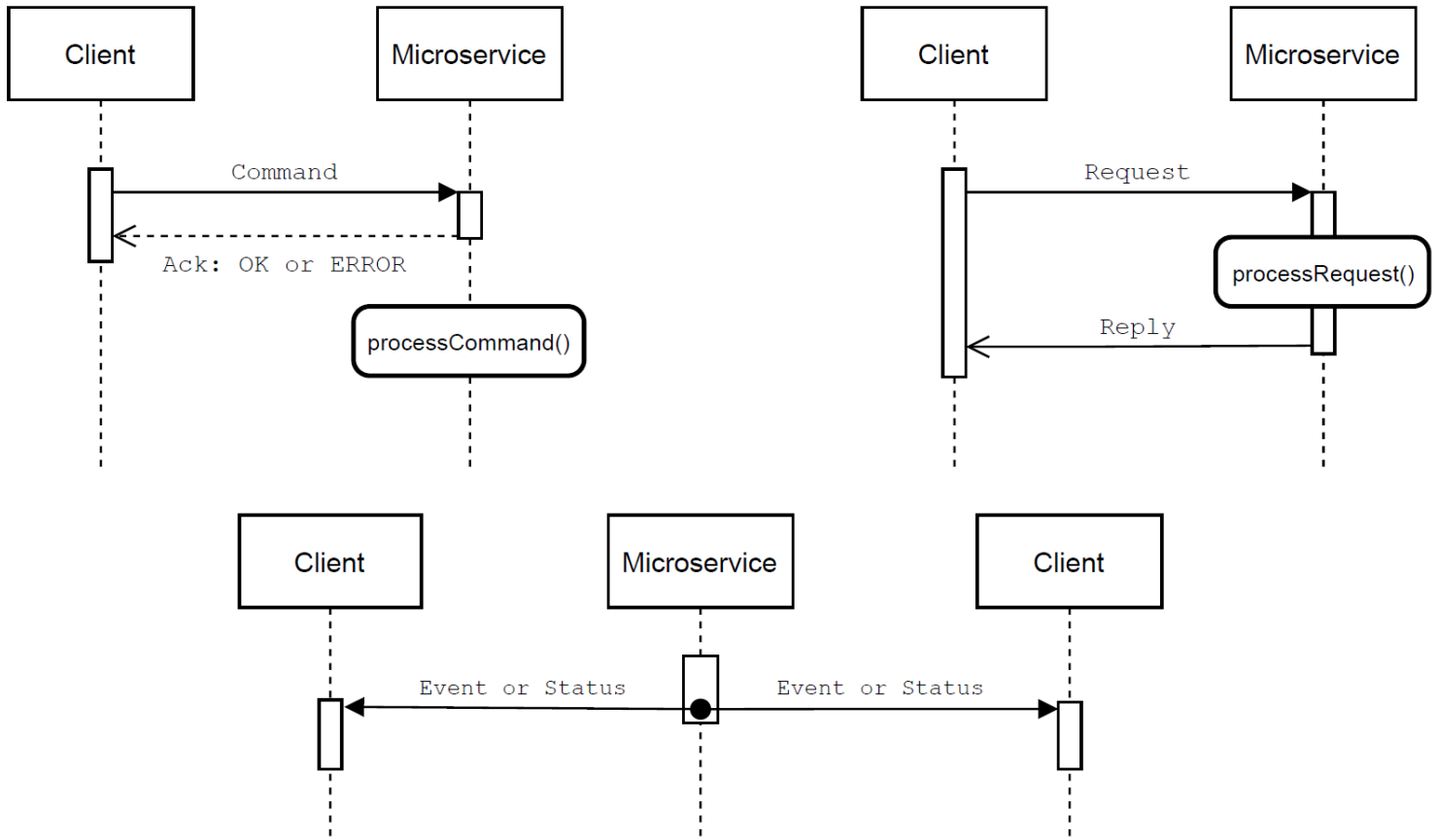


Communications – Connecting the Pieces



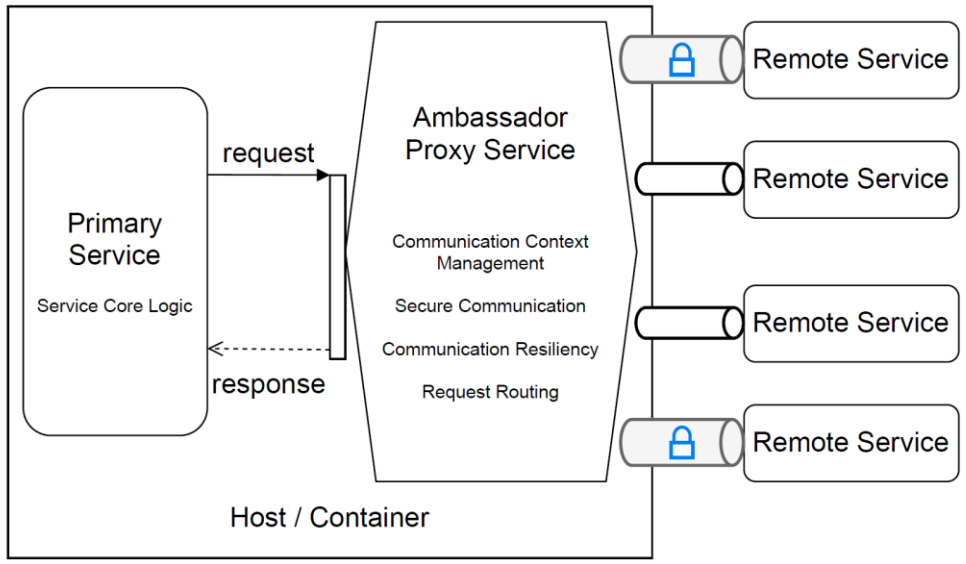
Using the systems of systems approach, we have broken down our experimental workflow into architectural components and microservices turns these components into capabilities. So how does this work in physical systems, any physical system…… through simple messages

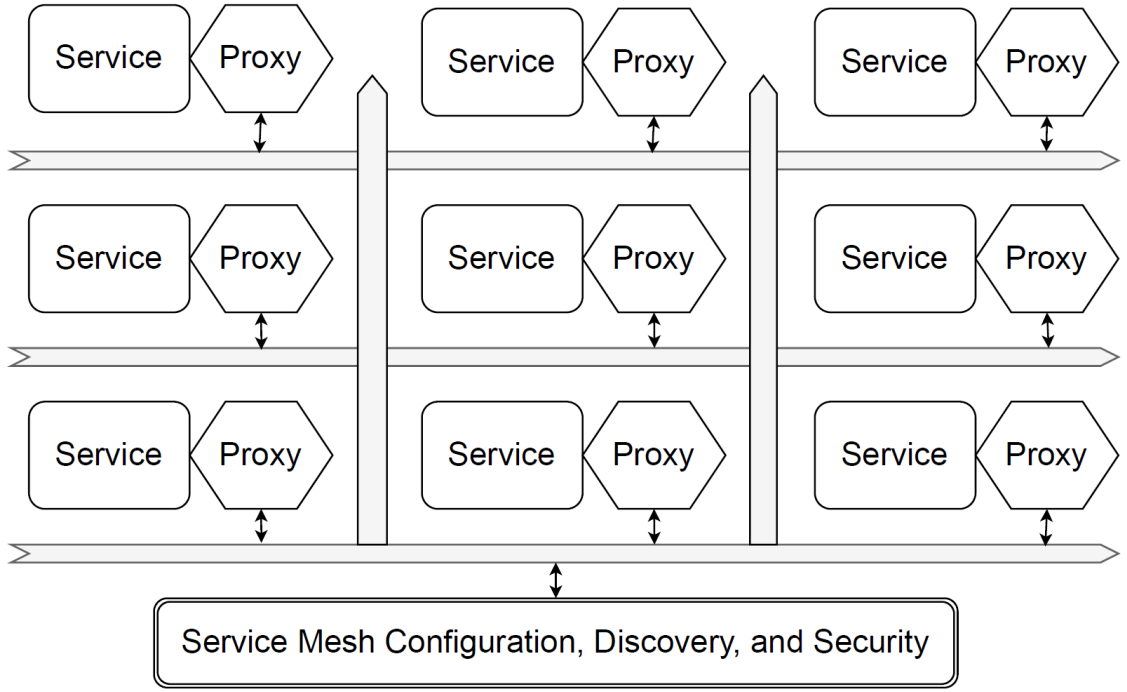
Communications – Connecting the Pieces



Through simple, abstract messages we can coordinate tasks and actions between architectural components using the microservices. By design, this abstract messaging layer doesn't care what resources we are coupling or where they are located.

Communications – Connecting the Pieces

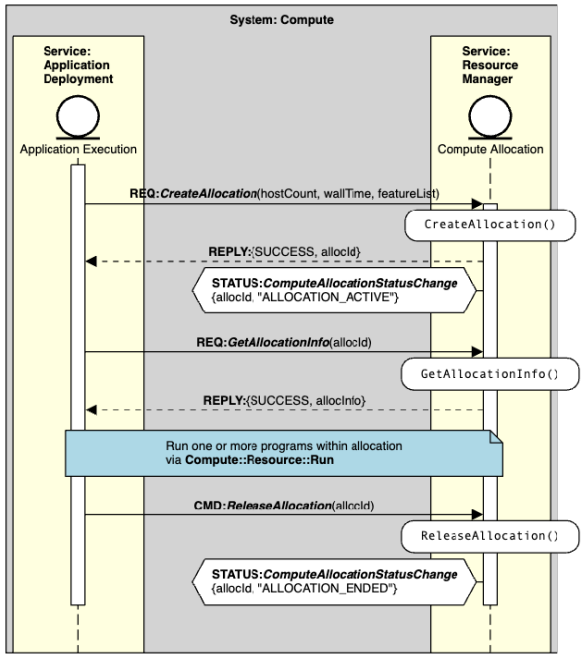




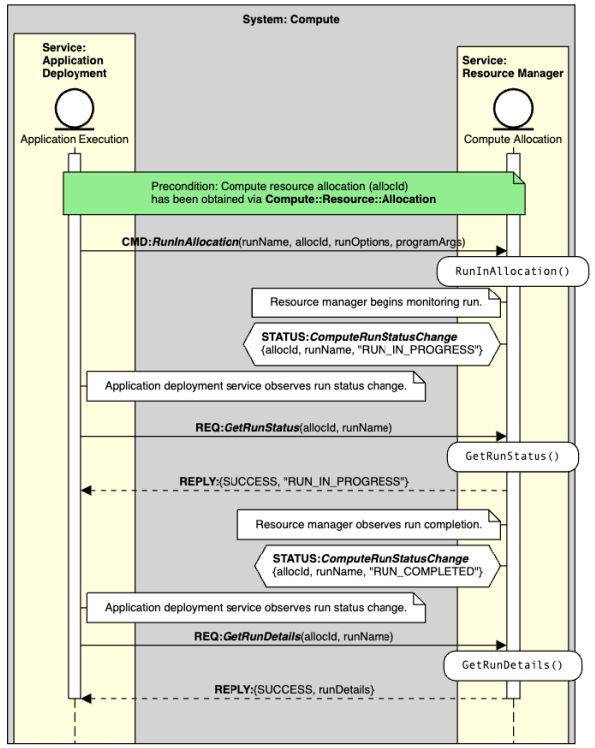
The simplicity of the messages alleviates restrictions on how they are deployed

Messaging Example: Compute Resources

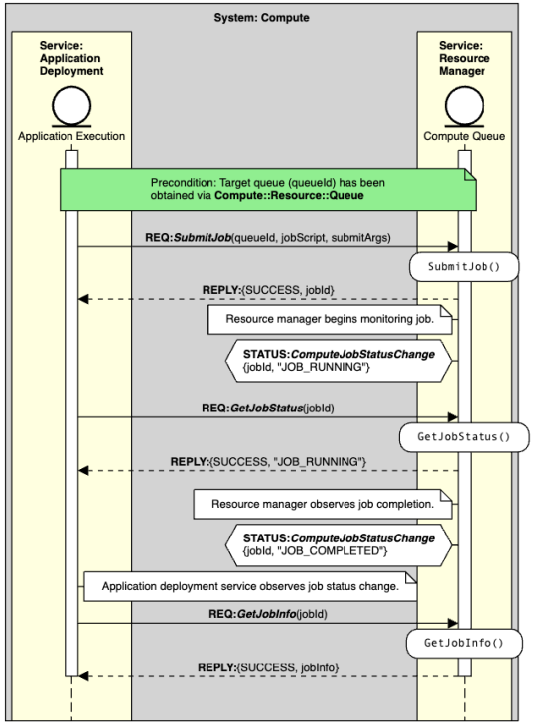
Compute::Resource::Allocation - Allocation of Compute Resources



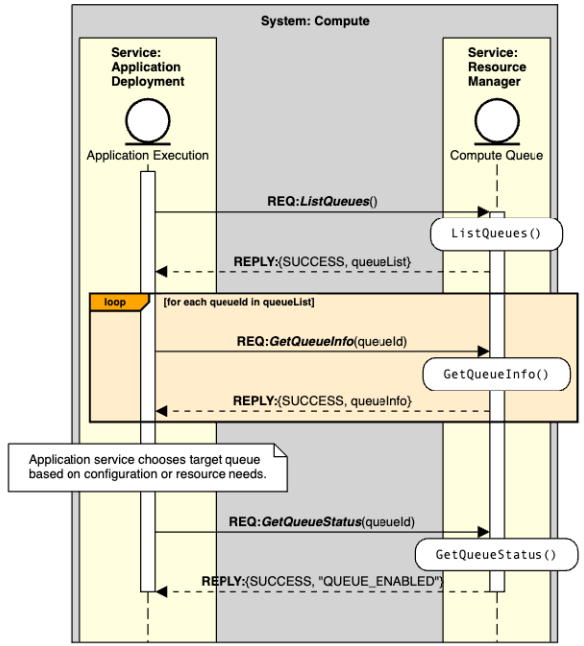
Compute::Resource::Run - Run a Program within a Compute Allocation



Compute::Resource::Job - Submit a Job to a Compute Queue



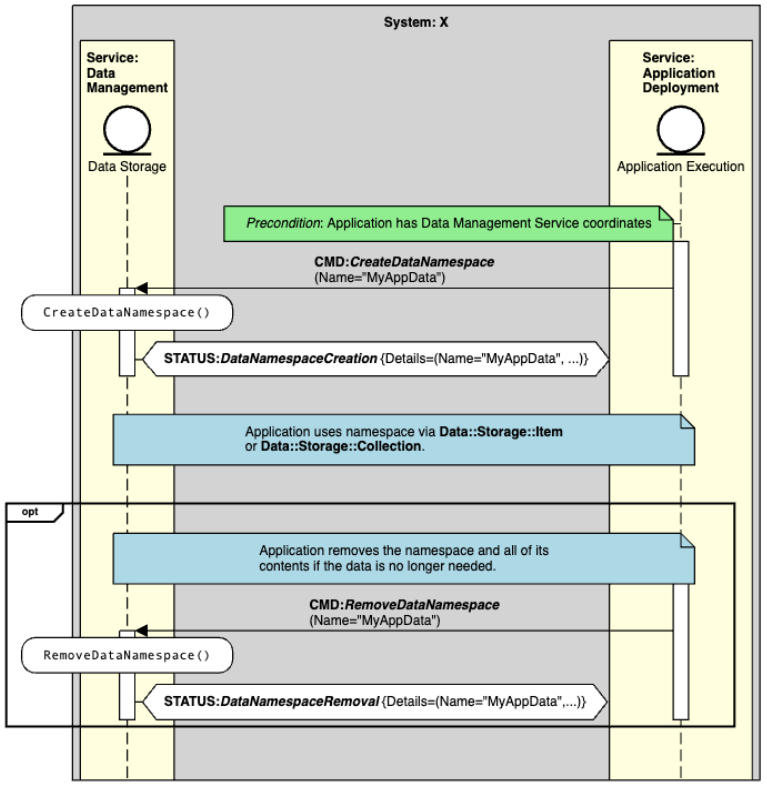
Compute::Resource::Queue - Discovery of Batch Computing Queues



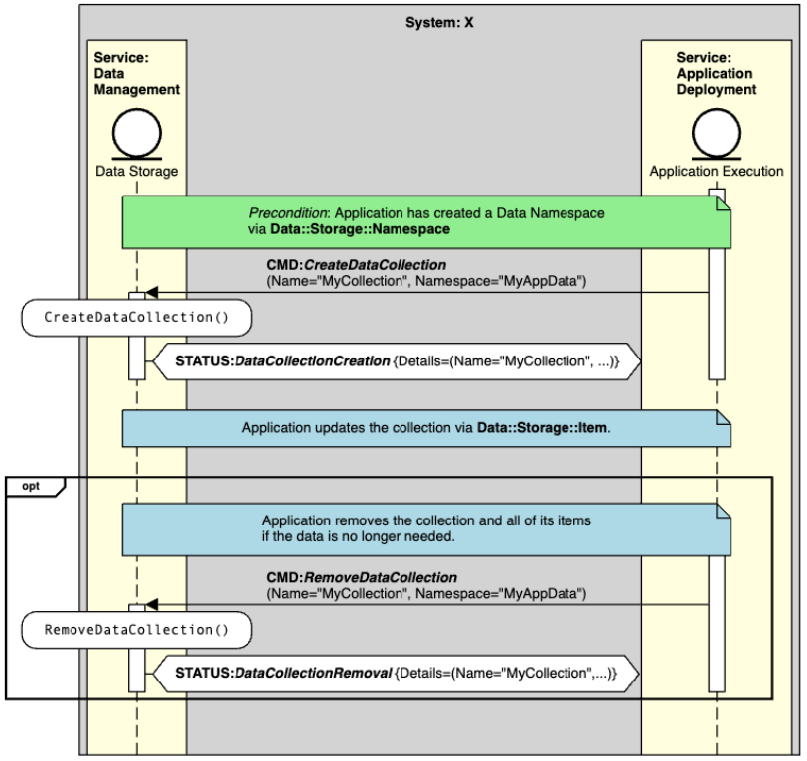
Searching for and allocating compute resources, running applications within the allocation, discovering batch computing queues, and submitting jobs to a queue is now handled with simple, low-level messages.

