Simulation-Based Testing (SBT) for DoD Software

Jeremy Loomis, Alex Matthews
NextGen Federal Systems

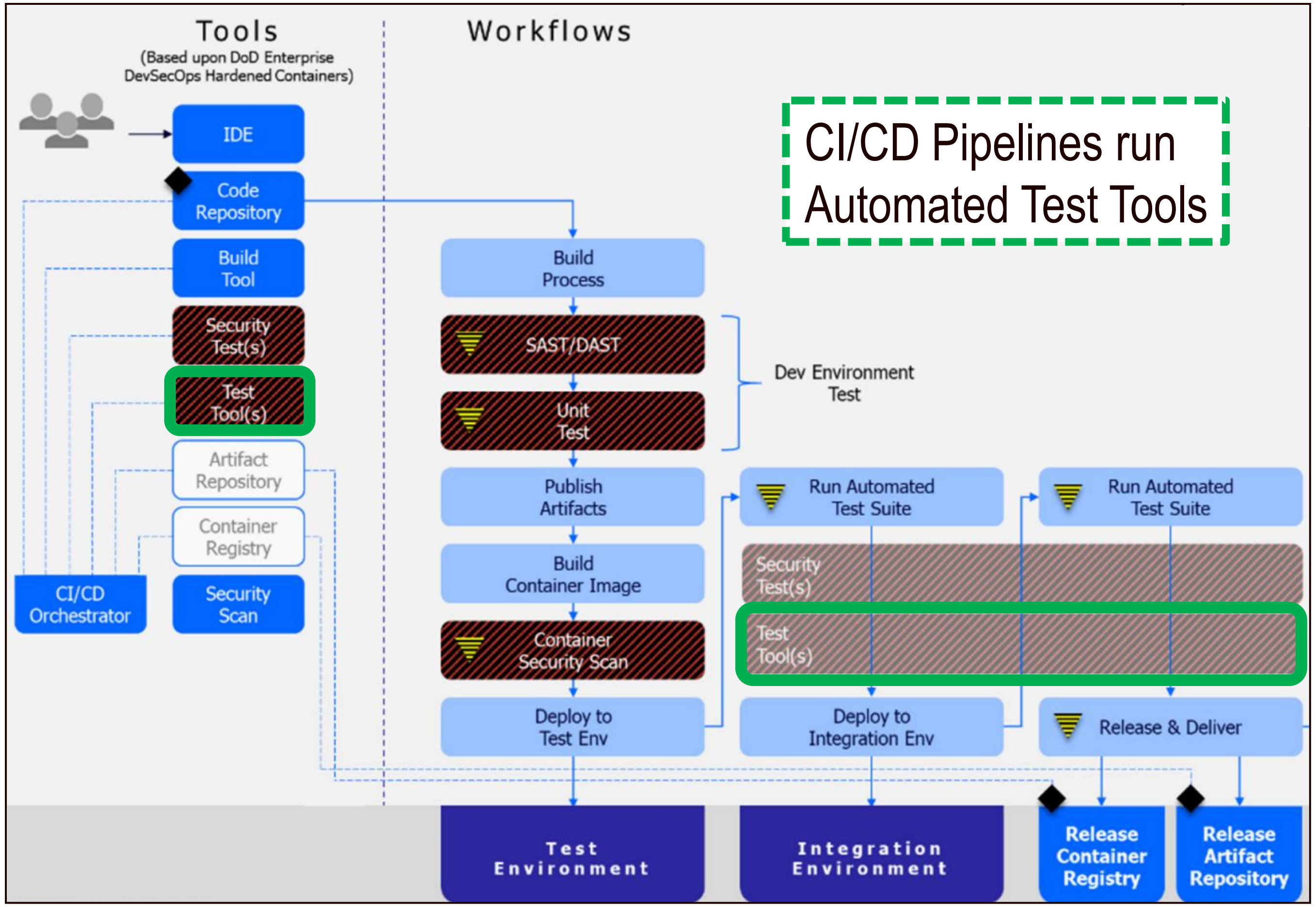
Introduction

- Software testing is changing to include more automated testing
- With many types of software (such as military applications), a challenge for automated testing is generating synthetic data and activity
- Modeling and simulation (M&S) can help
  - Provides representative test data from synthetic actors in a synthetic environment

