RULES FEST

Rule-Based Automatic Management of a Distributed Simulation Environment

Ronald Bowers

Staff Scientist

US Army Research Laboratory

Agenda

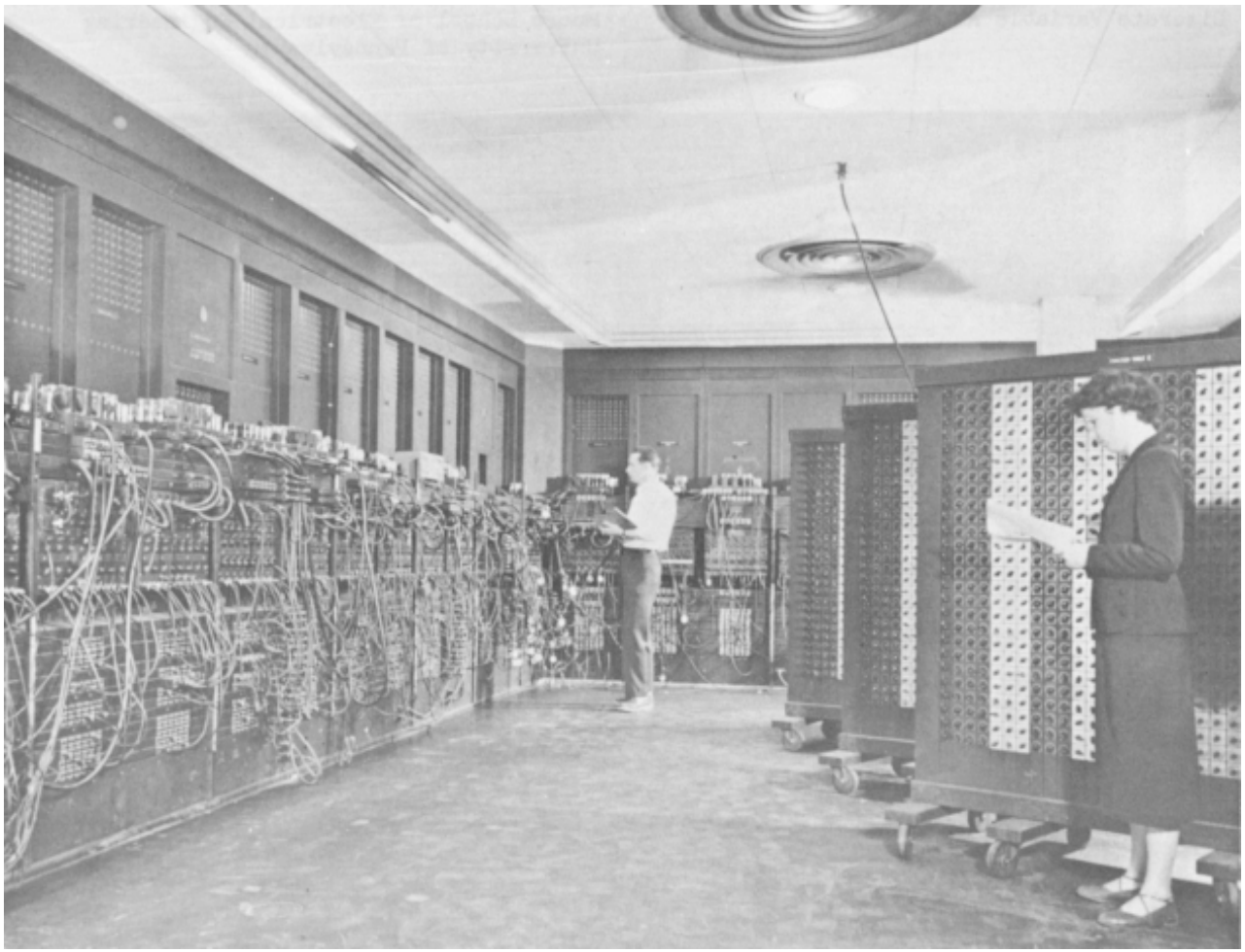
- Who we are and what we do
- Overview of the MUVES 3 architecture
- Our system management problem
- Details of our solution
- Status, issues and future work

The Army Research Laboratory

- The Army Research Laboratory (ARL) is the Army's corporate basic and applied research laboratory. Our mission is to provide innovative science, technology, and analysis to enable full-spectrum operations.

- I represent the Survivability/ Lethality Analysis Directorate (SLAD) of ARL.

We have some experience with computers.



TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

SLAD Mission

Ensure that US personnel and equipment…

…survive and function effectively in hostile circumstances.

Ballistic Threats

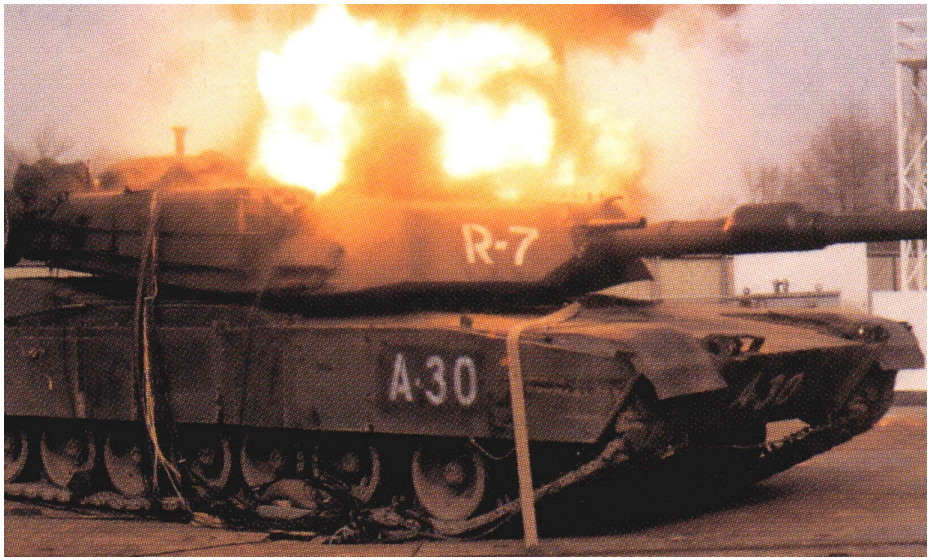
Nuclear, Biological and Chemical Attack

Electronic Warfare

Information Warfare

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

SLAD performs both experimentation and modeling.



TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

SLAD Ballistic Vulnerability/Lethality (V/L) Modeling

- SLAD's primary tool for performing ballistic V/L analysis is MUVES.
- MUVES development began in 1984.
- The current version (2.x) is a single-threaded C application.
- We are currently developing MUVES 3, which is an all-new replacement system.

MUVES 3

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

Program Focus

- Provide the next generation of simulation system for the V/L analyst community
  - Mostly Java.
  - Dynamic distributed and service-oriented.
  - Will support over 100 concurrent users.
  - Incorporates a computational grid, parallelized system that distributes tasks and computes results that are graphically displayed.
  - Will operate in both “batch” and interactive modes.