Messaging Example: Handling Data

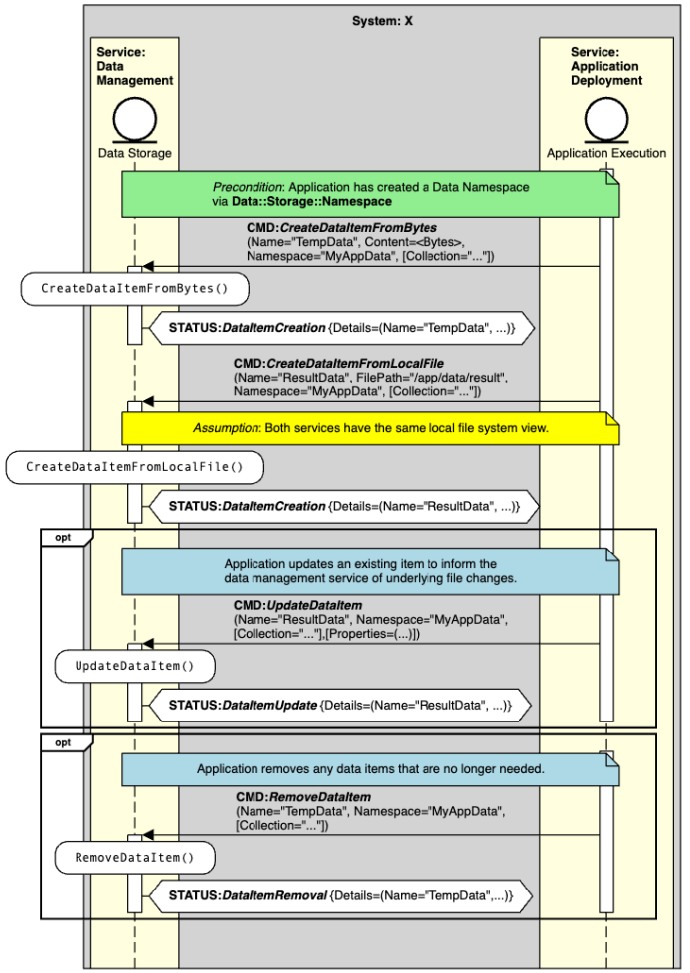
Data::Storage::Namespace - Create or Remove a Data Namespace



Data::Storage::Collection - Create or Remove a Data Collection



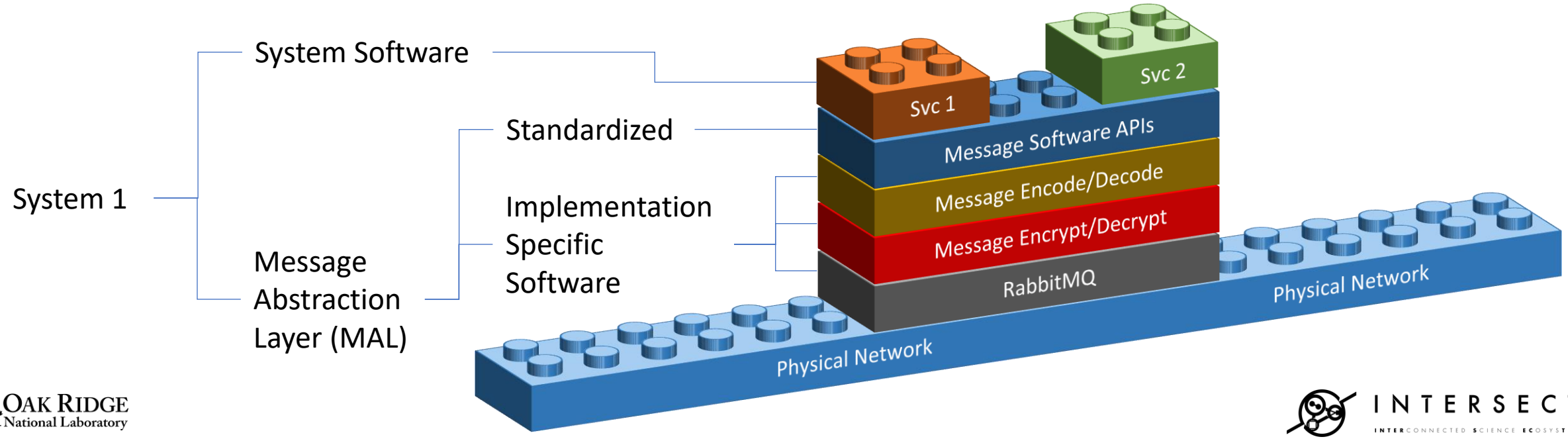
Data::Storage::Item - Management of Data Items



Regardless of the resources utilized or microservices triggered, we can coordinate complex tasks using a simple message service

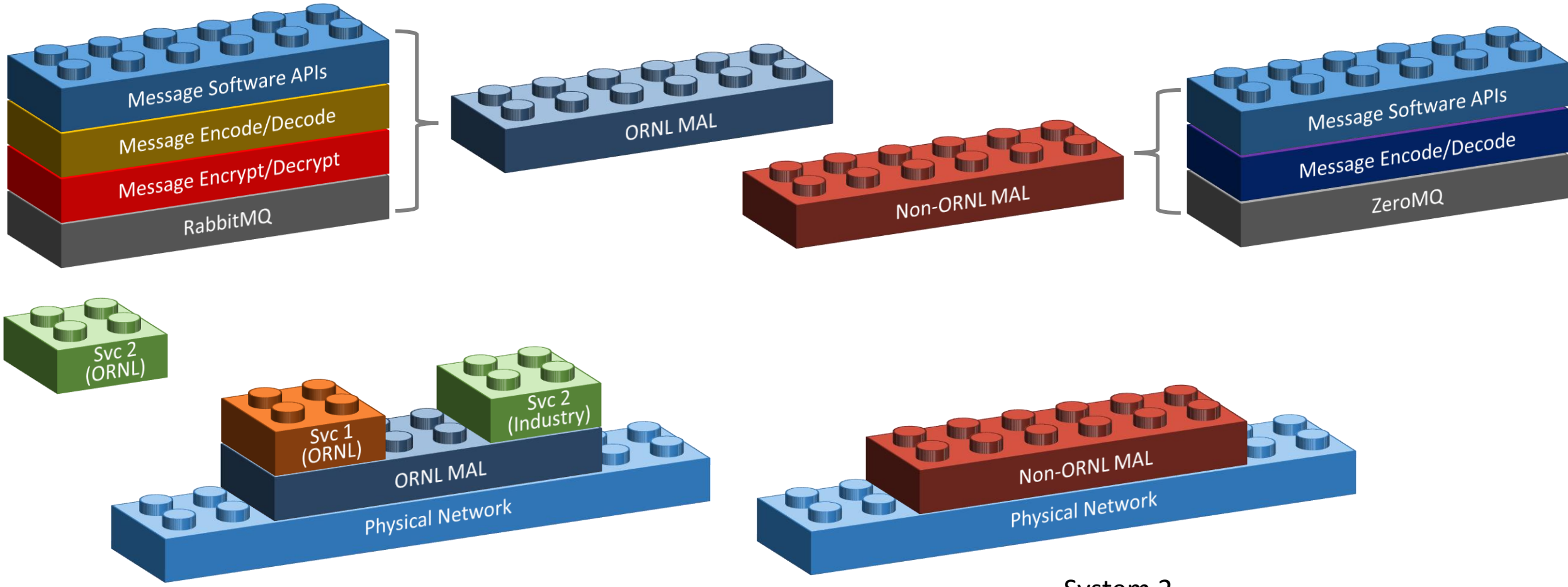
Single System Software Detailed

- MAL is the combination of a standardized set of Message Software APIs and Implementation Specific Software to send/receive messages
- Message Software APIs provide common functions to send, receive, and manipulate message objects
  - create message_object instance or receive a message object
  - message_object.get_header().set_SystemId(151001132117188)
  - send (message_object)
- Implementation Specific Software is required to package up a message and send it over the physical network



Multiple Systems (Not Connected)

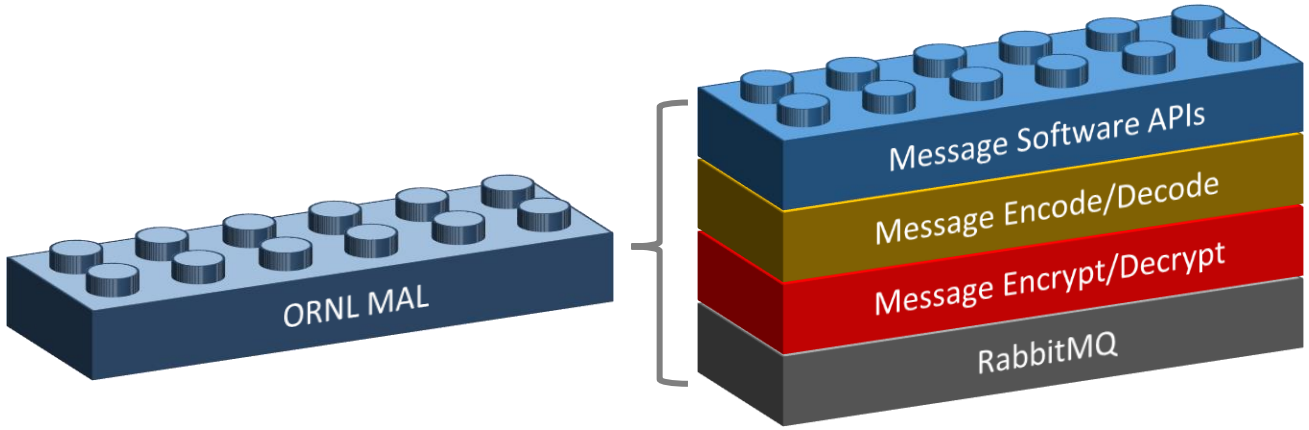
- MAL enables flexible software reuse through the common/open Message Software APIs
- MAL increases the number of teams that can develop software capabilities (e.g. DOE, Industry, and Academia)
- MAL enables plug-and-play integration testing to find the best-of-breed software



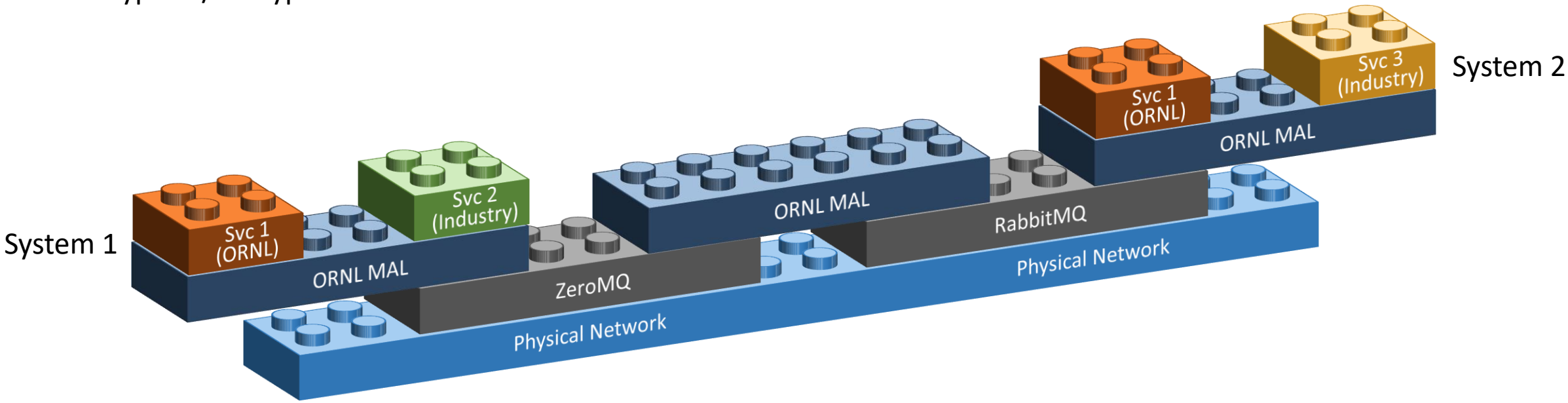
System 1

System 2

Connected SoS
(Single MAL / Multiple Brokers)

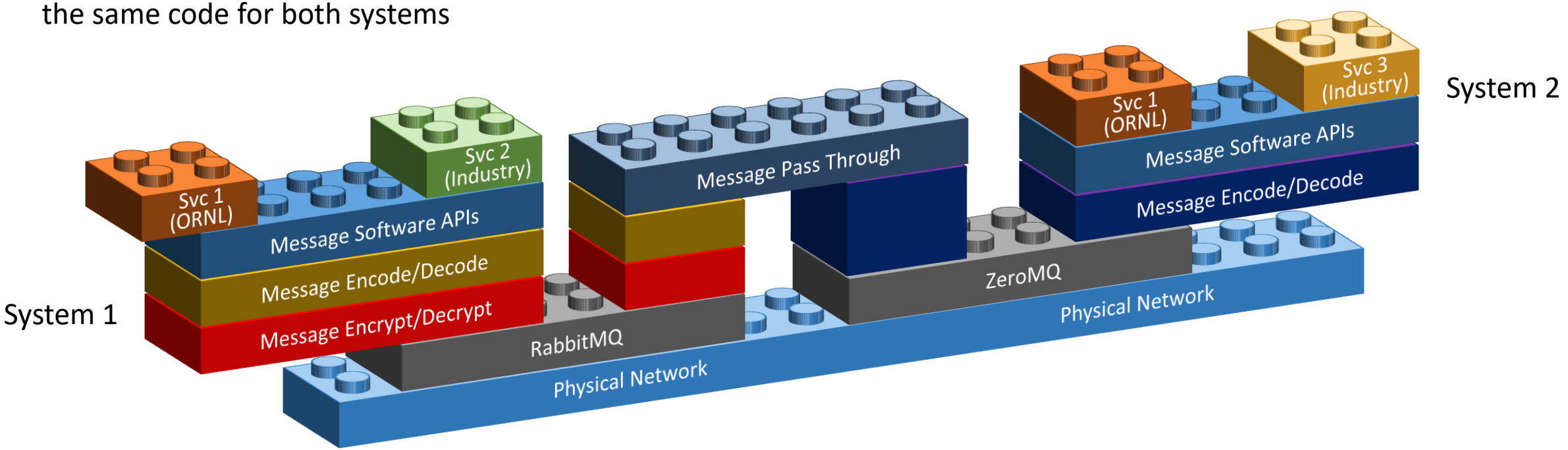


- A simple connection can be made across multiple brokers if the MAL is the same.
- Specifically, the encode/decode and encryption/decryption must be the same