- The approach leverages current M&S technologies to implement a **Simulation-Based Testing (SBT)** toolset
  - Injects synthetic test data into a Continuous Integration & Deployment (CI/CD) pipeline
  - Combines M&S-as-a-Service (MSaaS) services with Automation & Orchestration

Background: Automated Testing

- **Test Tools** (green outlines) are part of a CI/CD toolchain
- An instance of the **System Under Test (SUT)** is created during a pipeline run
- **Automated Test Suite** is run on the SUT before release



DoD Enterprise DevSecOps Reference Design: CNCF Kubernetes.
March 2021, Version 2.0

Background: Unique Testing for Military Apps

- Military applications are complex systems that process unique datasets
- Require extensive testing so that the software systems can be trusted by warfighters and analysts
- Software testing is needed at multiple architectural levels:
  - System, Subsystem, Component, Service, and Application levels

Example Application

- Notional all-source intelligence production system
  - Provides data ingestion, content extraction, normalization & correlation
- Includes an "analytics service" that performs initial content extraction
- Testing goal: evaluate the service for performance and regression errors
- Need representative "intel feeds"

Background: M&S-as-a-Service (MSaaS)

"[MSaaS is] a new concept that includes

- **service orientation**
- **provision of M&S applications via the as-a-service model of cloud computing**

to enable more composable simulation environments that can be deployed and executed on-demand."

STO/NATO (2019). *MSaaS Technical Reference Architecture*

MSaaS approach simplifies M&S deployment

- M&S packages require complex IT configuration
- MSaaS wraps them into consumable services

M&S Enabling Services
M&S Integration Services
• M&S Mediation Services
• M&S Message-Oriented Middleware Services
M&S Composition Services
Simulation Control Services
Simulation Scenario Services
M&S Information Services
• M&S Repository Services
• M&S Registry Services
M&S Services
Simulation Services
Modeling Services
Composed Simulation Services