MUVES

TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

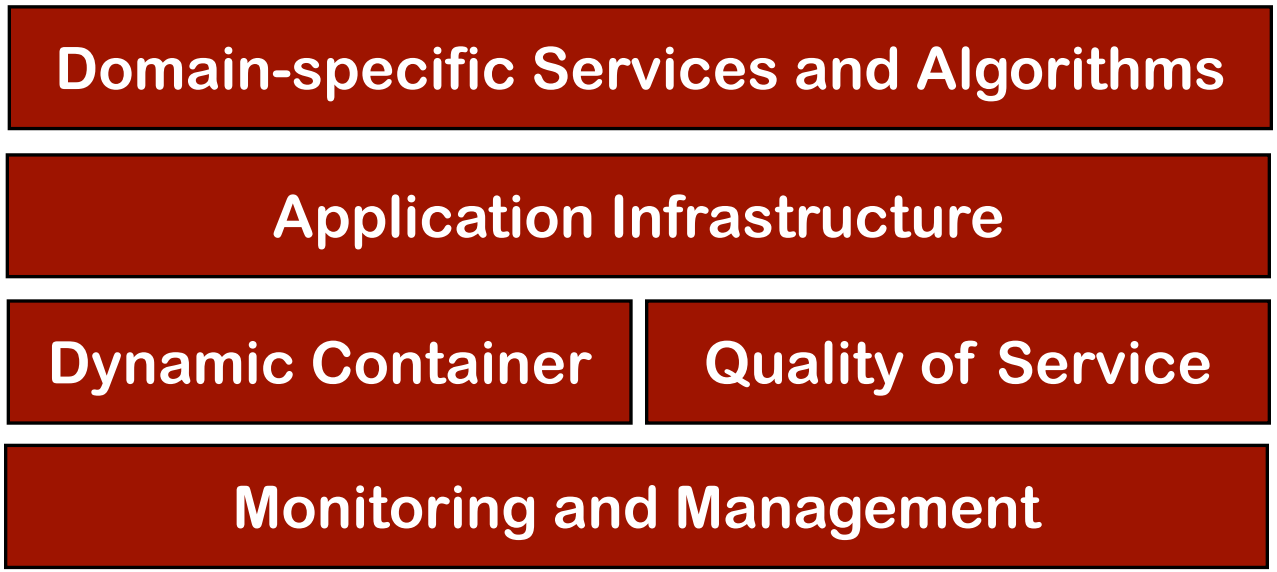
Interesting Challenges

- We have few servers, but many powerful workstations.
  - Architecture must exploit analyst community machines
  - Share CPU, memory and disk
  - Heterogeneous deployment environment
- Required application assets vary
  - Need real-time provisioning of application assets
  - Must be able to route functionality to machines that are best capable of executing tasks/ functions
  - Must be able to scale on demand based on real time need and use of the system

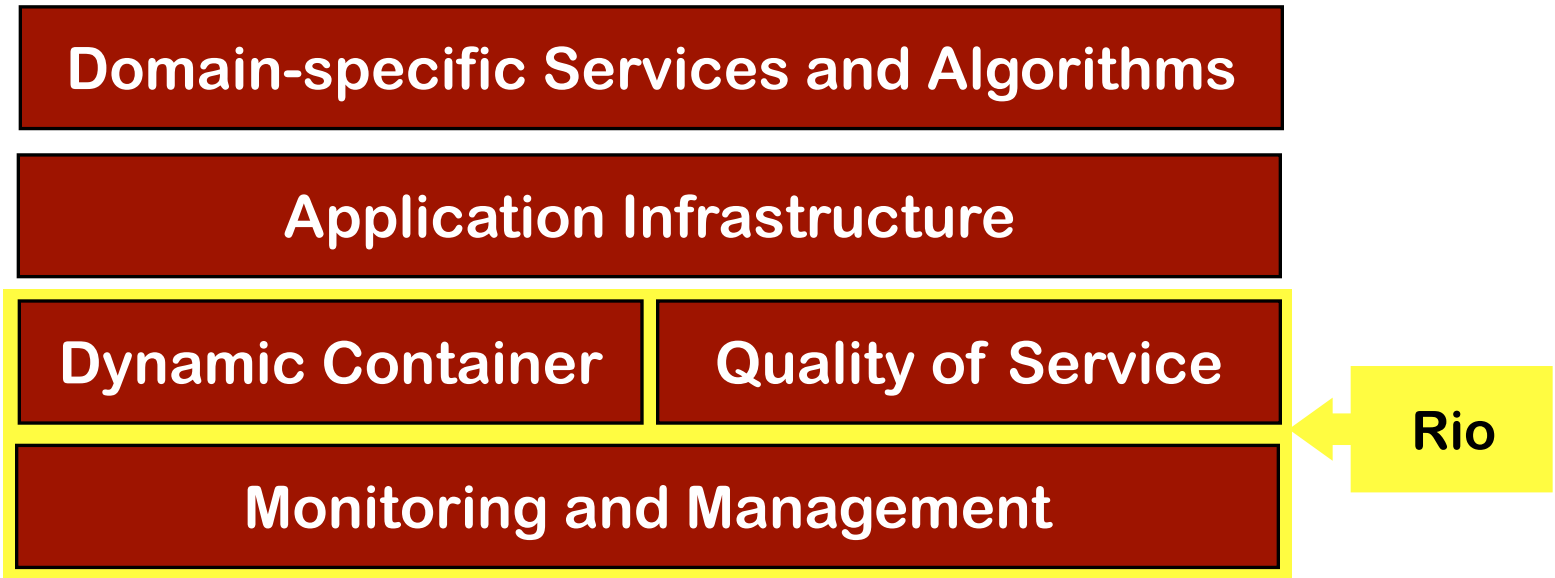
System Architecture Approach

- Choose technology that embraces dynamic distributed capabilities
- Craft a loosely coupled service oriented architecture that segments the system into functional roles
- Choose persistence technologies and approaches that allow for low latency and high concurrency
- Keep Disk I/O out of the main stream processing
- Track data as it moves through the system
  - In-flight (hot in-memory), On-Disk, Archived