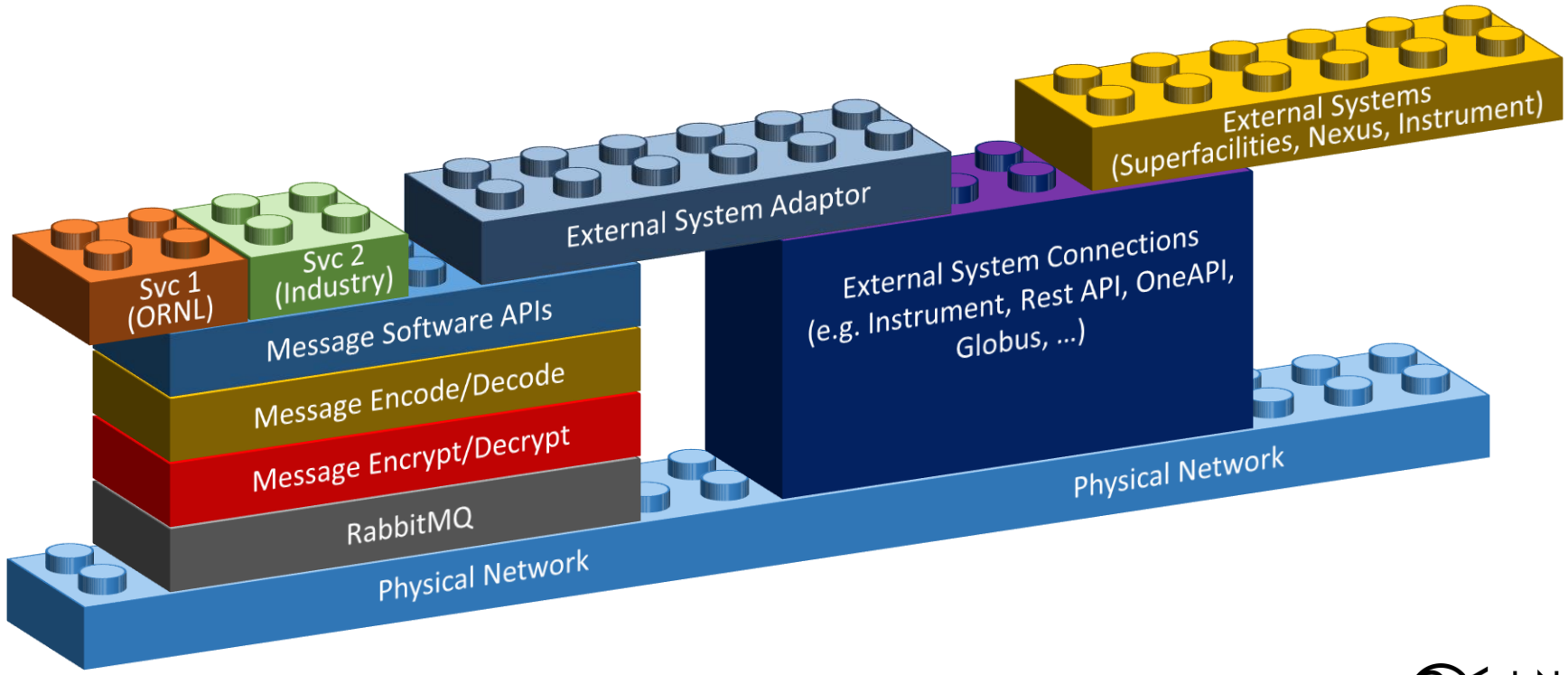
Connected SoS
(Multiple MALs / Multiple Brokers)

- A pass through is needed to connect multiple MALs if they have different Implementation Specific Software
- Since the Message Software APIs are standardized, software service 1 is the same code for both systems



Connected SoS
(External Non-Compliant Systems)

- External systems (i.e. not INTERSECT compliant) can connect through a message translation and/or message pass through
- Standardized messages would only need a pass through
- INTERSECT software services remain unchanged. The message translation is isolated to the edge of the INTERSECT boundary



So how does this work in real situations – Use Cases

Autoflows

Autonomous Continuous Flow Reactor Synthesis

- Periodic system status information
- Experimental status information triggered throughout the experiment

Step 5: System 4 completes the experiment; feedback is autonomous

User views status of the various systems

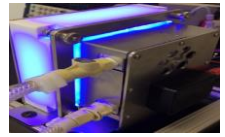
System 1
(includes HMI)

I-API

System 2

Continuous flow reactor

INTERSECT to EPICS translation



System 4
(Running EPICS)

Step 1: User creates the experimental plan using system 1
a) runs system 2
b) sends data to system 3 for analysis
c) starts system 4 with input from edge analysis

Step 2: User begins the experiment plan; setting which modes of in-line characterization will be used

Step 3: System 2 runs the first part of the experiment and data acquisition

Step 3.5: Data streams queued from System 2 are transferred to System 3

System 3

Edge Analysis Node(s)

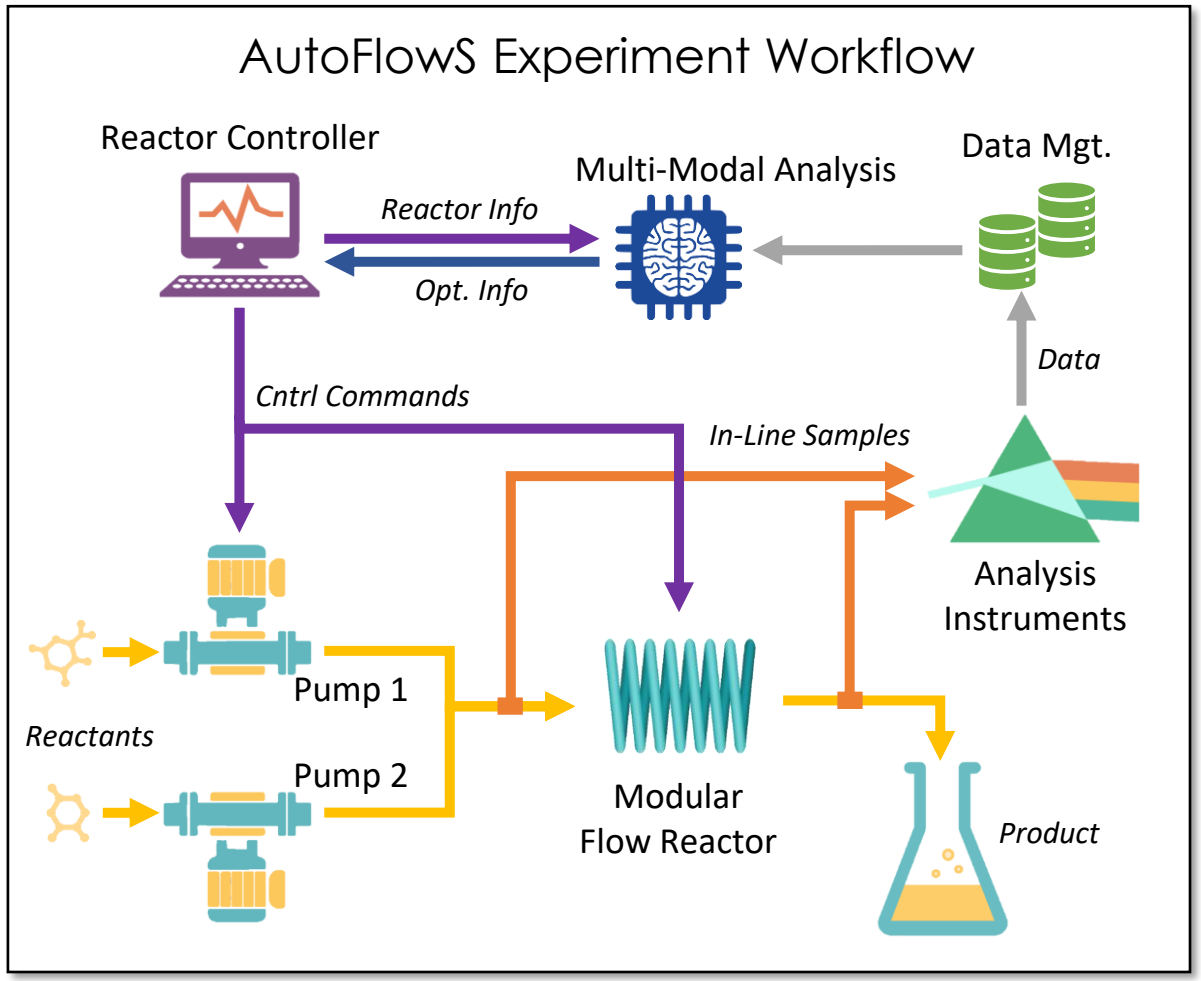
Step 4: Edge analysis of the data

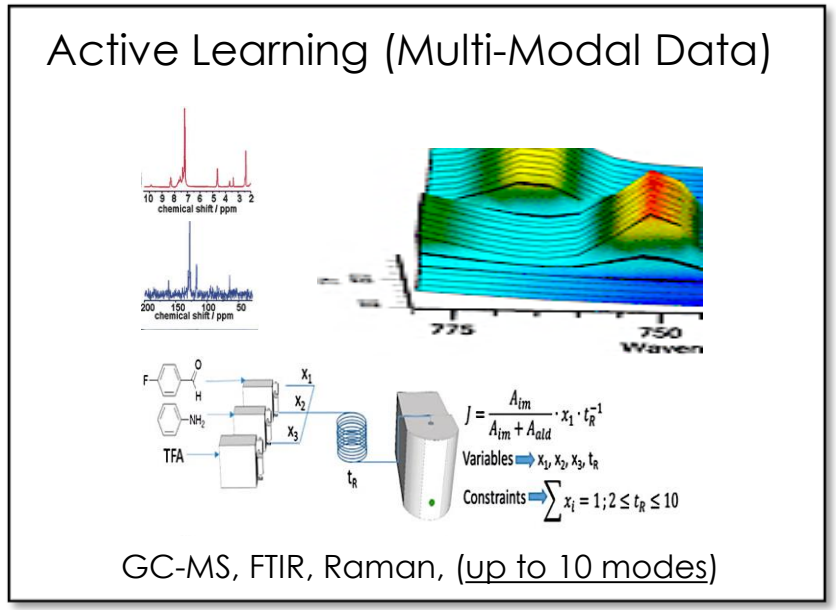
Step 4.5: Data transfers from System 3 to System 4

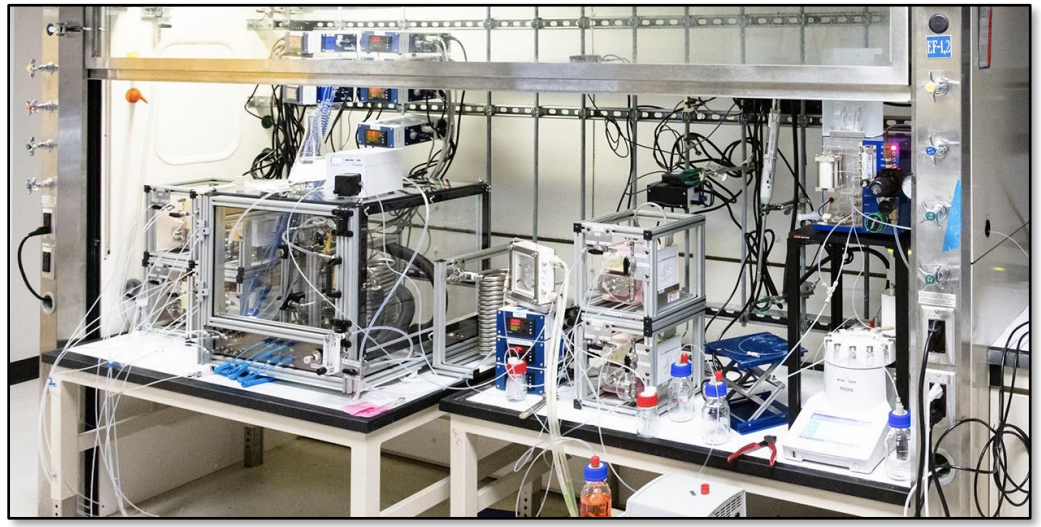
OAK RIDGE National Laboratory

INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

Autonomous Continuous Flow Reactor Synthesis (AutoFlowS)