Simulation-Based Testing (SBT) Approach

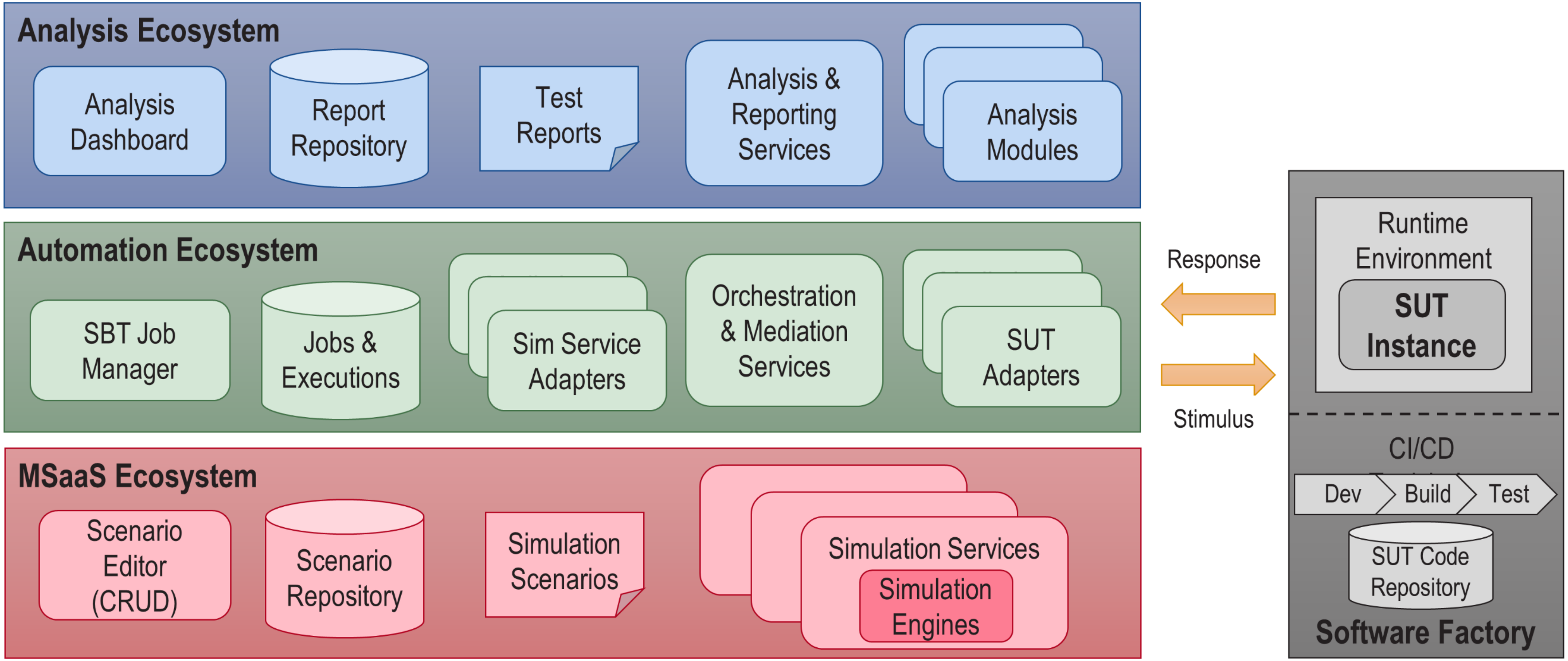
- Build on current COTS/GOTS M&S tools
- Run in a constructive-simulation mode
  - High-fidelity synthetic environments
  - Realistic synthetic systems
  - Intelligent synthetic actors
- Tools wrapped in M&S Services and orchestrated without user intervention
- Representative test data provided to the SUT using existing interfaces/protocols.

Result: SBT integrated into CI/CD process as another 'automated test tool'

Key attributes of an SBT system:

- ***Cloud-Native***: Scalable deployment via Kubernetes
- ***Modular***: Works with any SUT & Software Factory
- ***Flexible***: Users author scenarios, jobs, pipelines
- ***Continuous***: Tied into CI/CD process
- ***Automated***: No 'man-in-the-loop' steps required
- ***Cross-Functional***: M&S and testing work together
- ***Interoperable:*** Models integrate with one another in larger scenarios, disparate tools/simulations communicating with one another

Building an SBT Toolset



MSaaS Ecosystem

- **Scenario Editors** are used to author relevant simulation scenarios
- **Scenario Repository** stores and catalogs versioned scenarios
  - Provides CRUD (Create, Retrieve, Update, Delete) functions
- Wraps existing Sim Engines (STK, AFSIM, NGTS, OneSAF) into **Simulation Service** microservices
  - Web API for simulation control, clock management, and data input/output

REST API for Simulation Services defined via OpenAPI (formerly SwaggerUI) specification

Automation and Analysis Ecosystems

Automation Ecosystem

- Centers on the SBT Job Manager
  - Coordinates execution of SBT jobs
  - Connects Sim Service Adapters (that control Sim Services) to SUT Adapters
  - Includes example 'SBT job templates' that can be tailored as needed
- **Orchestration & Mediation Services**
  - Stimulation (injecting data into SUT)
  - Collection (retrieving data from SUT)

Analysis Ecosystem

- **Analysis & Reporting Services**
  - Analysis Modules compute metrics such as accuracy and performance
  - Test Reports are created based on results from an SBT job
- Allows plug-ins to provide 3rd-party analytics for different forms of V&V
- Resulting reports and metrics are
  - Collected in a database
  - Presented in an intuitive dashboard