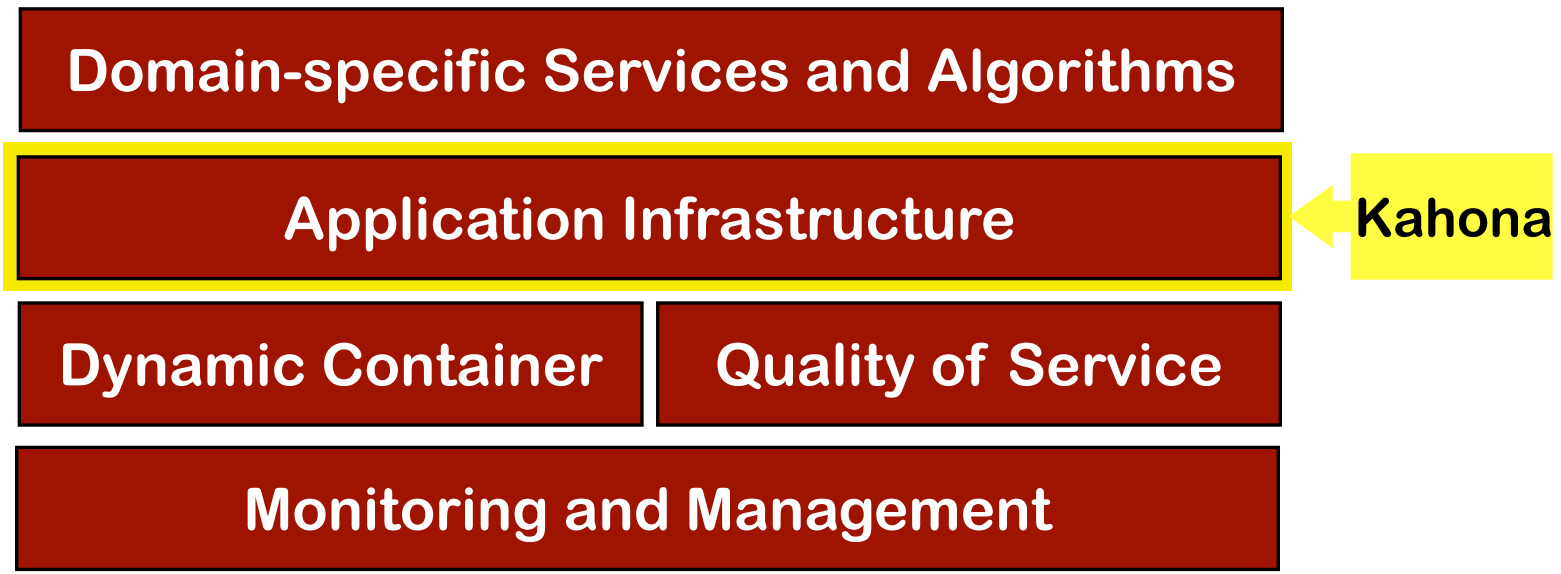
What's Underneath



What's Underneath



What's Underneath



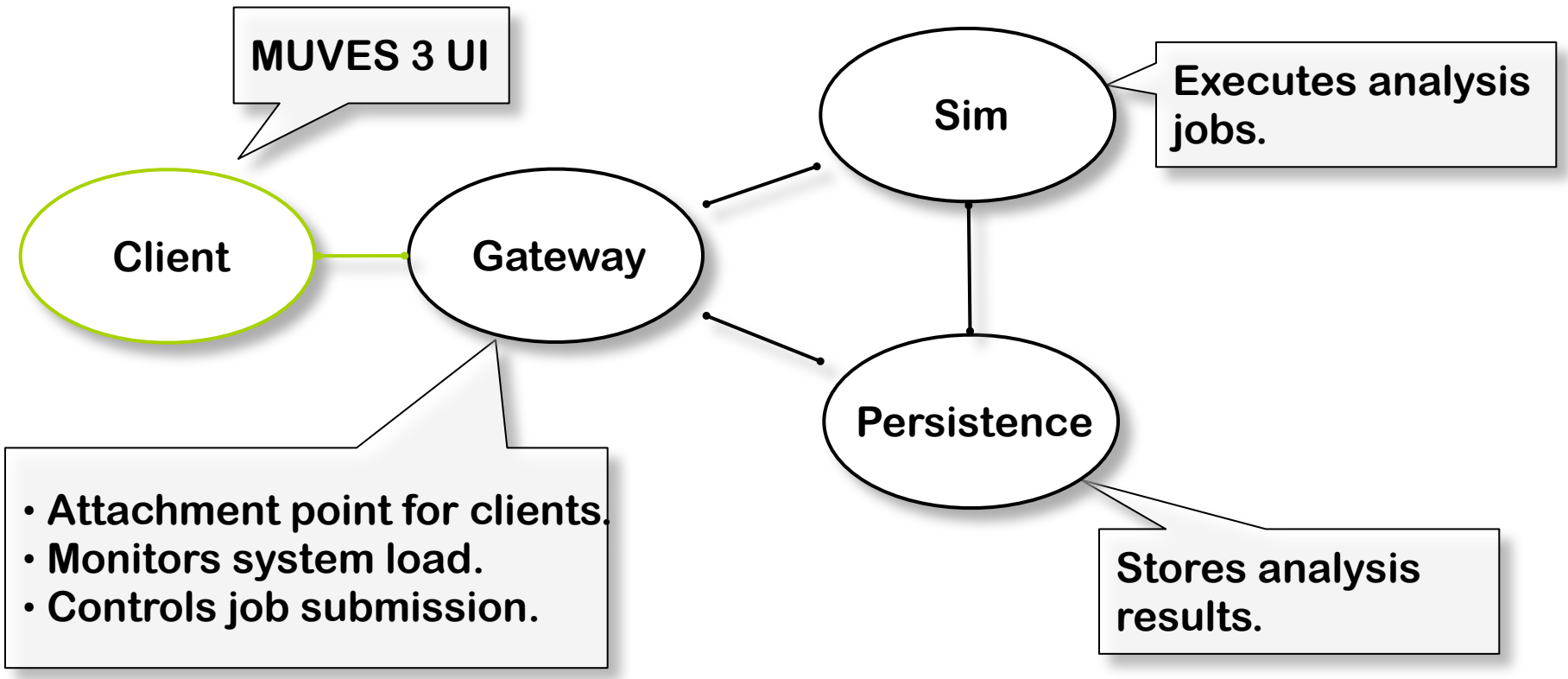
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

Kahona

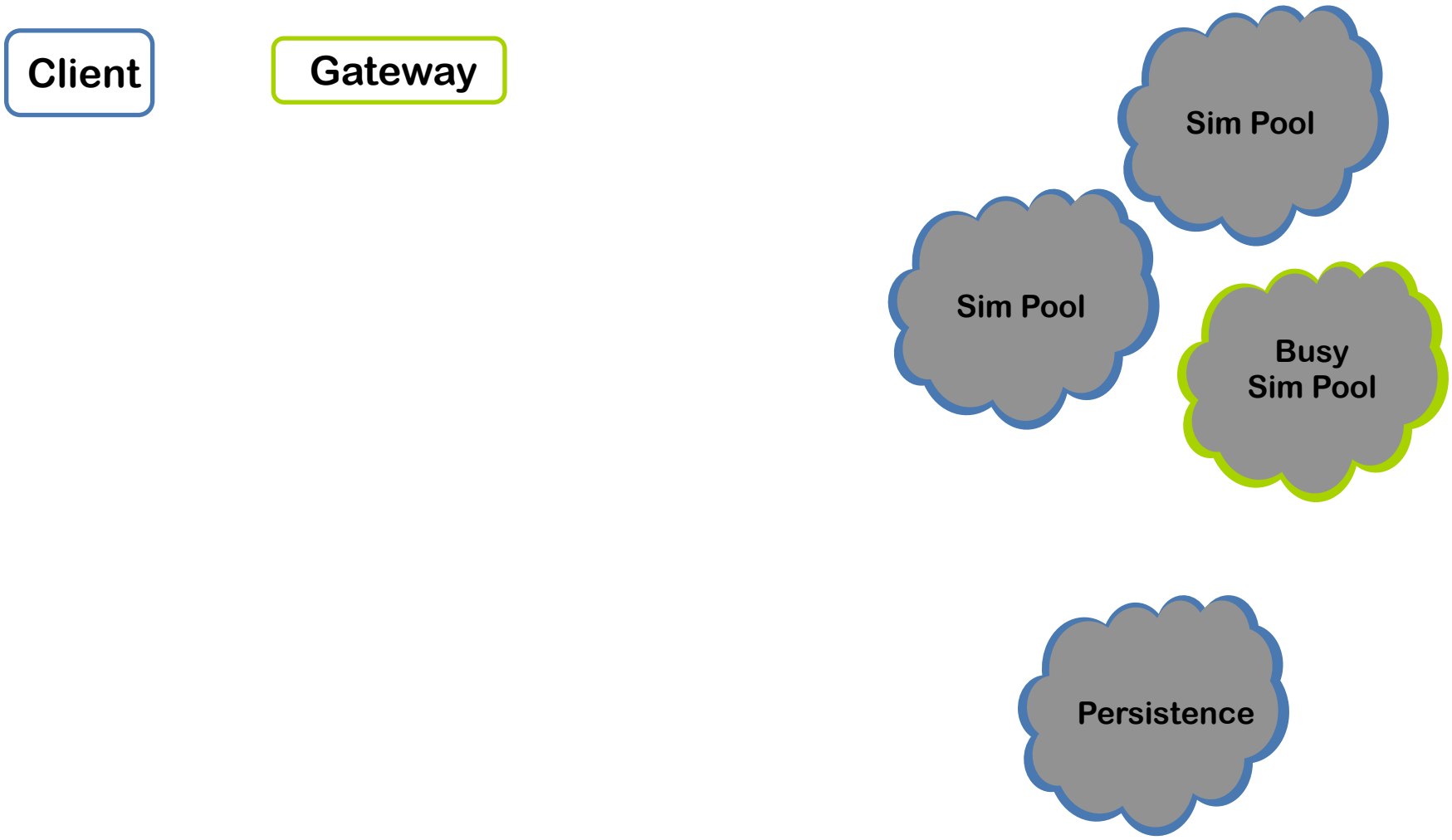
- Was established as a prototype for MUVES 3 architectural enhancements.
- Now forms the basis of the MUVES 3 service-oriented architecture (SOA).
- Includes all of the non-sensitive services used by MUVES 3.
  - Task execution services
  - Input and result persistence
  - Geometry interrogation
- Is an open source project (LGPLv2.1) on Java.net.