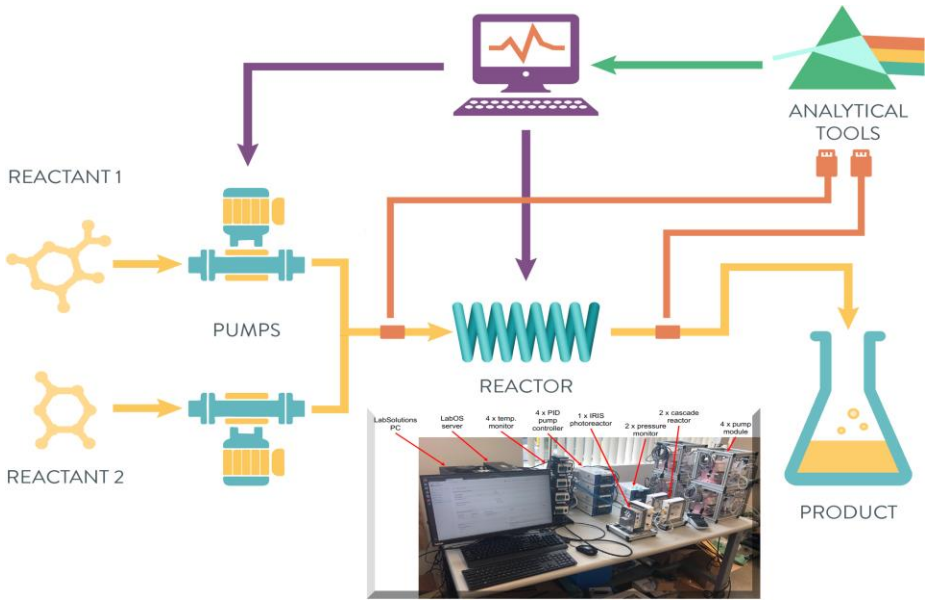


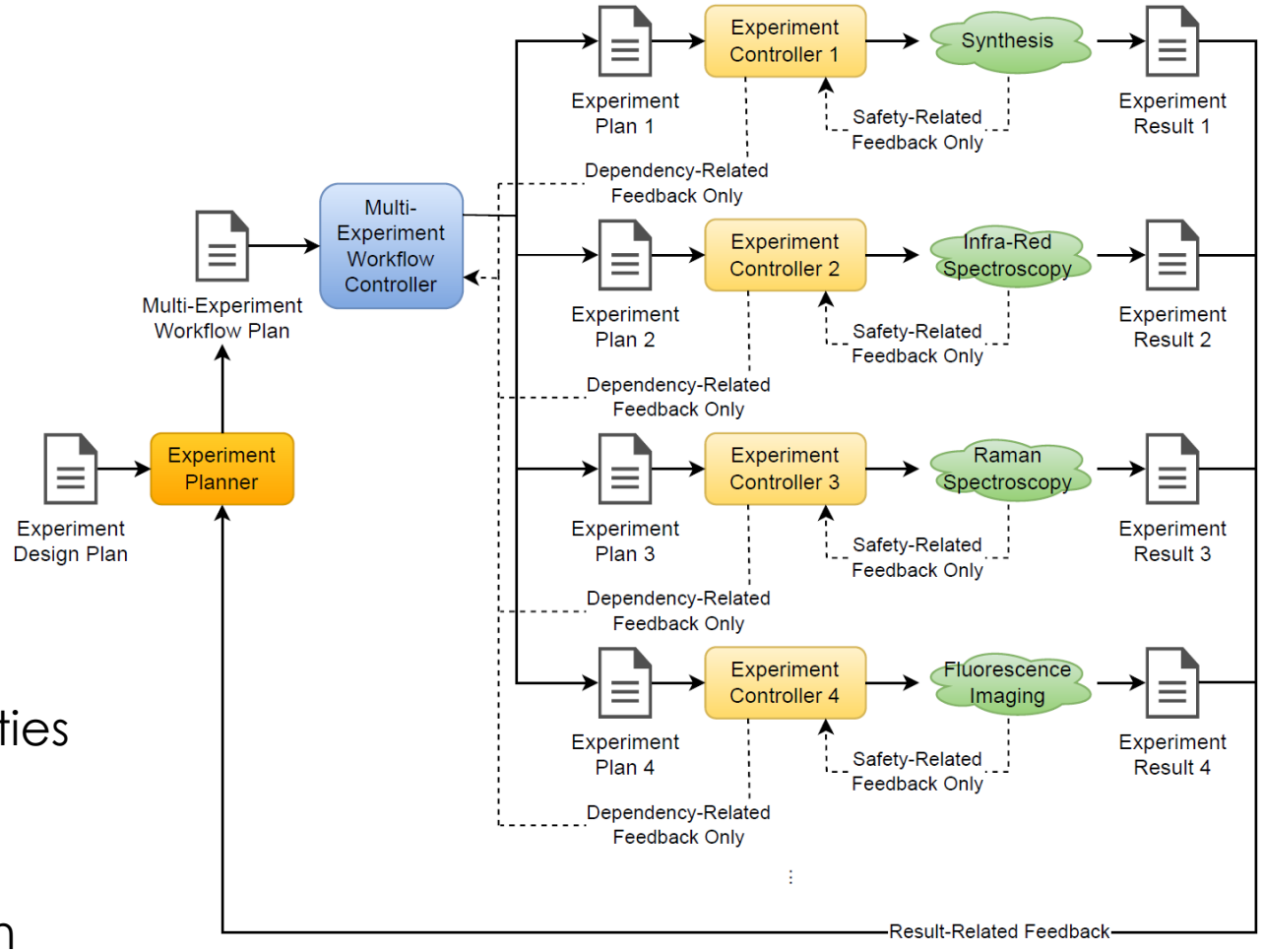


OAK RIDGE National Laboratory

INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

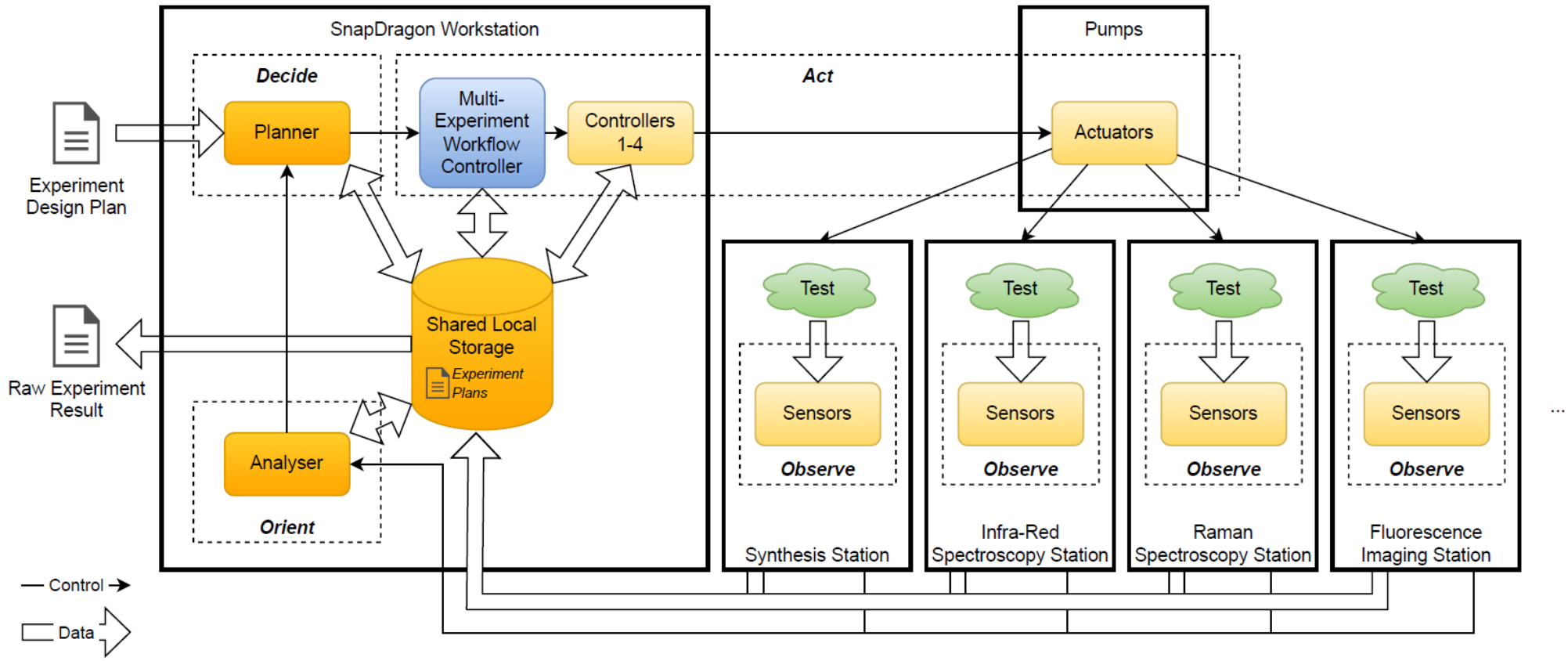
INTERSECT – Connecting the Pieces





- Requires plug-n-play capabilities as the synthesis chamber and analysis capabilities changes depending on chemistry
- Requires multi-dimensional datasets for AI/ML training
- Requires workflow agent to connect with HPC (high performance computing) or edge computing (DGX box) depending on chemistry

INTERSECT – Connecting the Pieces



Mapping existing hardware onto architectural components and combining through design patterns allows for quick integration of the project into the ecosystem for developing customizable workflows.

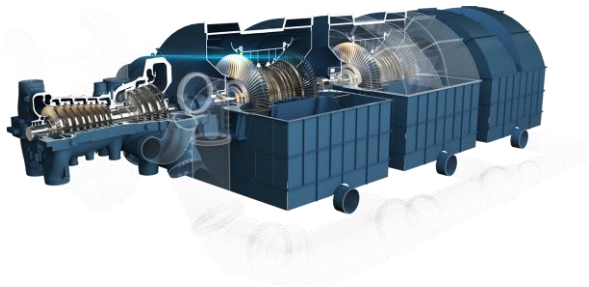
Electrical Grid

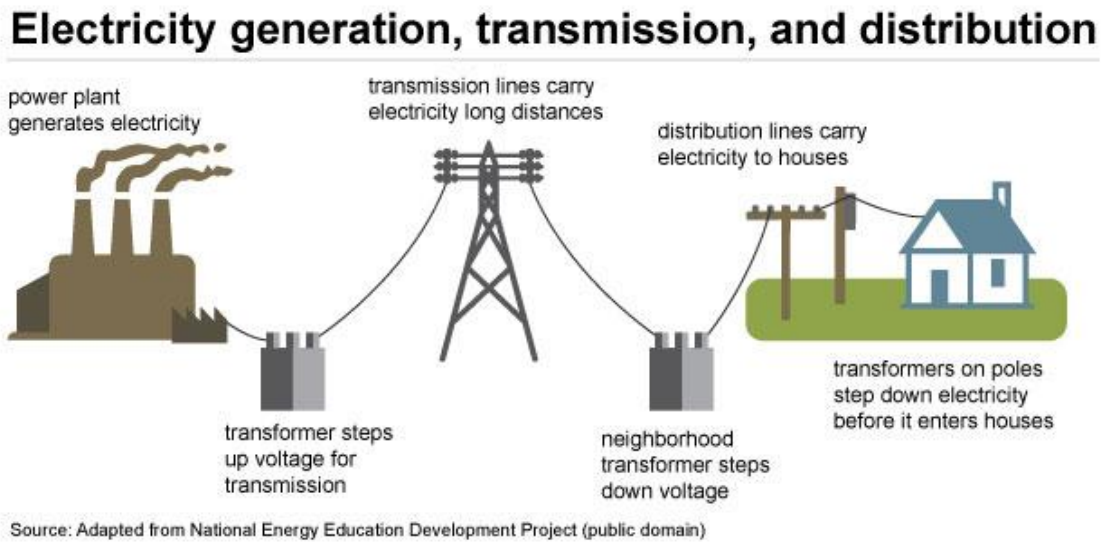
The Grid – We Take it for Granted

The Grid – We Really Notice When it Fails

Northeast Blackout of 2003

The Grid of Today

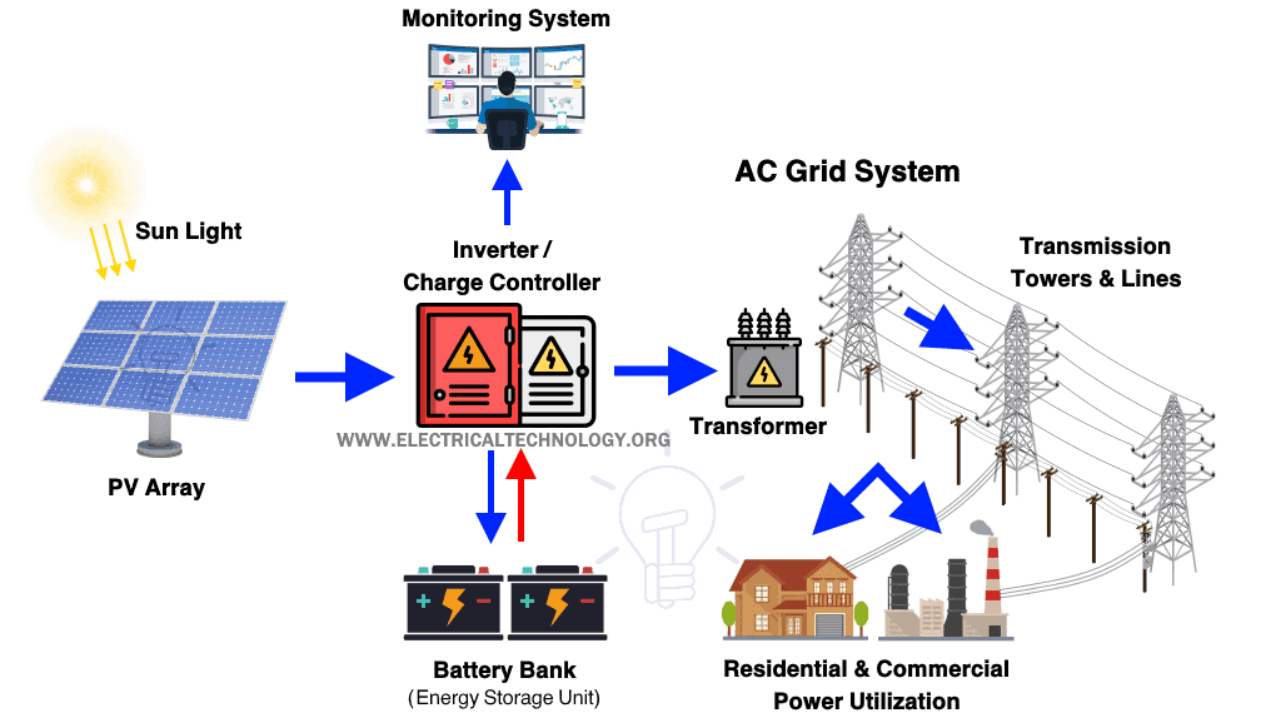




OAK RIDGE National Laboratory

INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

The Grid of Tomorrow



OAK RIDGE National Laboratory

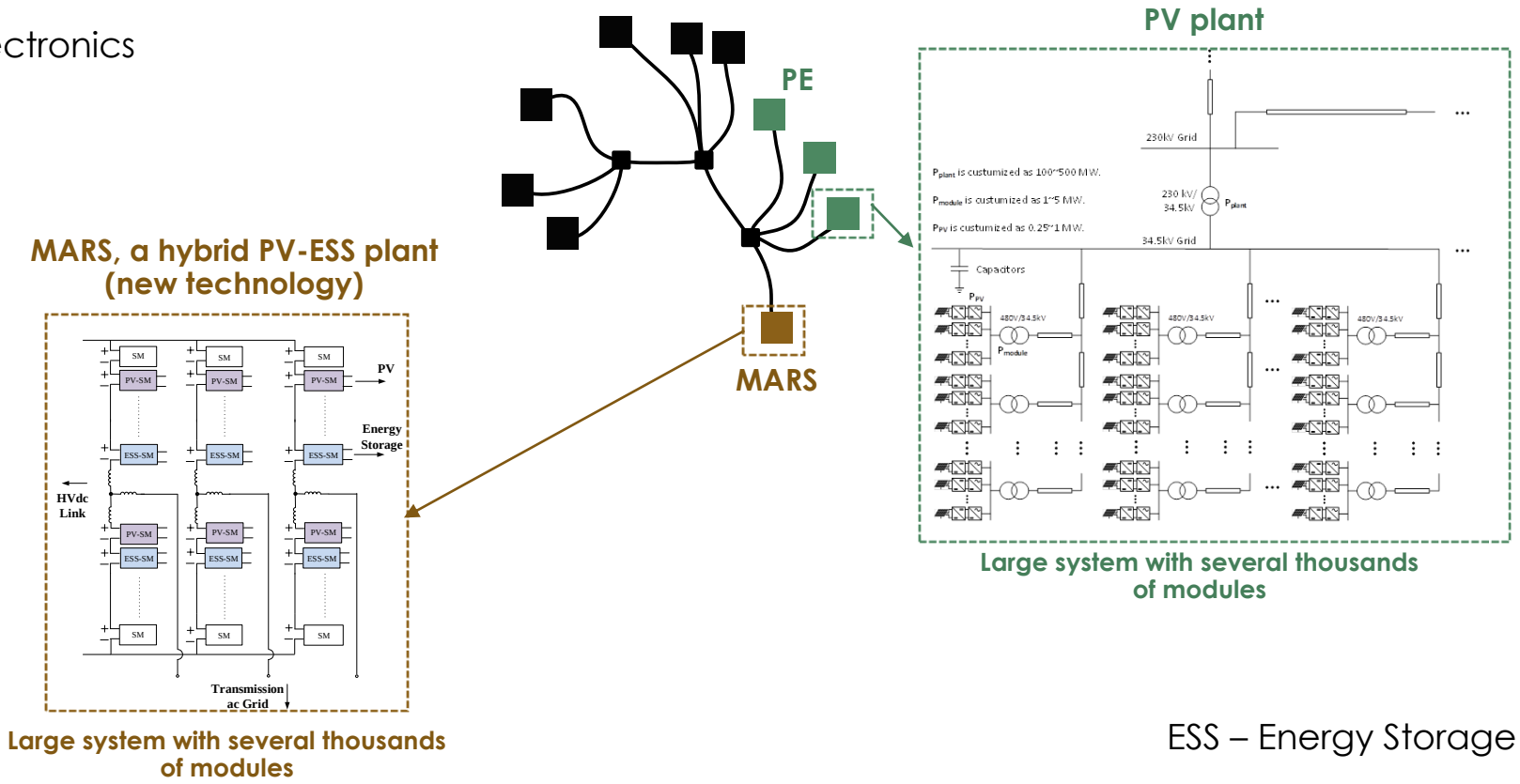
INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

How to Understand the Grid of Tomorrow

PV – Photovoltaic

PE – Power Electronics

Representative power grid with PEs

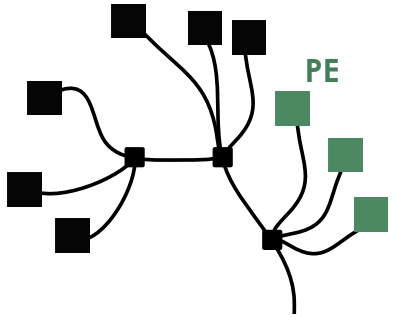


ESS – Energy Storage System

Mars – Multi-port Autonomous Reconfigurable Solar power plant

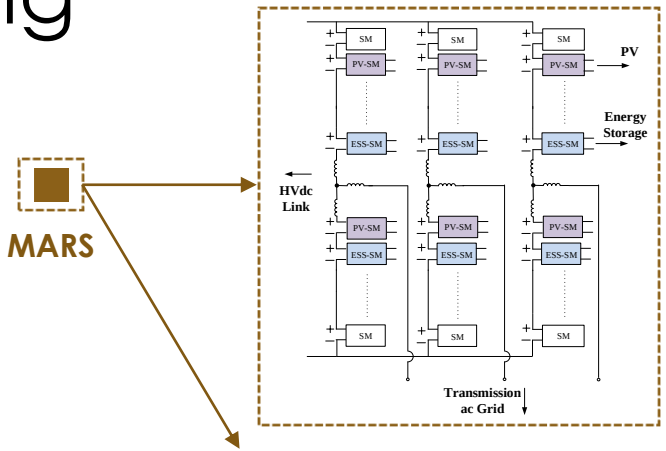
Current Path Towards Grid Modeling

Representative power grid with PEs



Step 1: Simulate power grid with PEs (without physical equipment or control node) in PSCAD based on data from MARS control nodes – DESIGN OF EXPERIMENT

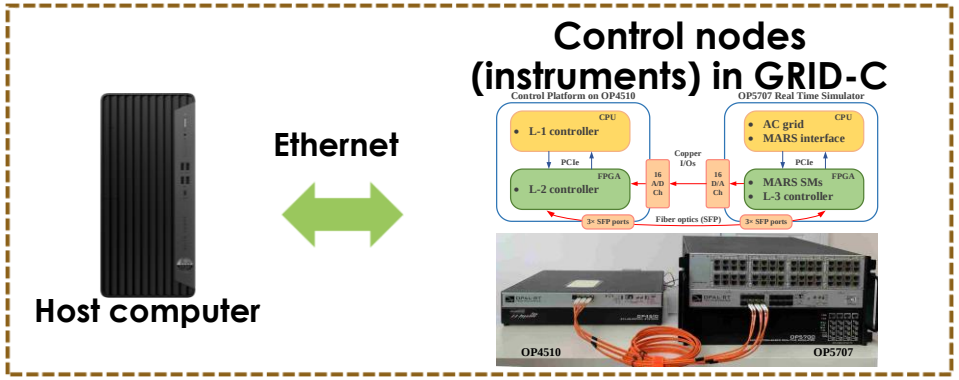
Human-in-loop: Transfer data (non-real-time) to MARS control nodes from PSCAD simulations



MARS, control nodes in GRID-C

Human-in-loop: Transfer data (non-real-time) to PSCAD simulations from experimental runs