What's Underneath

MUVES 3 System Organization

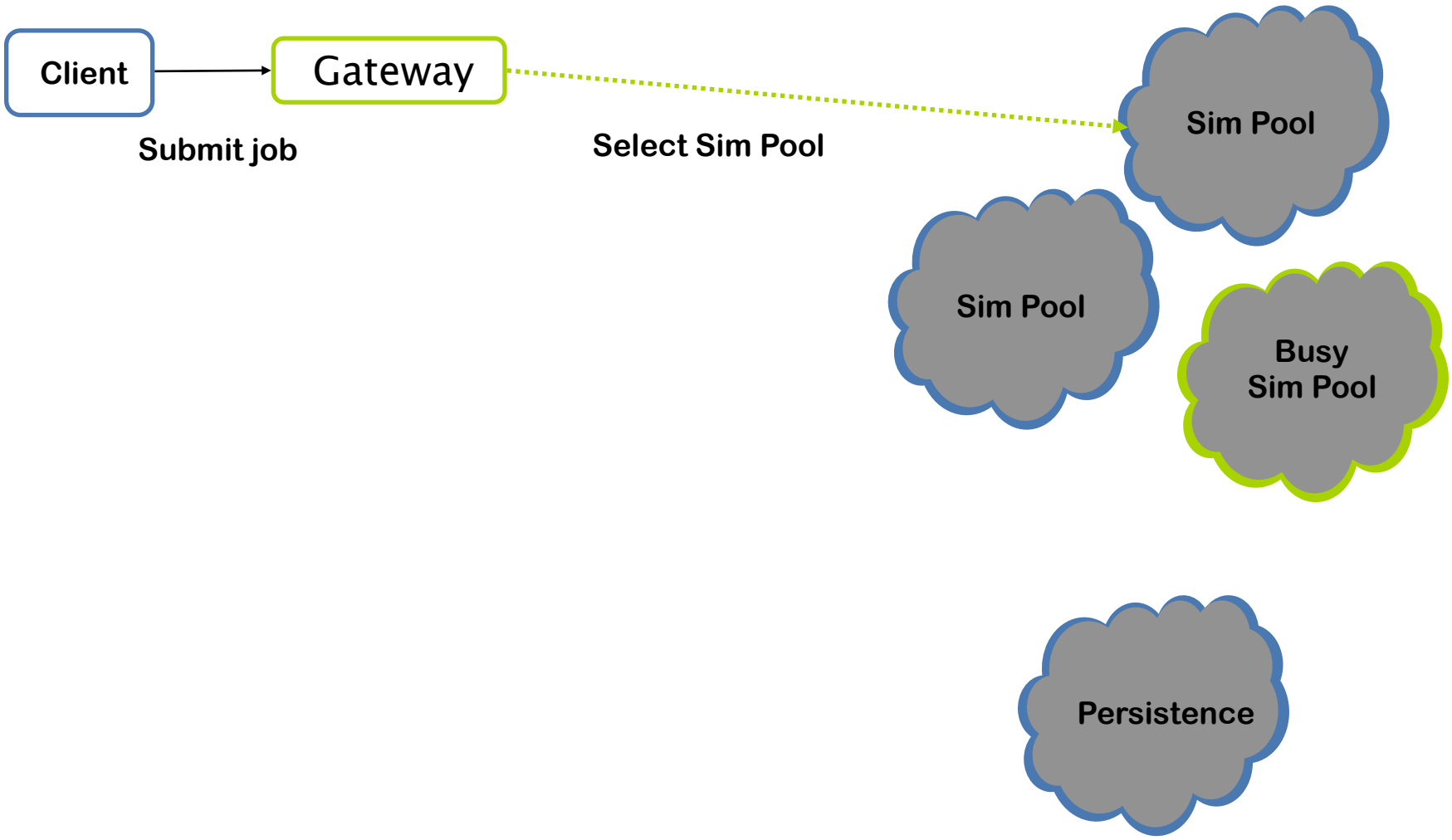


MUVES 3 Execution



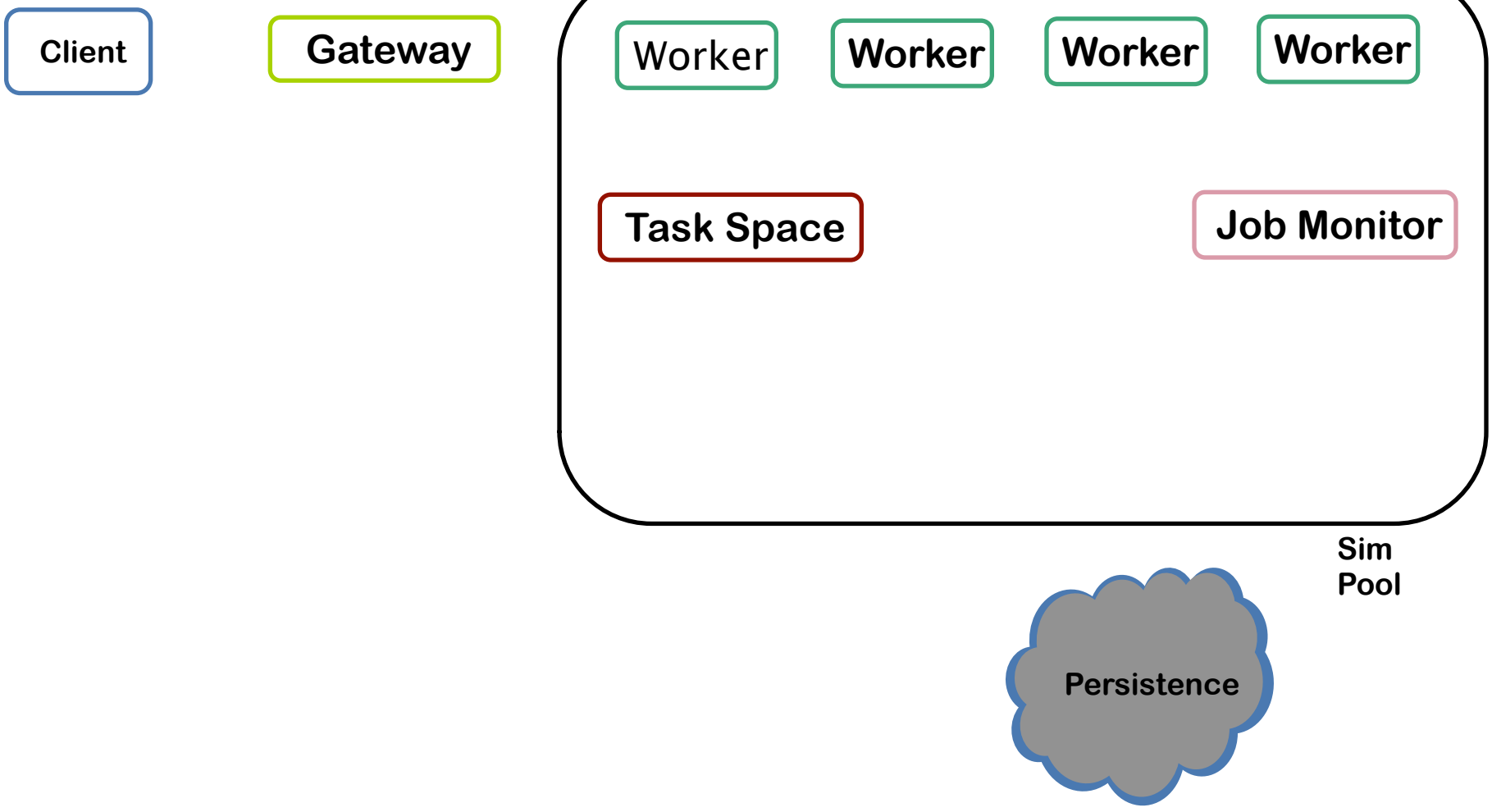
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

MUVES 3 Execution



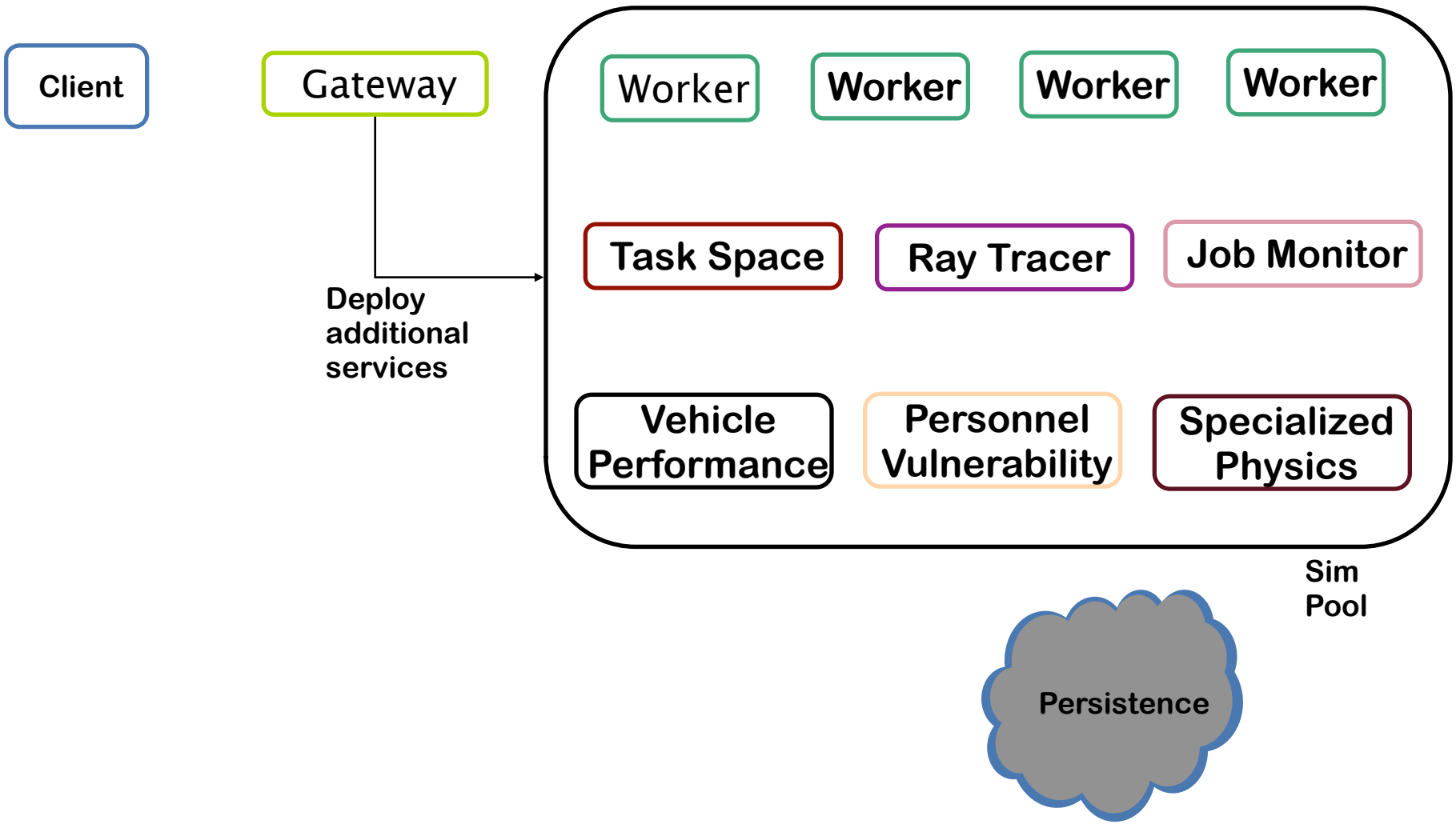
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

MUVES 3 Execution



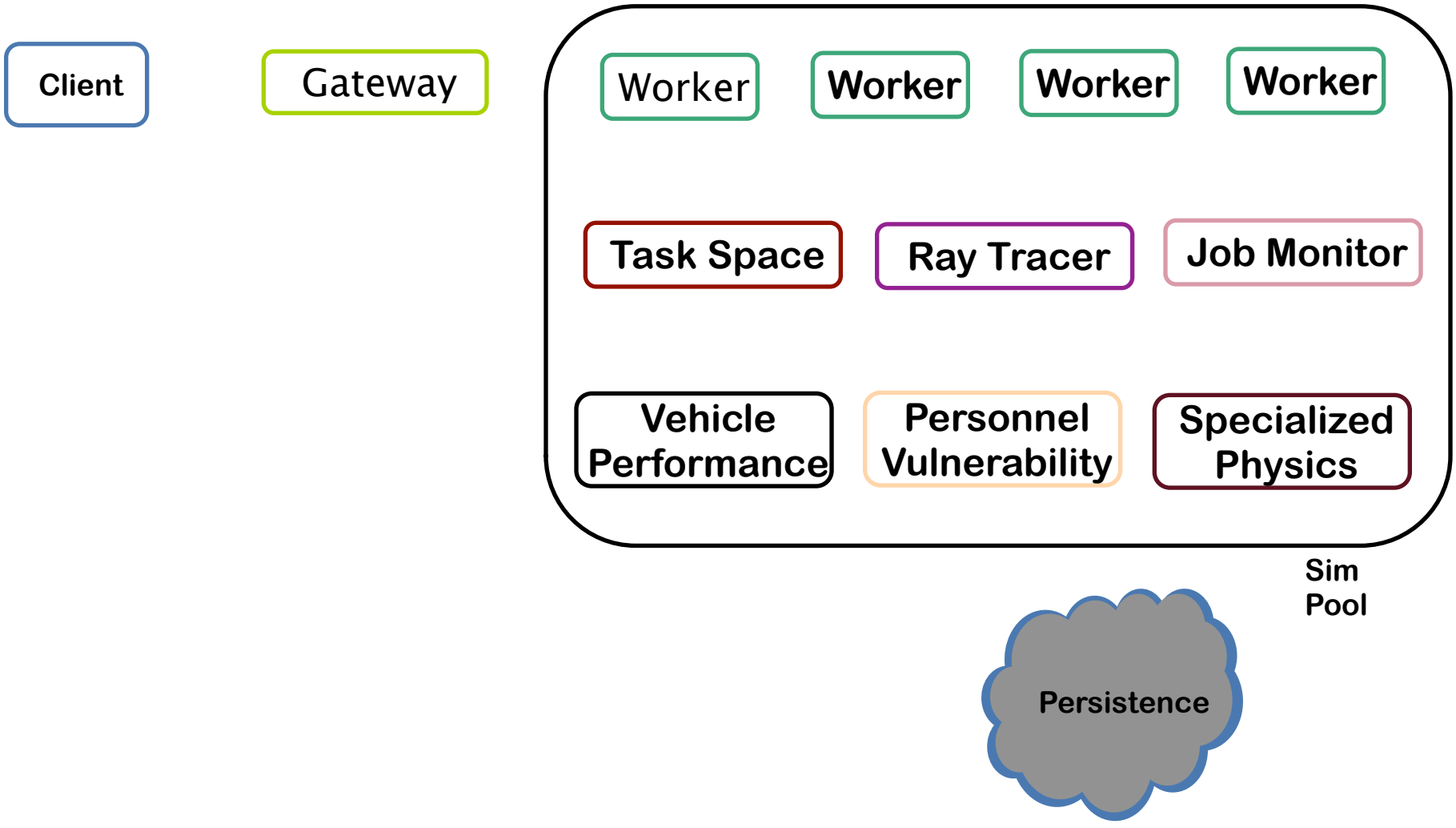
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