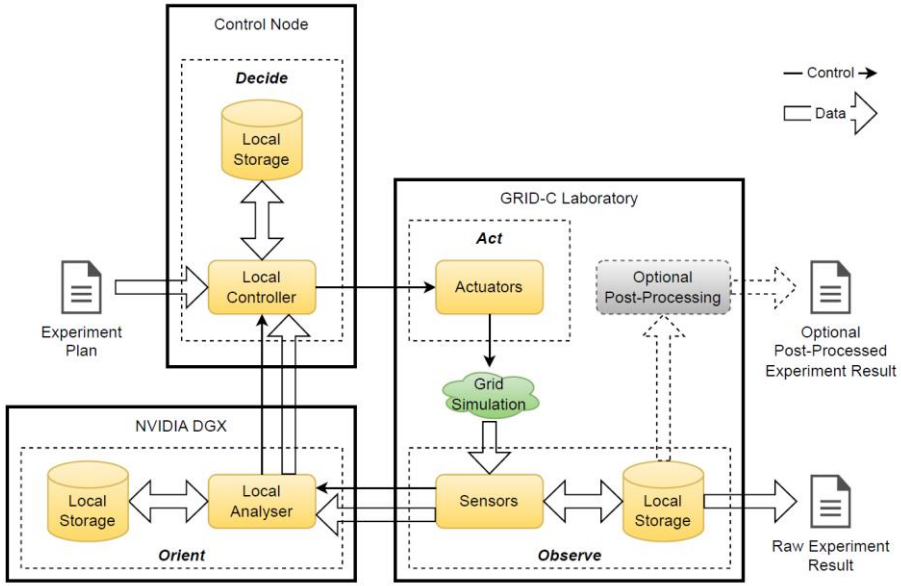
Control nodes (instruments) in GRID-C

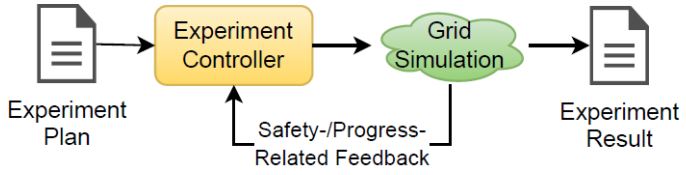


Step 2: Run MARS control nodes with the transferred data from PSCAD– EXPERIMENT

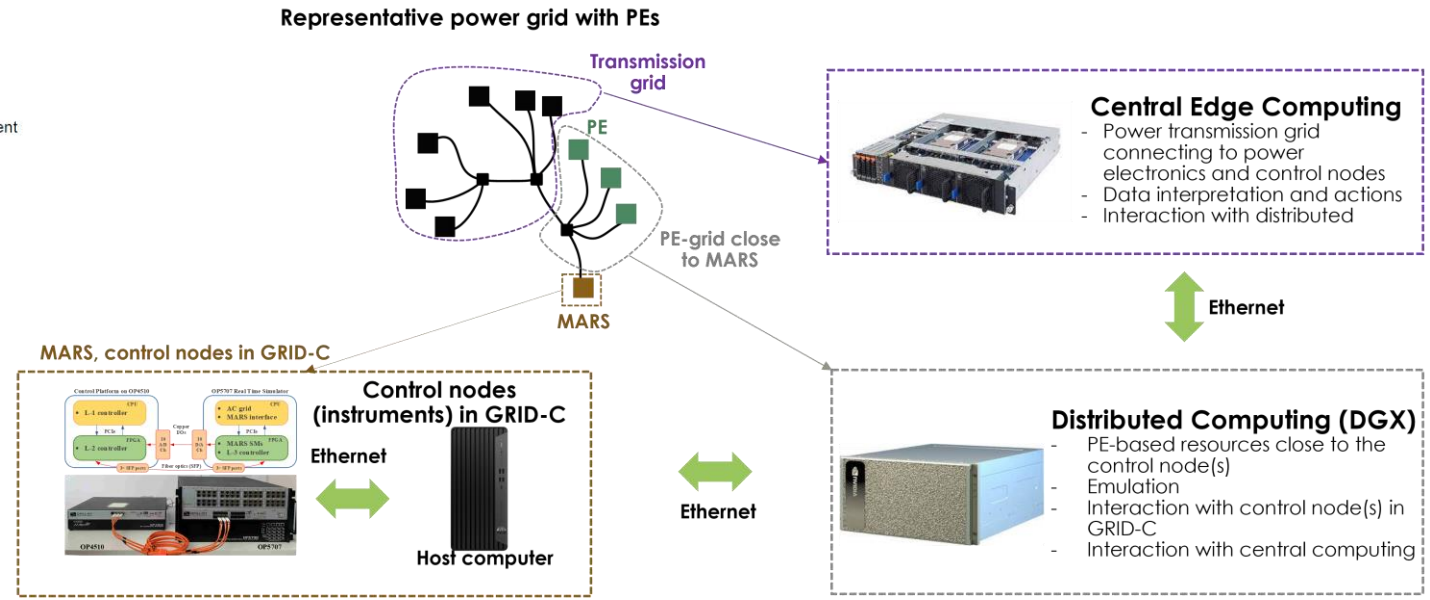
Iterative design of experiment and experimental runs needed for convergence in the above process to evaluate new real-world physical equipment – typical process time expected is of the ***order of several weeks***

INTERSECT – Connecting the Pieces





- Different parts of the grid emulated on different computational resources based on proximity to the MARS
- Observe response of MARS power electronics as changes/disturbances are injected into the grid
- Latency is crucial for real-time emulation
- AI/ML optimizes power electronics parameters to increase grid reliability

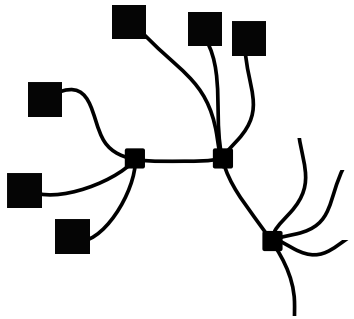


OAK RIDGE National Laboratory

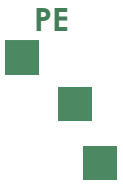
INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

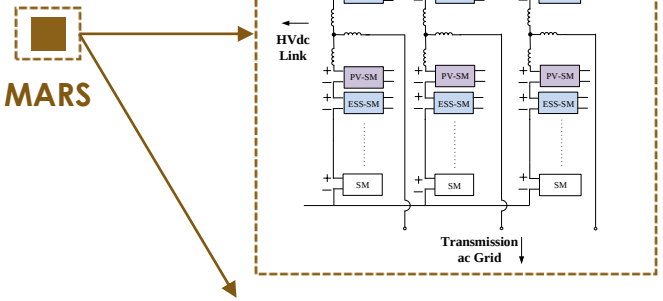
INTERSECT – Connecting the Pieces

Representative power grid



PEs: PV Plants





MARS, control nodes in GRID-C

Low latency data transfer (real-time)

Low latency data transfer (real-time)

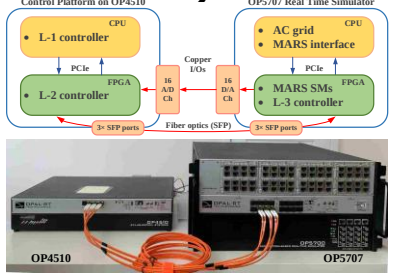
Low latency data transfer (real-time)

Low latency data transfer (real-time)

Host computer

Ethernet

Control nodes (instruments) in GRID-C



Central Edge Computing:
Power grid fast simulation with AI/ML for speed-up, experimental control & status monitoring

Control plane

Distributed Computing:
Fast simulation of PEs, status monitoring

Control plane

Run MARS control nodes with the real-time data transfer

OAK RIDGE National Laboratory

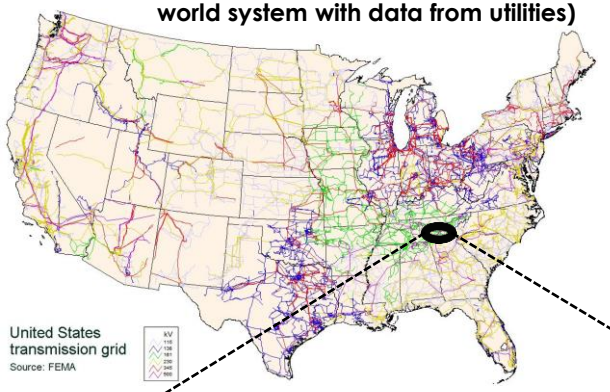
INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

Grid Emulation and Control of Tomorrow

External Laboratory Connectivity
- Interconnected national laboratories
- Potential international connection

Ethernet (ESnet)

US energy system and power grid (real-world system with data from utilities)

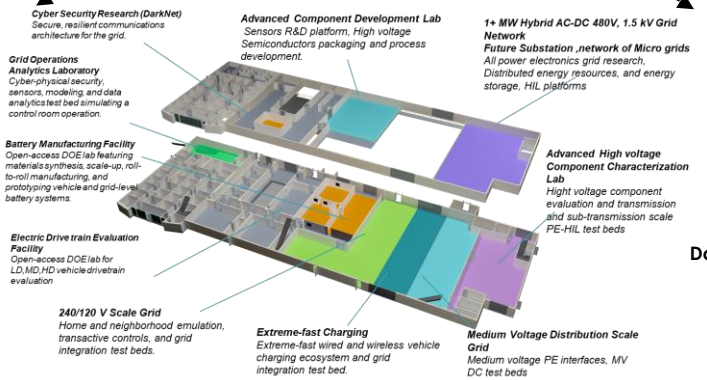


Grid data from real-world systems (eg, utilities)

Central edge computing
- Physics-based models (larger power grid)
- AI-based models
- Interface with laboratory computing resources
- Data interpretation and scientific observation (actions)
- Interaction with distributed computing

Data exchange with digital twinning of labs

Visualization dashboard



GRID-C laboratory at ORNL
- Physical equipment/control nodes

Data exchange with labs

Distributed computing
- Interface with real-time and central edge computing resources
- Interconnected laboratories
- Twin laboratory equipment with power electronics

Scalable Interconnected Laboratories for Large-Scale Connectivity and Real-World Energy Systems and Power Grid Emulation

Additive Manufacturing

Additive Manufacturing – 3D Printing of Anything

AM: Building a part layer-by-layer by melting small amounts of powder/wire at a time

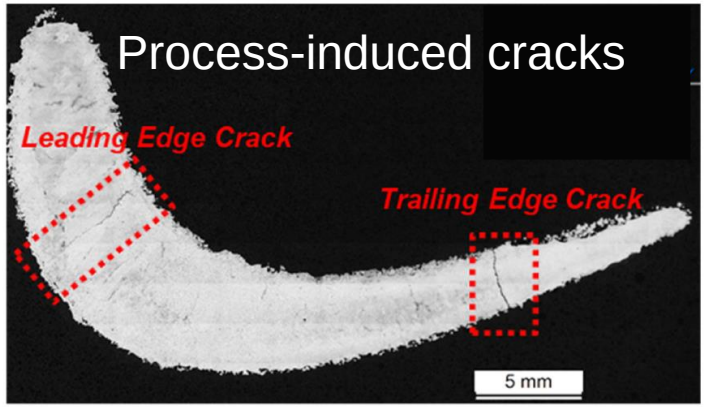
Potential for new, complex part topologies and optimized, location-specific material properties

A difficulty…

Part quality is sensitive to process parameters

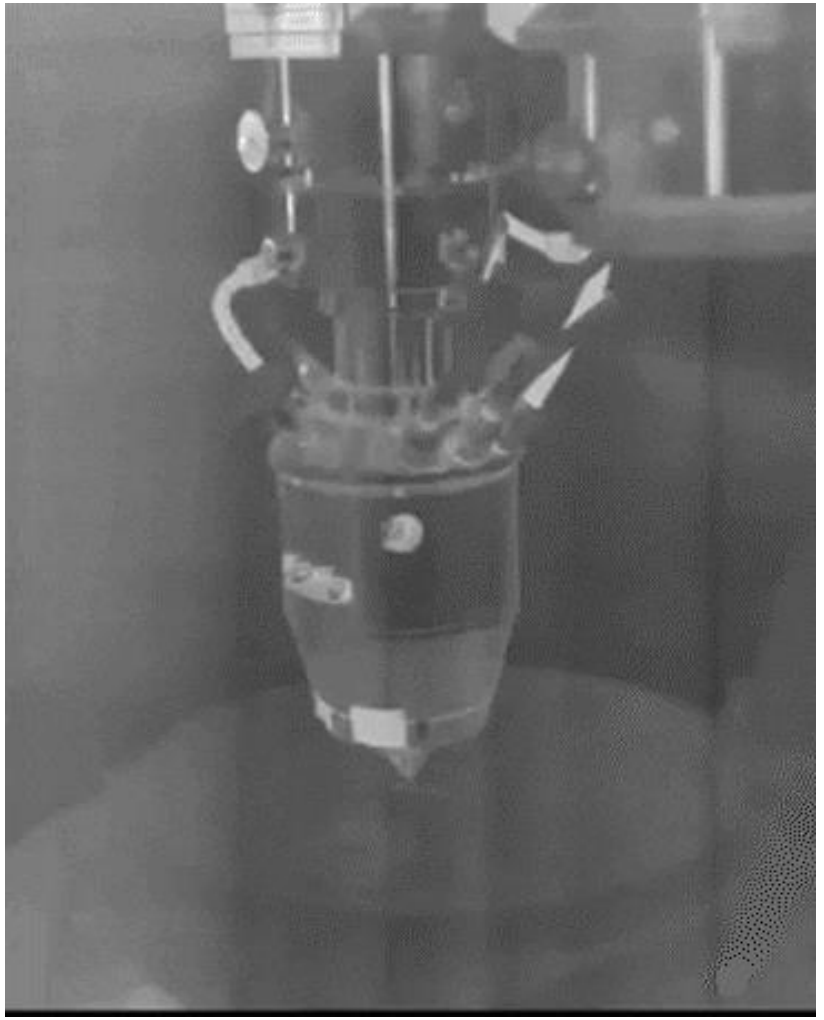
→

Any change can require unintuitive process changes



Produce p dy to use im ”

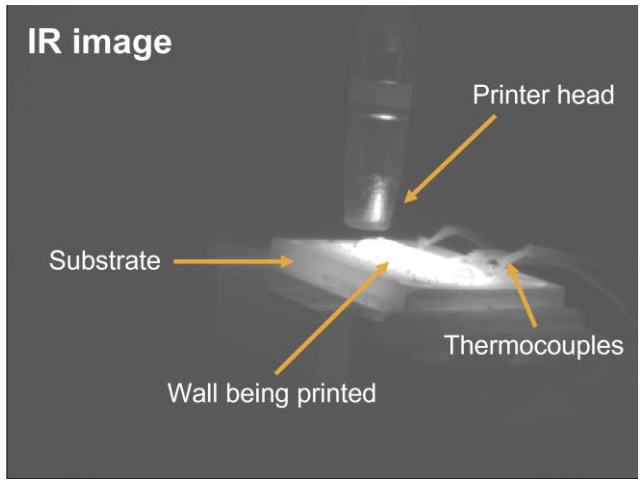
Auto ntrol



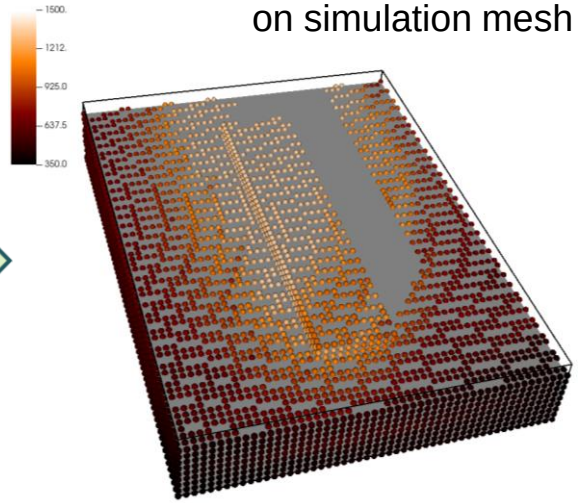
Lee, et al, Additive Manufacturing, 2020.