MUVES 3 Execution



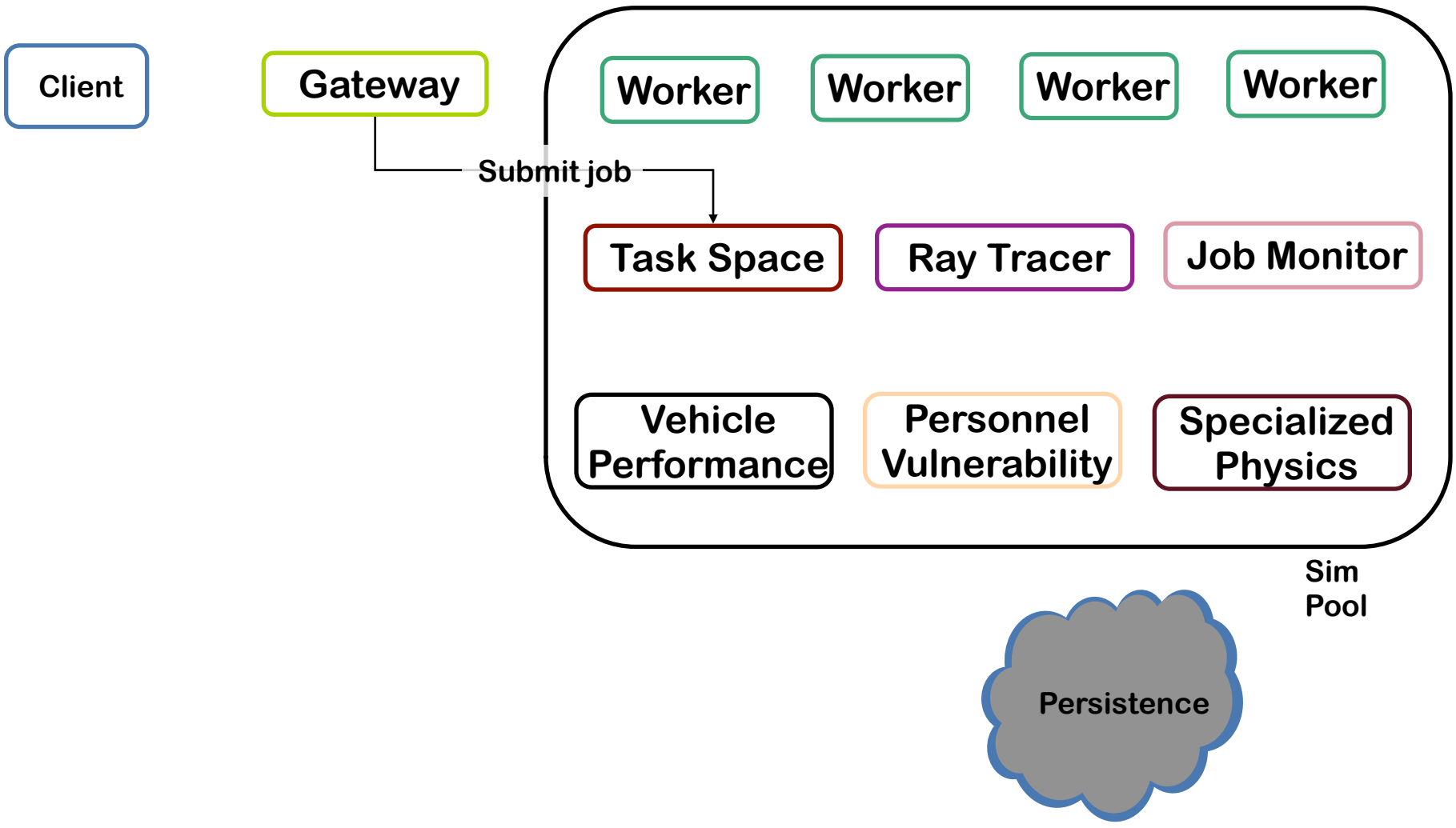
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

MUVES 3 Execution



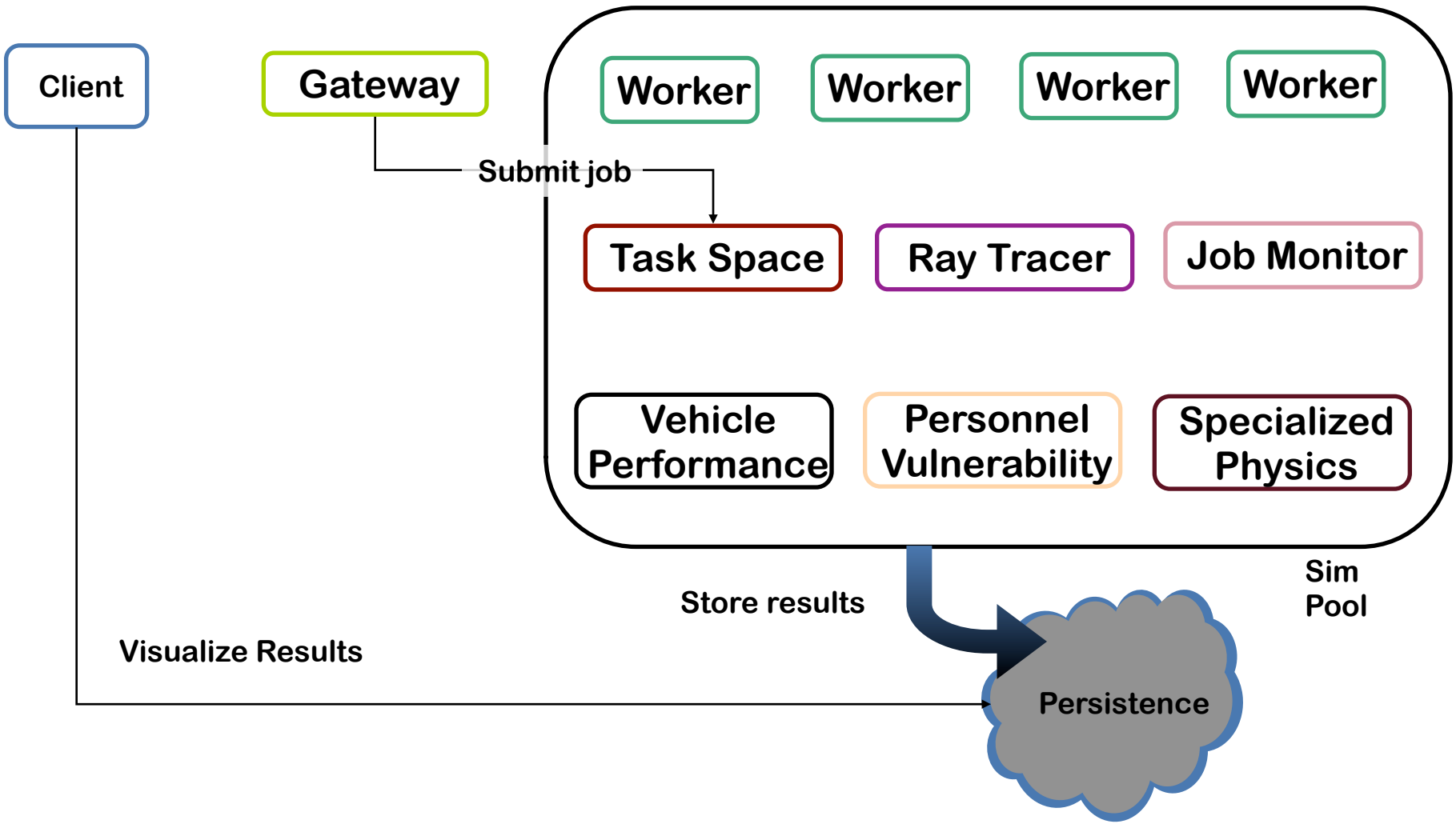
TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

MUVES 3 Execution



TECHNOLOGY DRIVEN. WARFIGHTER FOCUSED.

MUVES 3 Execution



Our System Management Problem

- The system is complicated.
- It must be managed 24/7, but we have no resources for human oversight.
- It must react to changing demands, changing resources, and failures.
- It has dynamically changing optimization goals:
  - best effort on all analyses, or
  - all resources to an emergency analysis.

Our Solution Approach

Apply a rules-based system to dynamically configure the system

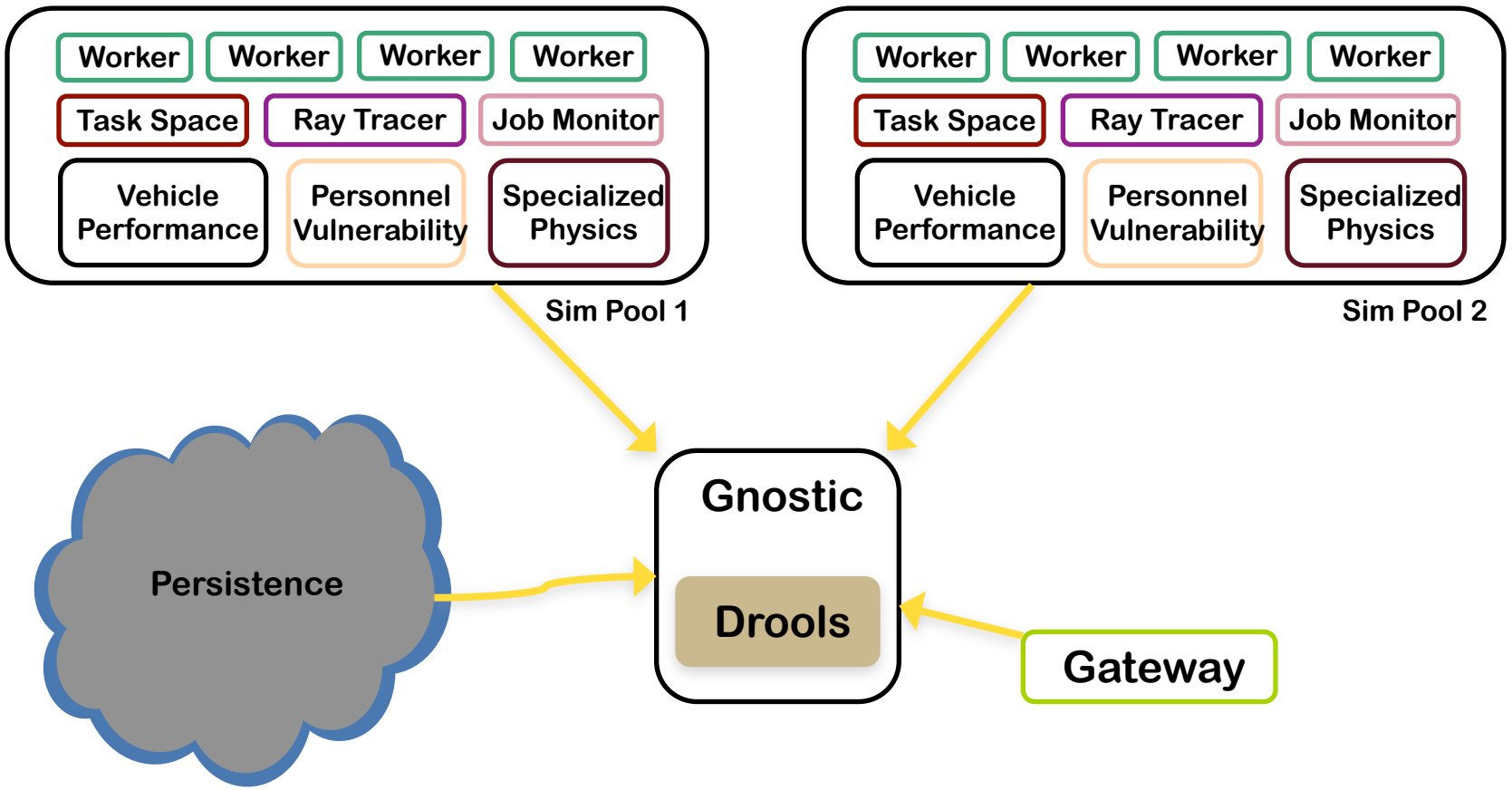
- Collect telemetry from the services
- Apply rules against the telemetry
- Issue management commands to the services

Key Technologies

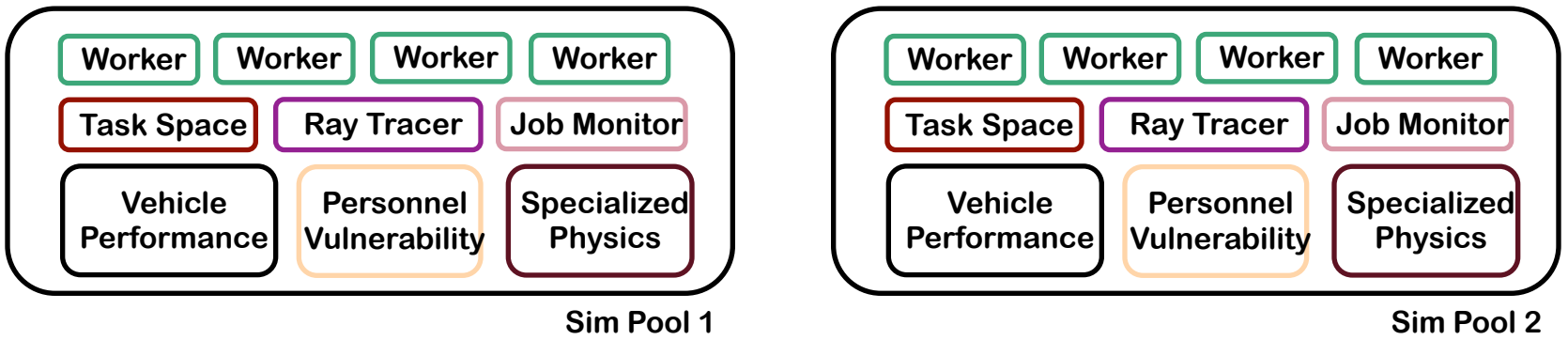
- Drools Fusion
  - performs time-based complex event processing and rule evaluation.
- Rio
  - provides the telemetry framework.
  - provides a service (“Gnostic") to host the rules engine, receive the telemetry, and issue the commands.

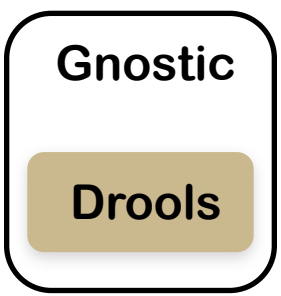
✳ Many thanks to Mauricio Salatino (Drools) and Dennis Reedy (Rio) for their work making this possible.

Telemetry comes to Gnostic from the various services



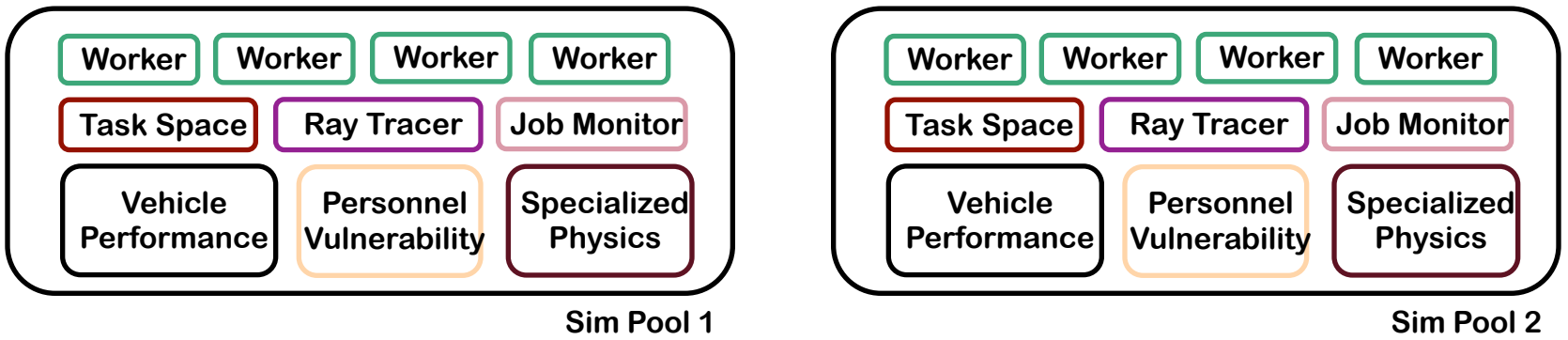
Gnostic uses Drools to evaluate the telemetry and make decisions.