Connecting Data with Simulations

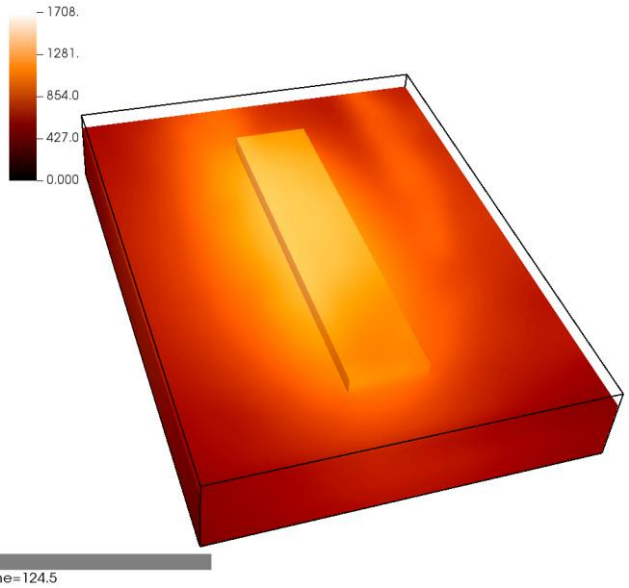
Experimental image

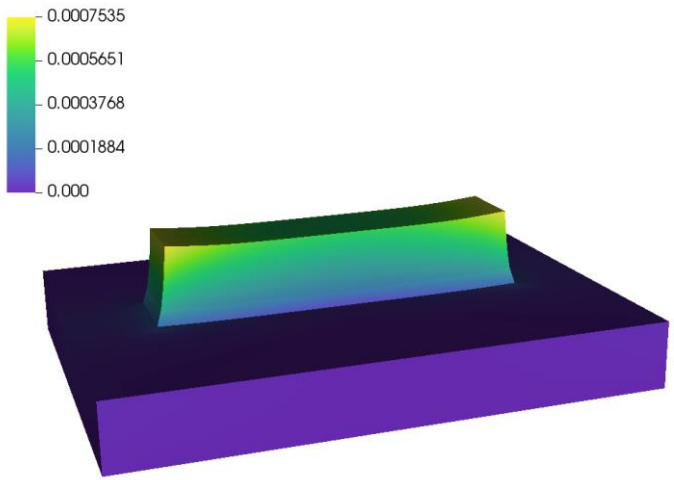


Experimental data projected on simulation mesh

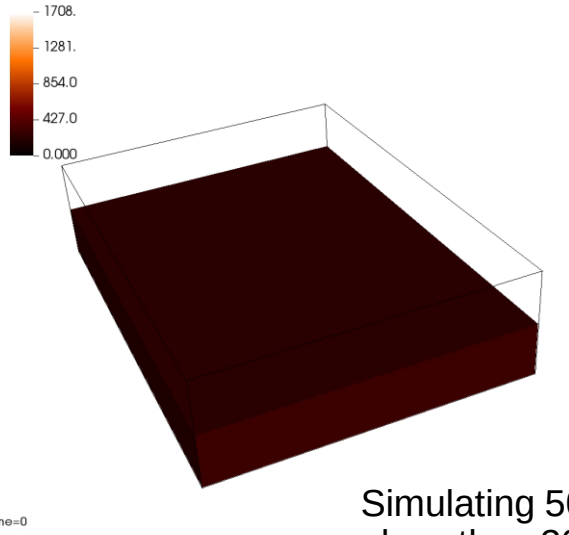


Data and simulation assimilation



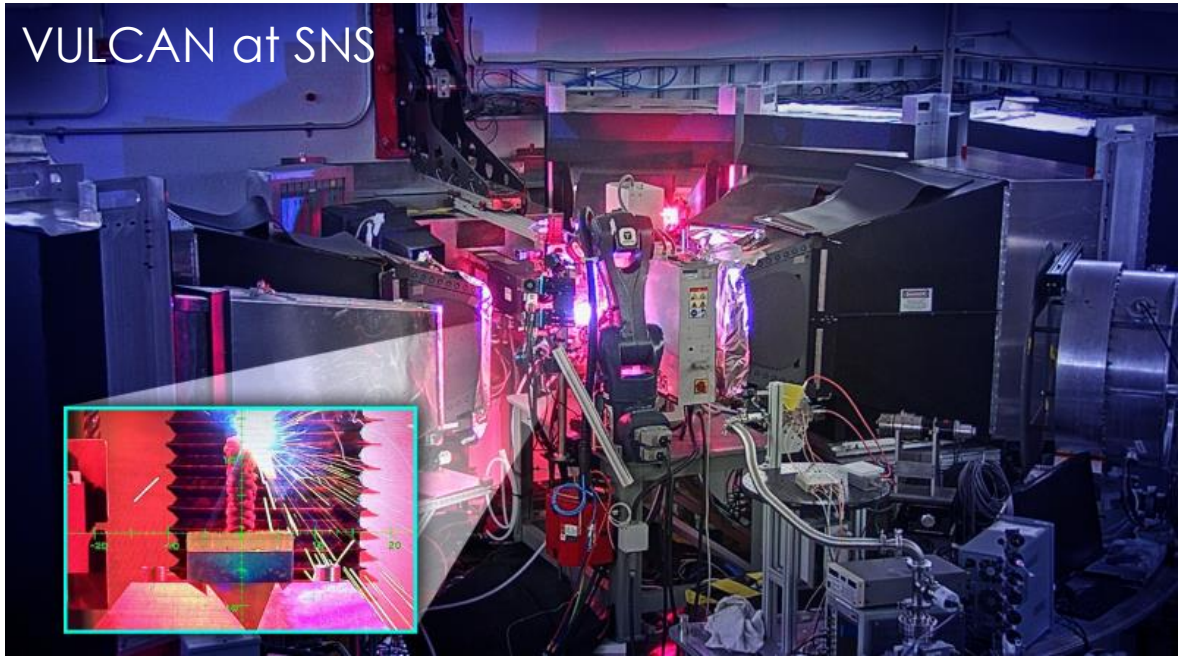


Thermomechanics

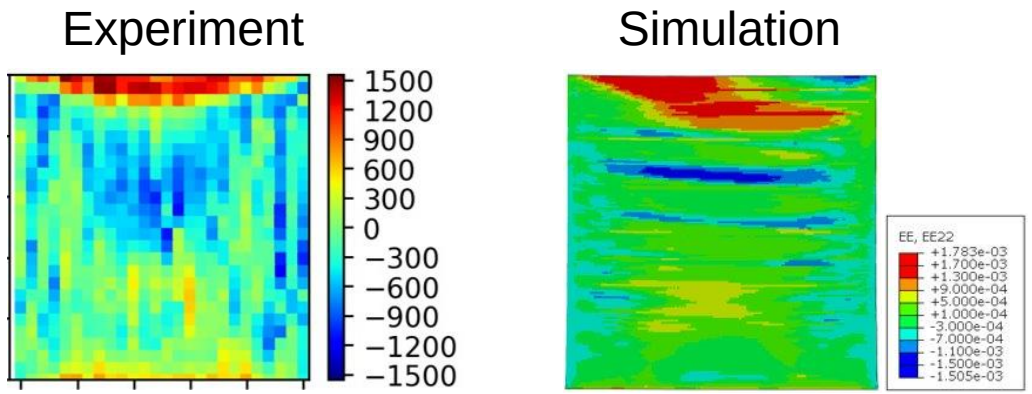


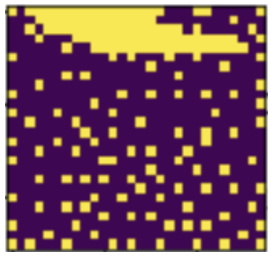
Simulating 500s of printing in less than 300s of wall time

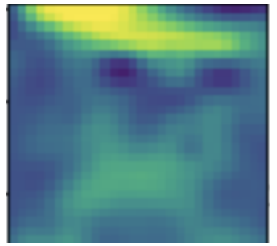
Connecting Data with Simulations

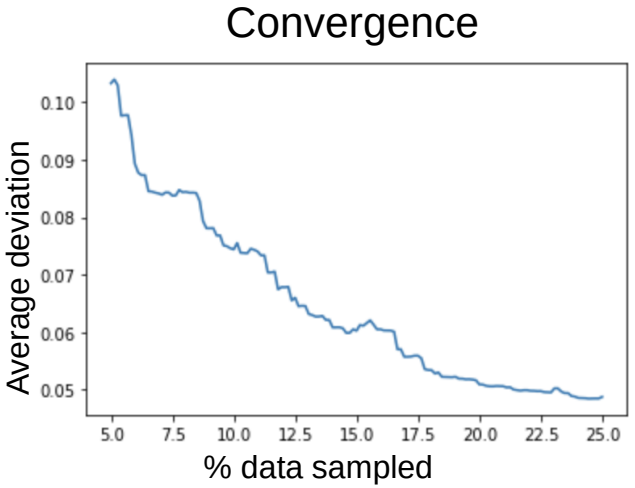


- Builds on "Digital Metallurgy" LDRD Initiative work for adaptive sampling at SNS
- Thermomechanical simulation to inform sampling
- Determines where in the part to collect neutron diffraction data to gain the most informative data







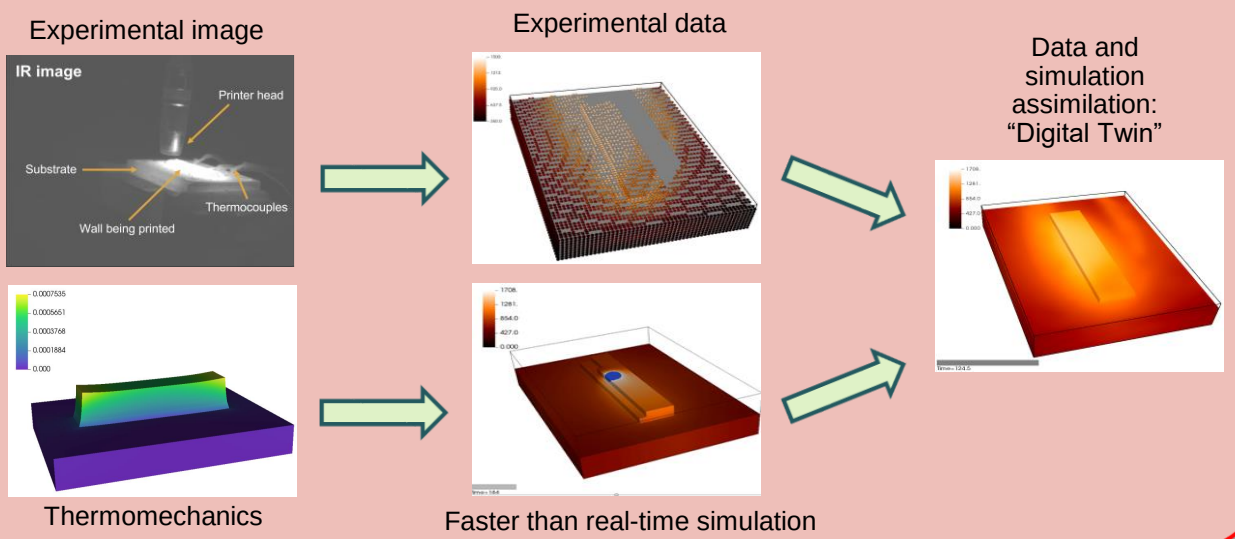


OAK RIDGE National Laboratory

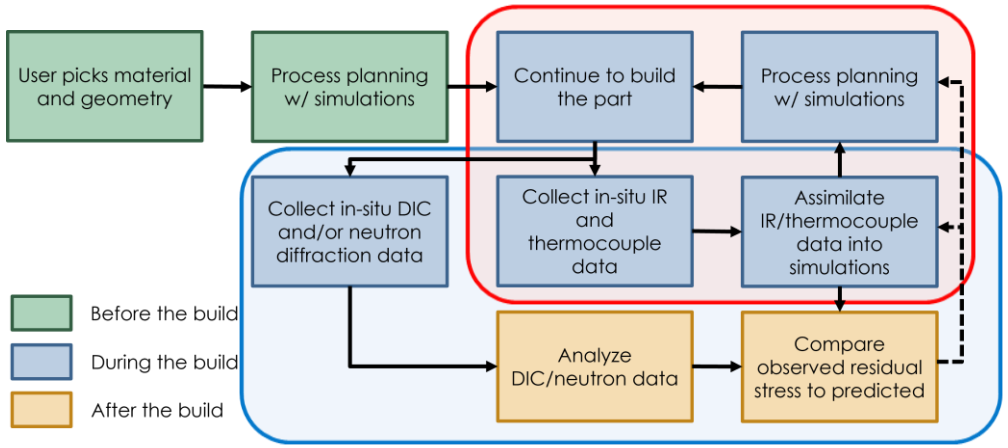
INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

Autonomous Additive Manufacturing

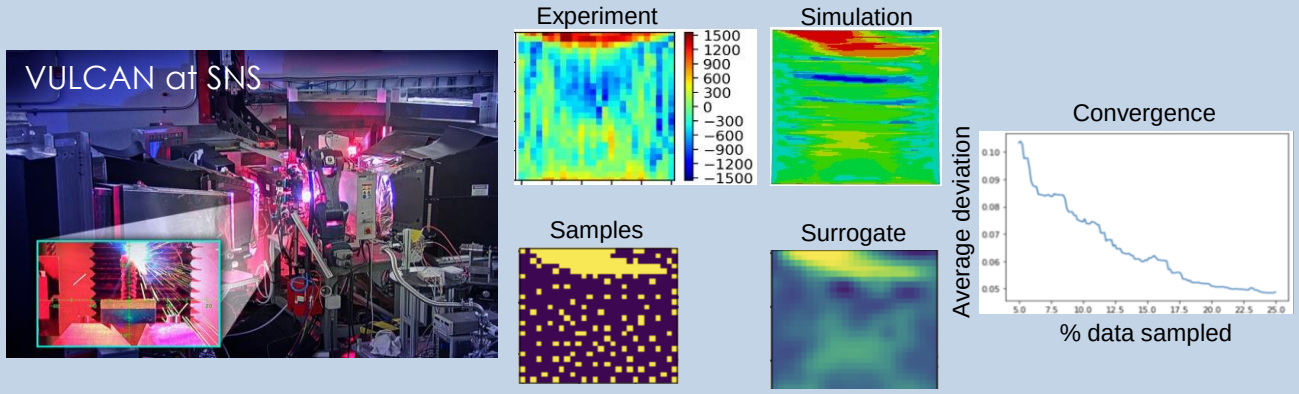
Additive Manufacturing Digital Twin



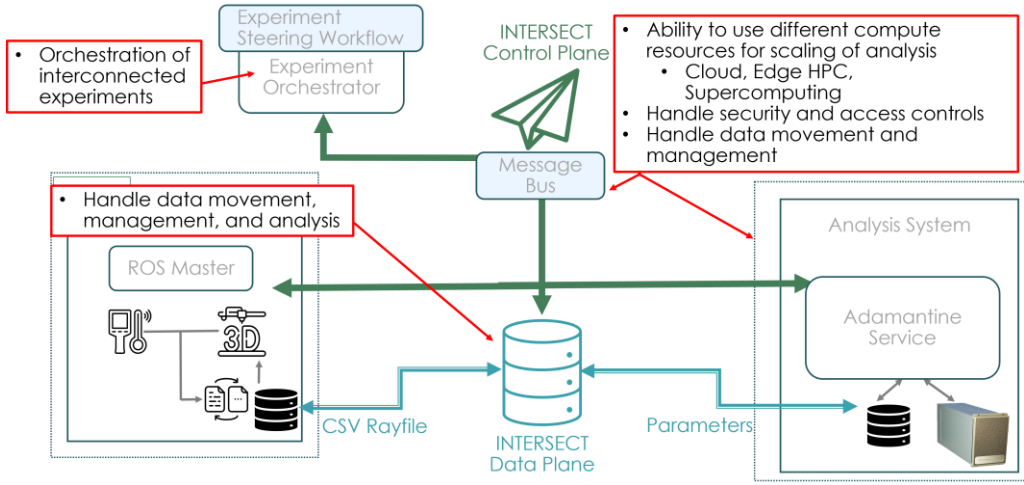
Additive Manufacturing Autonomous Workflow



Additive Manufacturing / SNS Integration



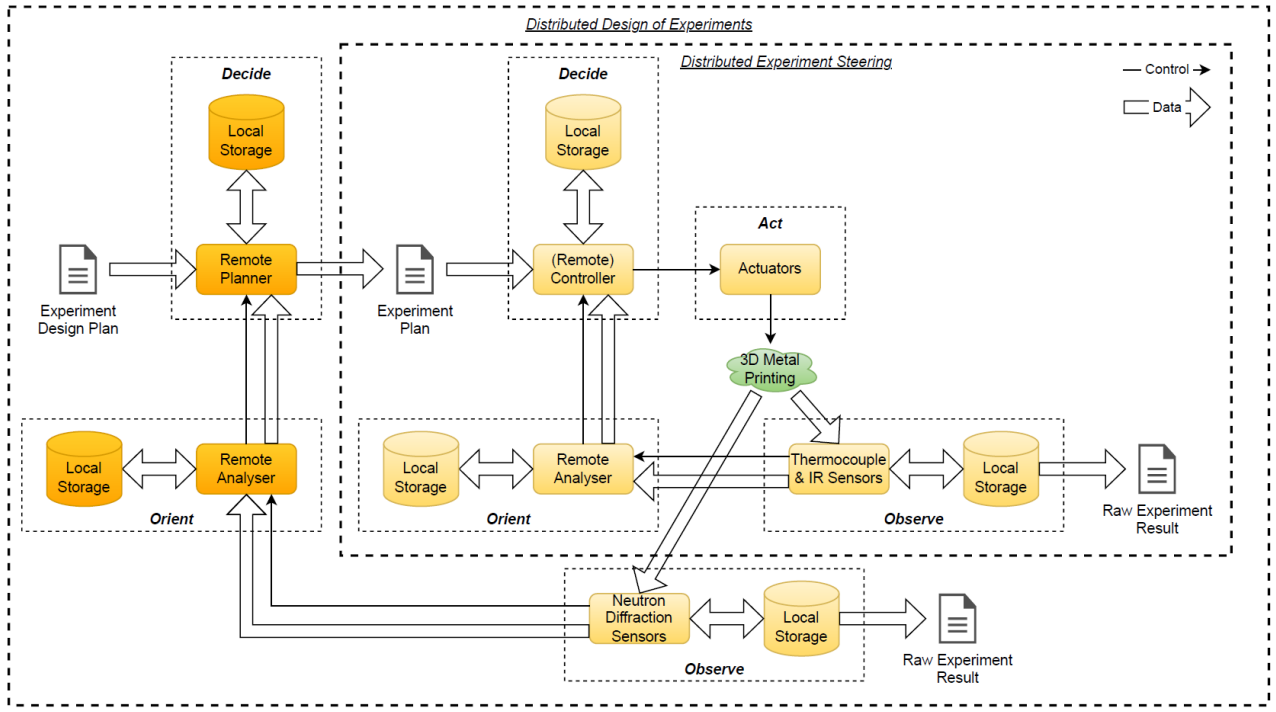
INTERSECT Software Services



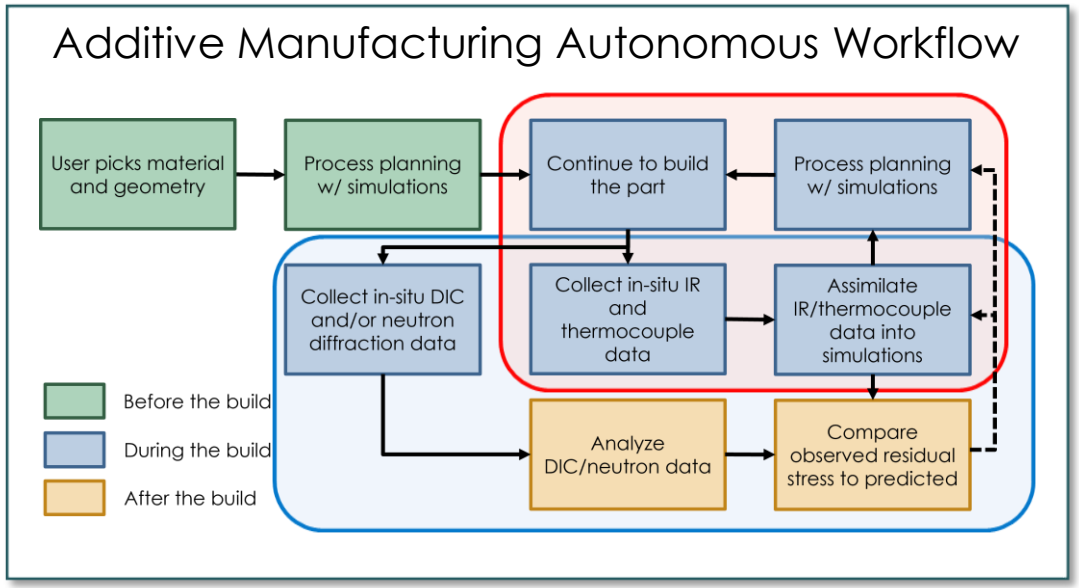
OAK RIDGE National Laboratory

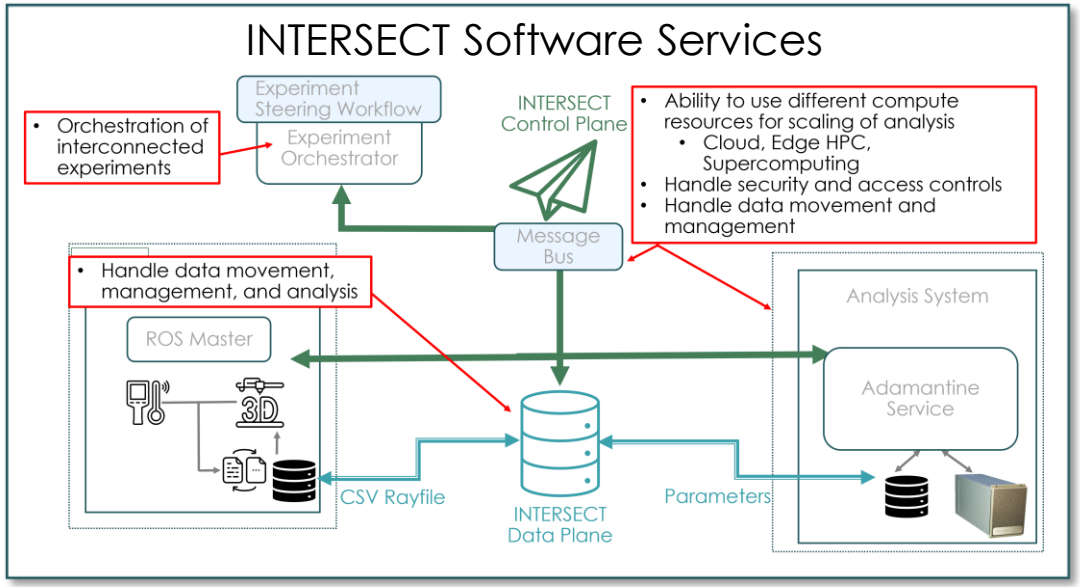
INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

Autonomous Additive Manufacturing



- Interconnected workflows – IR Sensors and Neutron Diffraction
- Mix of real-time AI/ML steering during build and AI/ML steering of neutron experiments after the build
- Reuses AI/ML codes from 4D-STEM Experiments

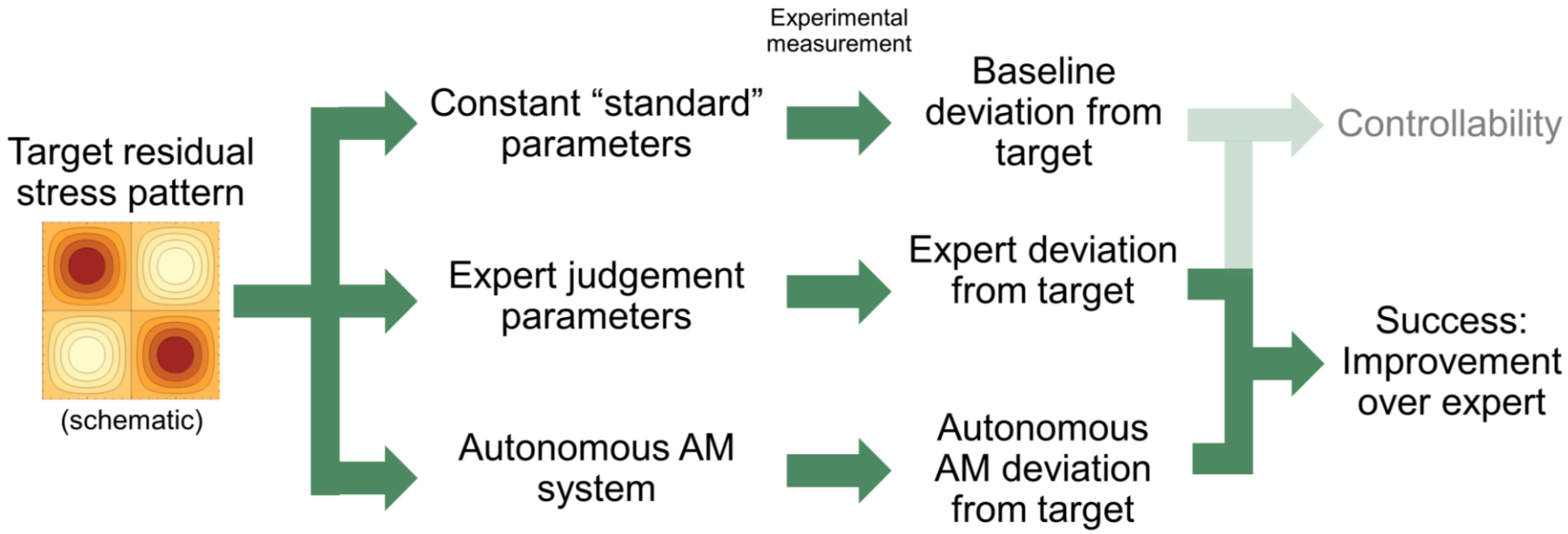


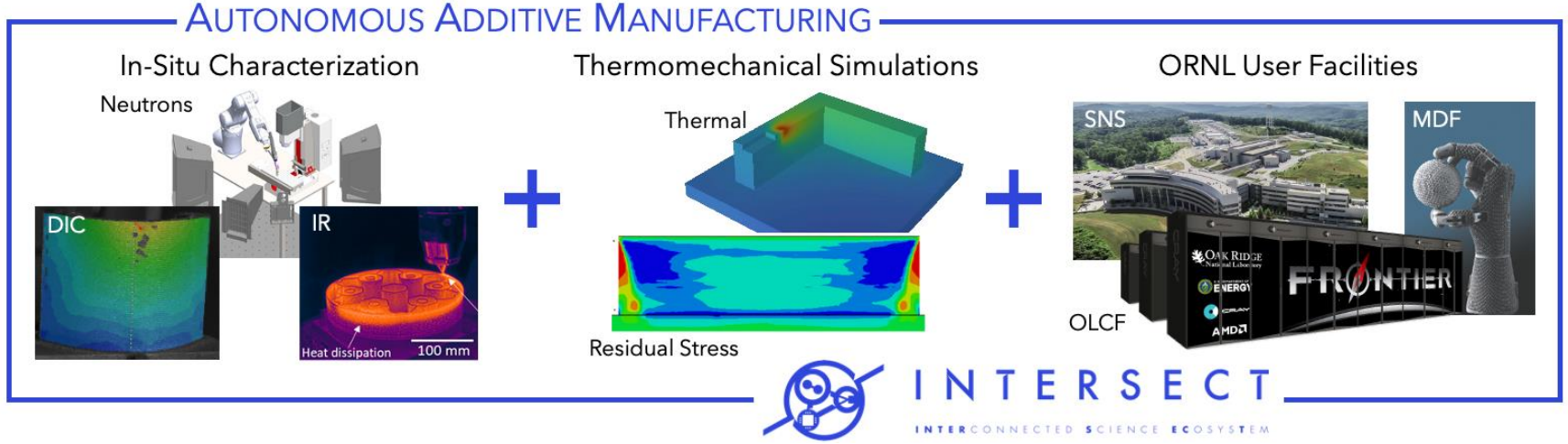


OAK RIDGE National Laboratory

INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

Validation and Certification





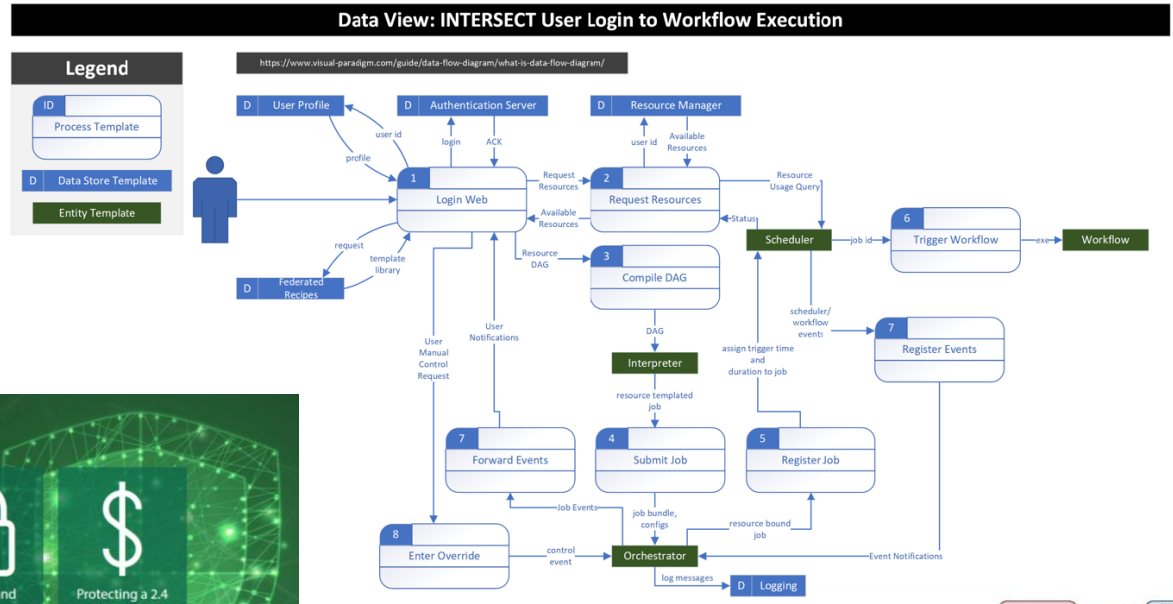
OAK RIDGE National Laboratory

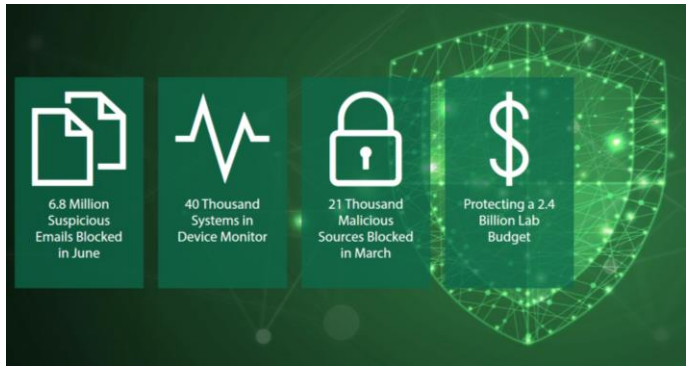
INTERSECT INTERCONNECTED SCIENCE ECOSYSTEM

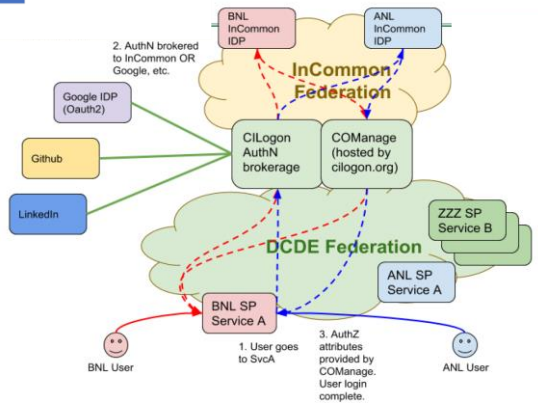
Things to Consider – Can't Forget the Other Views

General Data Protection Regulation
GDPR

gaia-x







Handling Data and Users

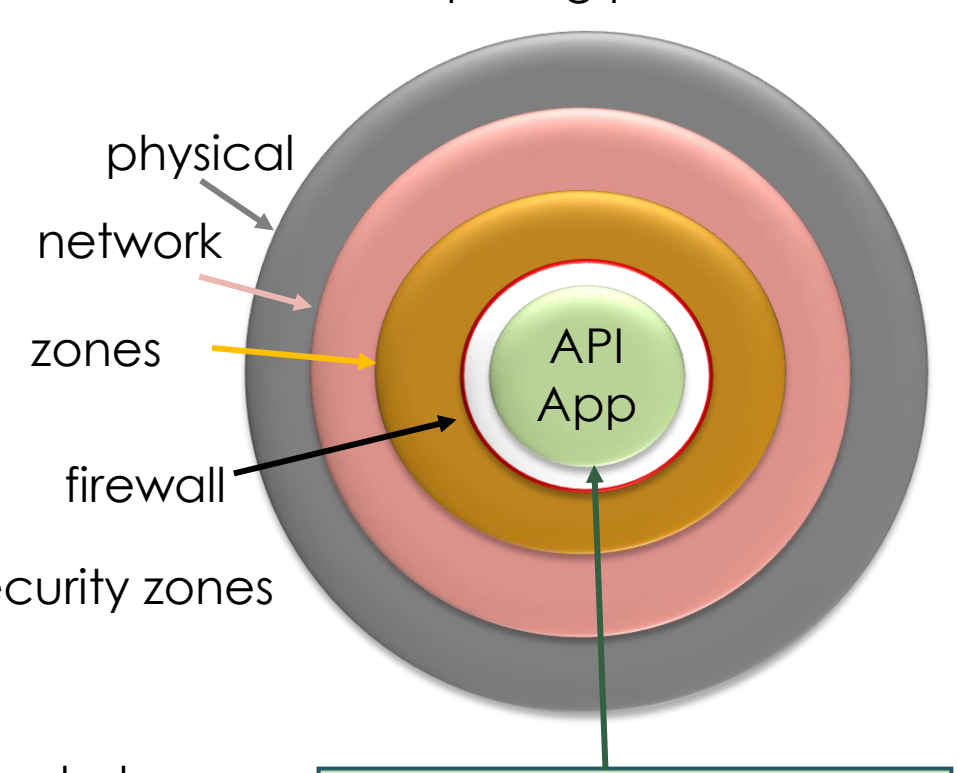
- Personal Information / Sensitive Information / Intellectual Property
- How to handle multi-factor authentication on different machines/facilities
- Making connections across different firewalls and different facilities with different policies
- Defining who can do what with what part of your data