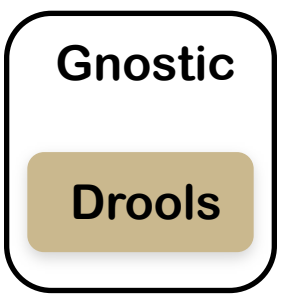




Gateway

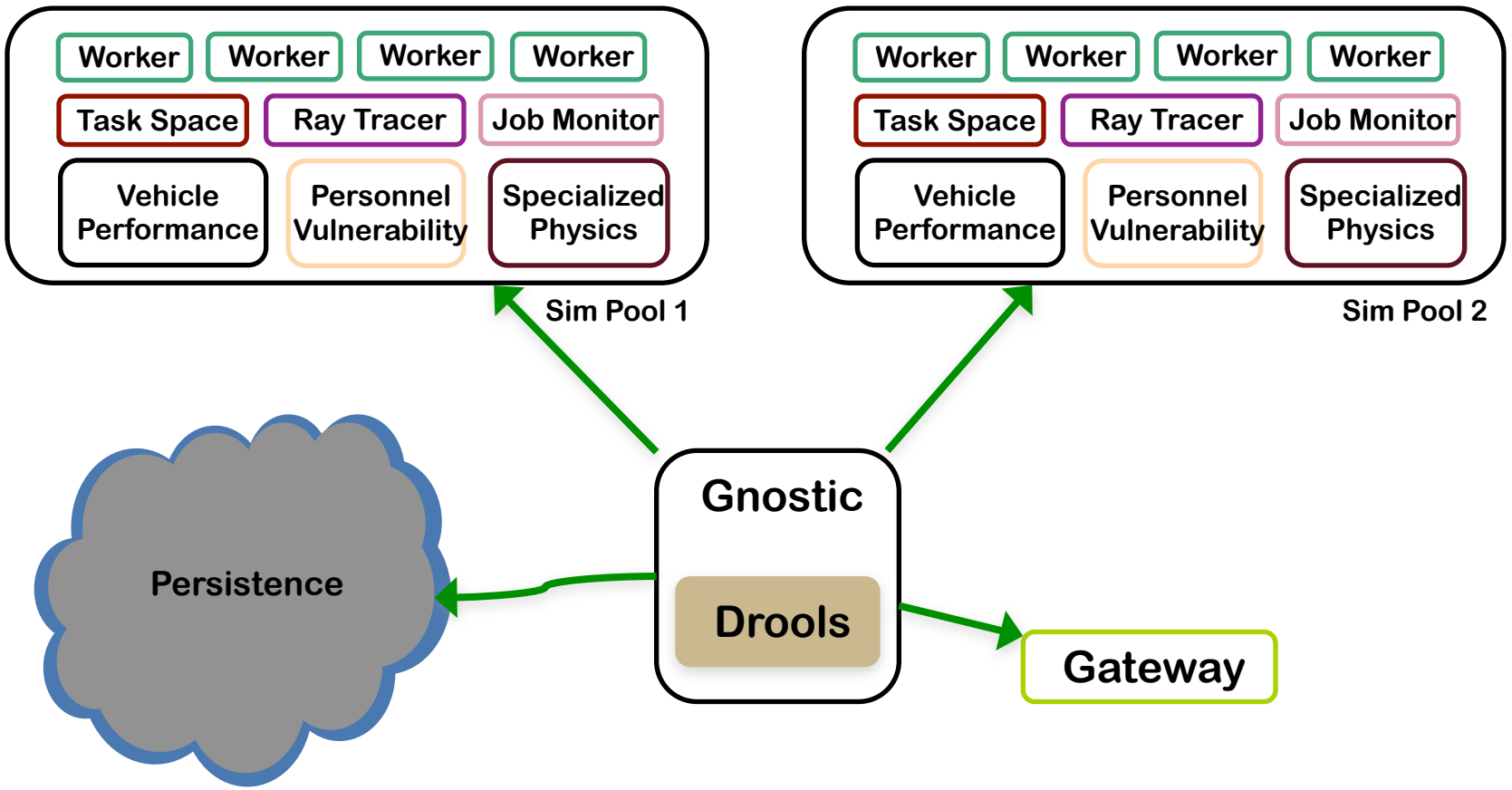
Gnostic uses Drools to evaluate the telemetry and make decisions.





Gateway

Administrative controls are invoked on the services.



Types of Administrative Actions

- Increase or decrease the number of instances of a deployed service.
- Pause/resume starting new jobs.
- Relocate result data.
- Relocate a service.

Types of Administrative Actions

- Increase or decrease the number of instances of a deployed service.
- Pause/resume starting new jobs.
- Relocate result data.
- Relocate a service.

MUVES 3 Result Storage

- MUVES 3 results consist of directed graphs that vary in size from tens to tens of millions of nodes.
  - Nodes can vary widely in size.
- The MUVES 3 persistence “meta-service” consists of two layers, both of which are network-distributed.
  - In-memory cache based on JavaSpaces
  - On-disk long-term storage
- The result data lifecycle:
  1. Simulation writes results to the cache.
  2. Results are replicated to disk when the job is complete.
  3. Results are evicted from the cache when space is required.

Gnostic configuration

```
deployment(name: 'System') {
   // configuration for Gateway service
   rules {
        rule {
            resource 'Muves3.drl'
            ruleClassPath "mil.muves.deployment:rule:${m3_version()}"
            serviceFeed(name: "Result Space") {
                watches "${SystemWatchID.PROC_CPU}, ${SystemWatchID.JVM_MEMORY}"
            }
        }
    }
}
```

Persistence Management Rule

```
package net.kahona.system;
import …
global org.rioproject.gnostic.DeployedServiceContext context;

declare CalculableMemory
    @role(event)
    @timestamp(date)
end

rule "MemoryRule"
when
    $mem : Number(doubleValue > 0.50)
              from accumulate(CalculableMemory($value : value) over window:time(1m)
              from entry-point "calculables-stream", average($value))
then
    List<JobStorageManager> managers = context.getServices("Job Storage Manager",
    "Persistence", JobStorageManager.class);
    System.err.println(new Date(System.currentTimeMillis()) + ": Memory is at " + $mem );
    for (JobStorageManager jsm : managers) {
        System.err.println("Invoking job storage manager: " + jsm.toString());
        jsm.triggerSpaceTransfer(256*1024*1024);
    }
end
```

Status, Issues and Future Work

- **Status**
  - Infrastructure is operational. Gnostic is included in Rio 4.0
  - The persistence management rule is included in the current MUVES 3 release, additional rules are under development.
- **Issues**
  - Rules are not compiled until Gnostic is deployed thus syntax errors are not discovered until runtime.
  - Designing good rules is hard and realistic testing is harder.
- **Future Work**
  - Convert all other MUVES 3 service level agreements to be rule-based.
  - Develop a rule compiler. Will be part of the Maven plugin for Rio.
  - Begin full-scale system testing and start refining our rules.

Thank You!

Ron Bowers

ronald.a.bowers2.civ@mail.mil