OAK RIDGE National Laboratory
INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

Integration: Multi-Faceted Challenges

Requirements for integration into ecosystems:

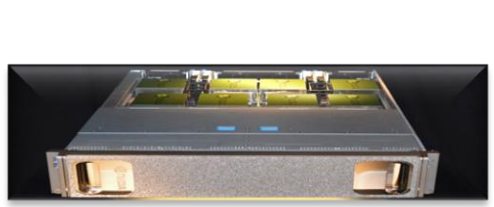
- **Physical** layer network connectivity
  - Basic requirement wireless, copper or fiber
- **Network** connectivity
  - Subnets and routing need to be configured
  - Many instruments network cannot be routed outside – ORNL quarantine
- **Security zones** alignment
  - many instruments and computing systems in different security zones
- **Firewalls**
  - Subnetworks typically firewalled
  - Firewall exception require understanding of implications – to be requested by edge providers
  - Host firewalls need customization for instrument computers
- **API** for network access
  - Many instrument APIs are not network accessible – need wrappers
- **Application**-level network access
  - Required deeper understanding of layers and software environment



Integration: Custom Instrument Computers

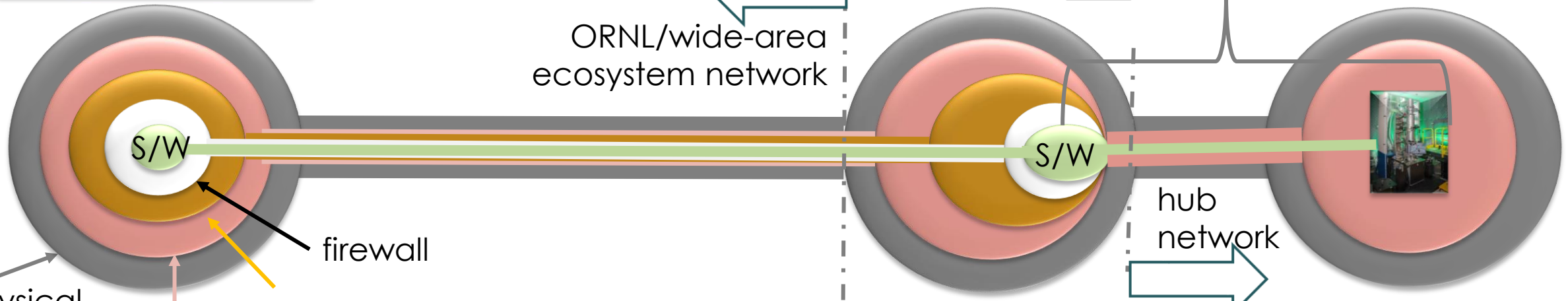
Instrument computers typically:

- Only connected to instruments via private local networks
  - several not allowed to connect to ORNL networks – policy implementations
  - impractical to connect to ORNL networks – reboots in middle of experiments
  - instrument software does not work on ORNL computers – policy implementation
- Windows OS – some older versions
- Very few APIs – needed for automation
- Very few network-capable application software – needed for remote orchestration



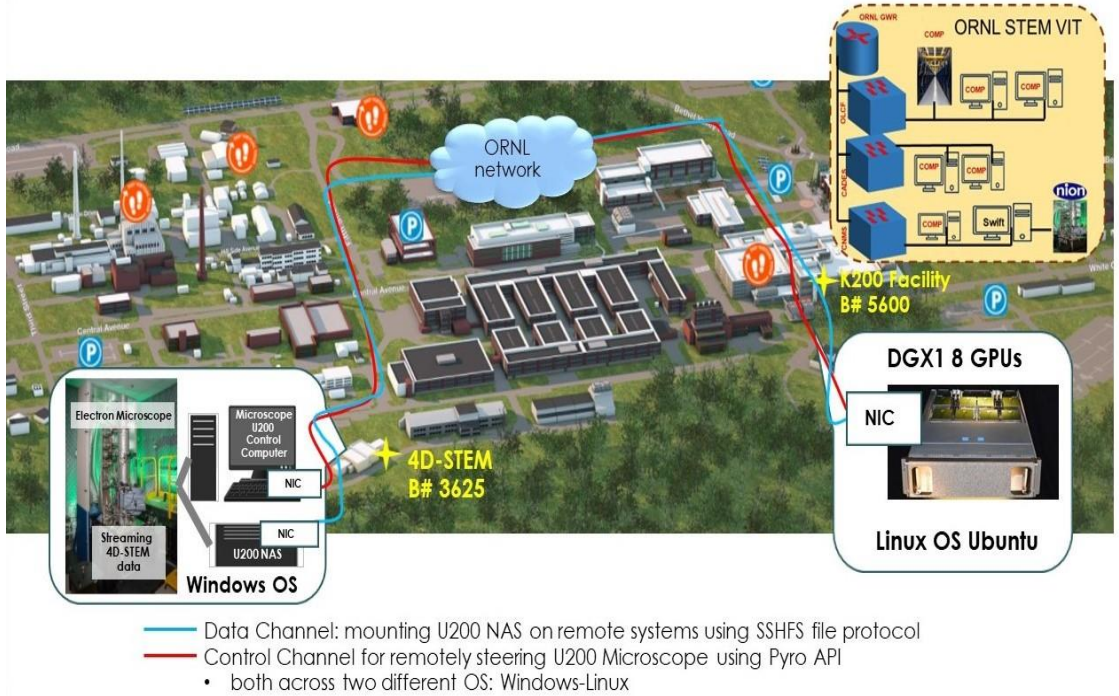
Instruments
- not on ORNL network
- local-use Software

ORNL/wide-area ecosystem network

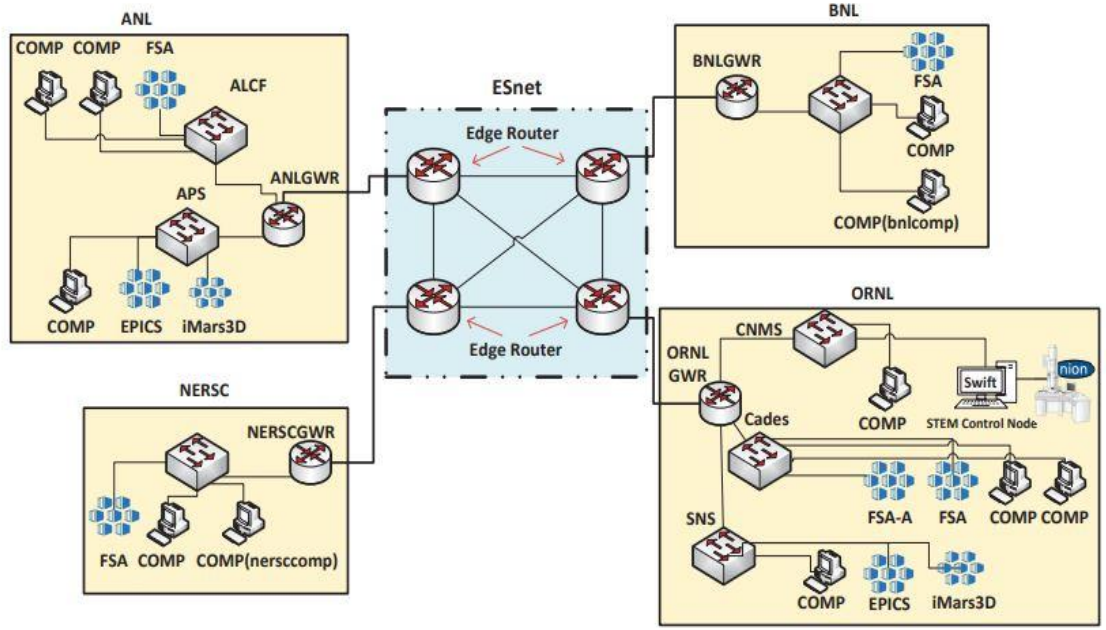


Across ecosystem all these layers need to be enabled and aligned
Communication between software need to cross all boundaries

Virtual Twins



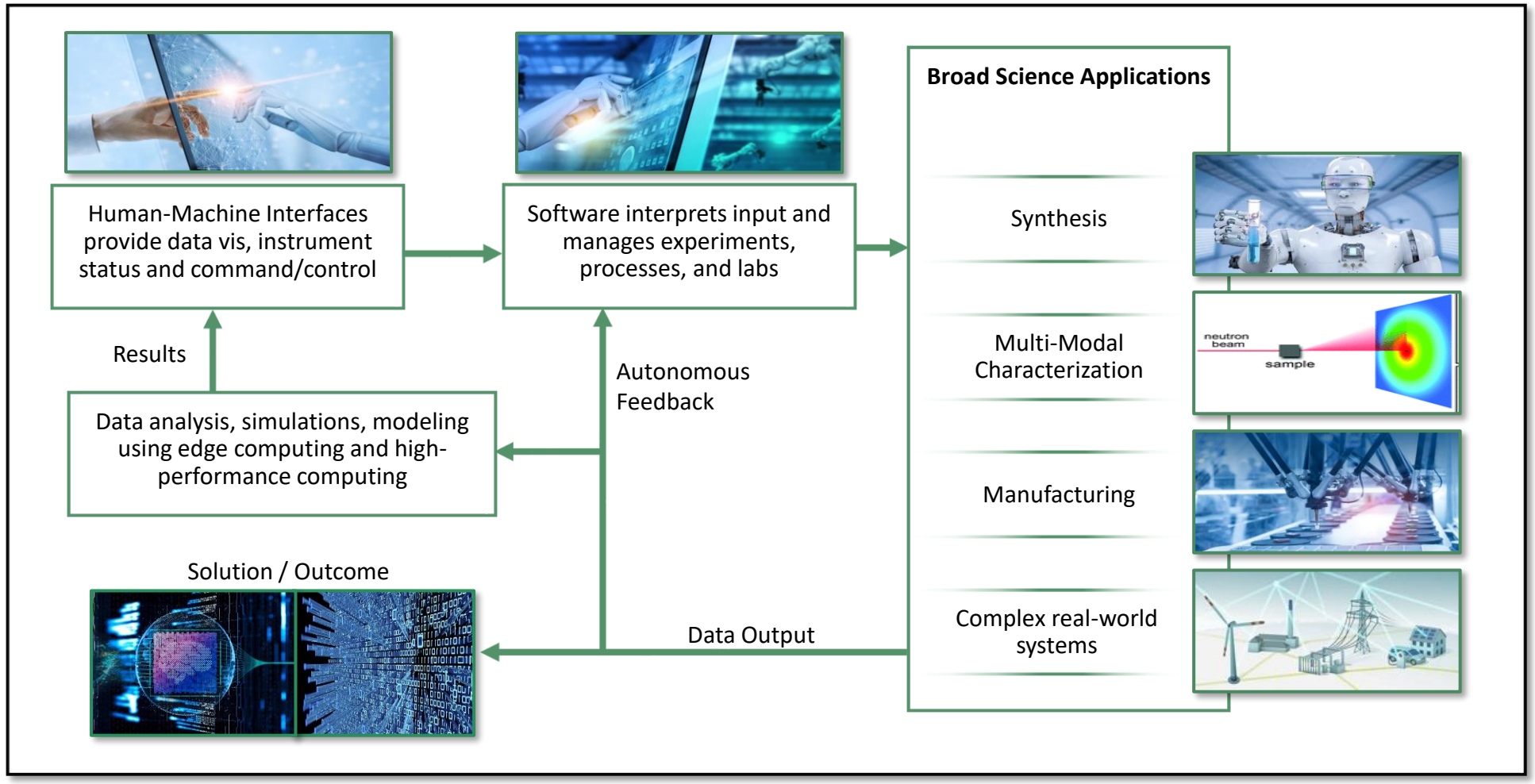
STEM workflows require orchestration experiments at Nion microscopes and remote computations. At ORNL, computing platforms, including DGX1, are located away from microscope facility to avoid interference due to vibrations. Separate network connections are setup between control computer and DGX1 for steering and measurement operations, and between network storage system (NAS) and DGX1 for cross-mounting files systems between NAS Windows and DGX1 Linux platforms.



VIT emulation of four national lab sites connected over ESnet. Each site is emulated with its computing platforms (Ubuntu Linux os) and local area network. The instruments are emulated by running instrument software on virtual hosts – Nion swift for STEM (scanning transmission electron microscope) at Oak Ridge National Laboratory (ORNL).

- Virtual Twins or Virtual Infrastructure Twins, i.e. emulation of networked resources and instrumentation, allows for development of autonomous workflows without disrupting ongoing research

Smart Autonomous Interoperable Laboratories

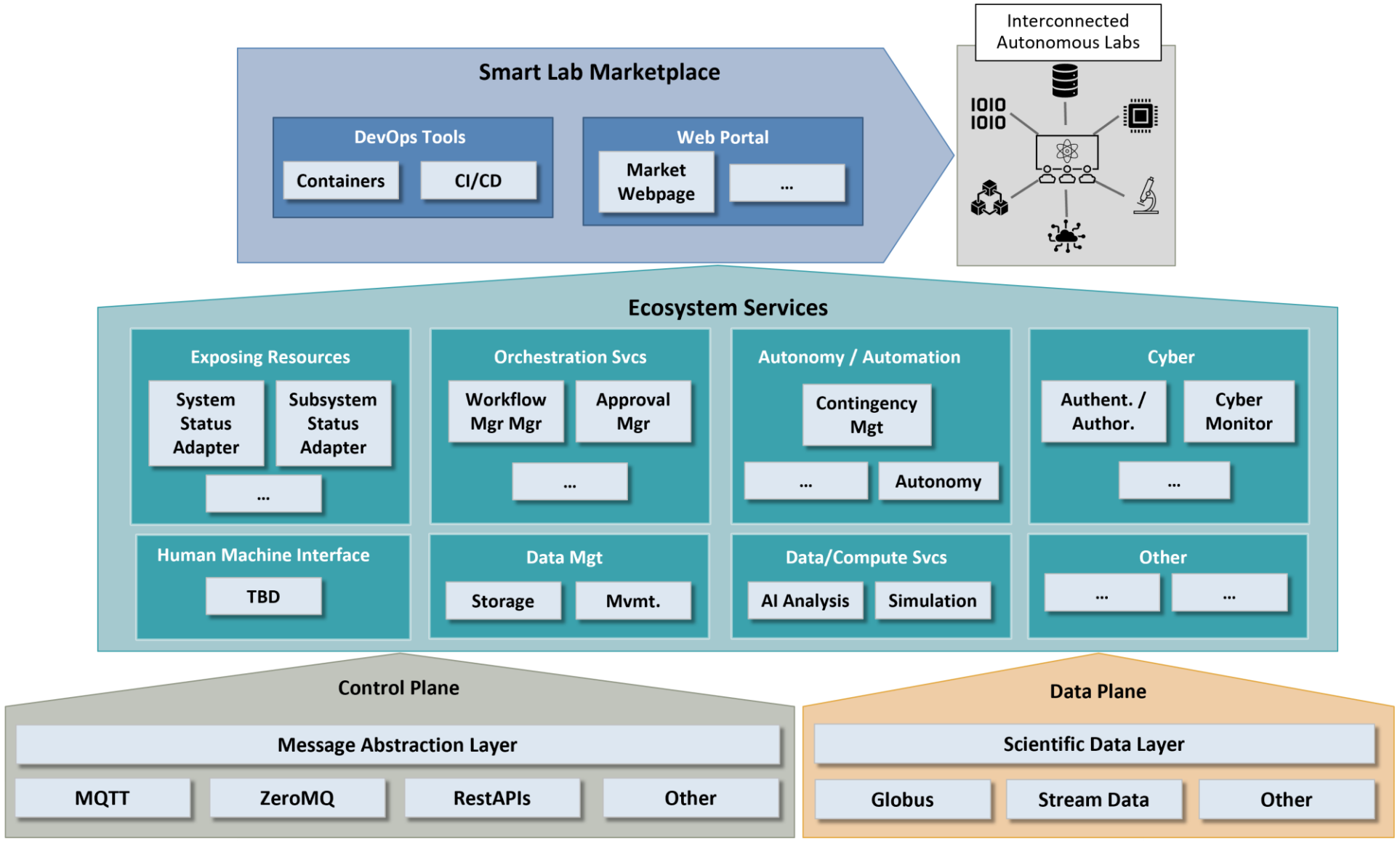


Common Ecosystem is Required for Interoperability

OAK RIDGE National Laboratory

INTERSECT
INTERCONNECTED SCIENCE ECOSYSTEM

Interconnected Science Ecosystem



Thank You!!