CI/CD Integration

- SBT Job Manager and Analysis Services can be controlled from any build orchestrator (e.g., Jenkins) using a REST API
- CI/CD pipelines can include customized SBT steps/stages that run as part of overall pipeline execution
- For example:
  - Scan SUT Source Code
  - Run SUT Unit Tests
  - Compile & Package SUT
  - Deploy SUT to Test Server
  - Launch SBT Job against SUT
  - Collect SBT Test Results

Checkout Code	Build Python Module	Run SAST Tests	Build Docker Image	Deploy to Test Environment	Run DAST Tests	Run SBT Job	Un-Deploy from Test Environment
988ms	1s	2s	2s	2s	1s	7s	2s
599ms	917ms	1s	2s	3s	2s	6s	4s

Notional Jenkins CI/CD pipeline with an SBT stage

IMAGRS Case Study

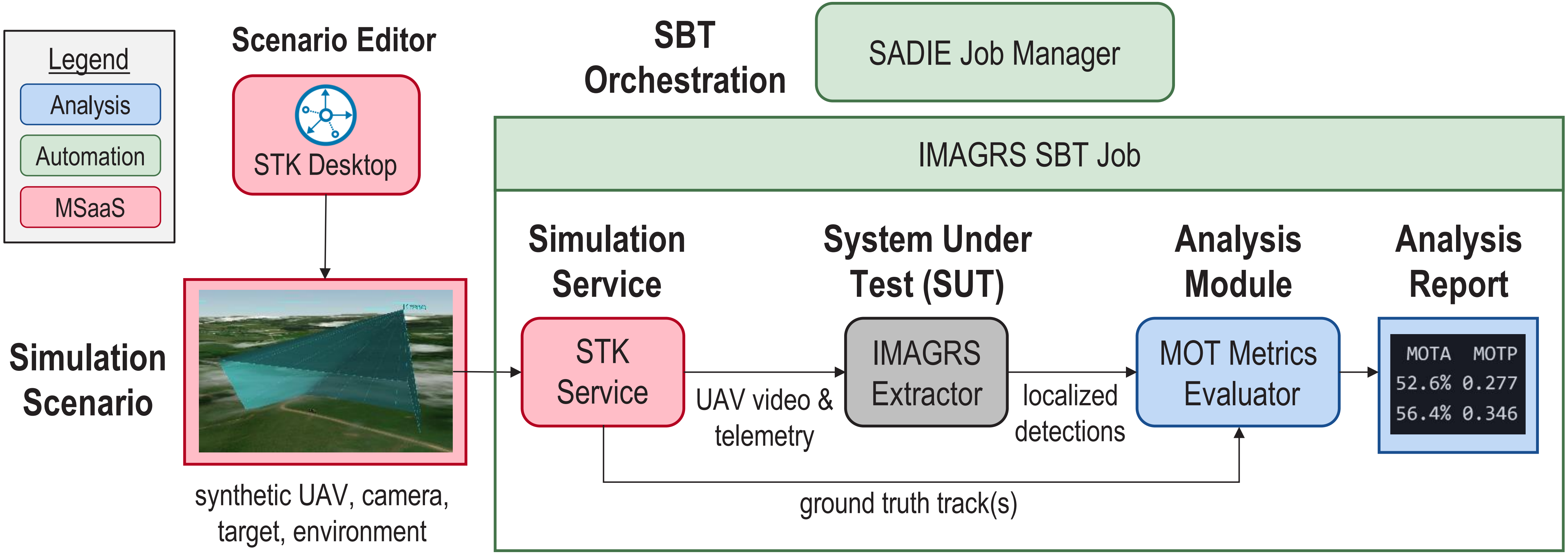
SUT Description

- *Intelligent Multirotor Autonomous Ground Relocatable Sensor (IMAGRS)*
  - Small UAS for persistent surveillance
  - Includes an integrated EO/IR payload
  - Onboard Computer Vision (CV) algorithms
- *"IMAGRS Extractor" module*
  - Processes georeferenced video stream to detect and localize targets of interest
  - Inputs: video and telemetry
  - Output: timestamped target detections

Desired Testing

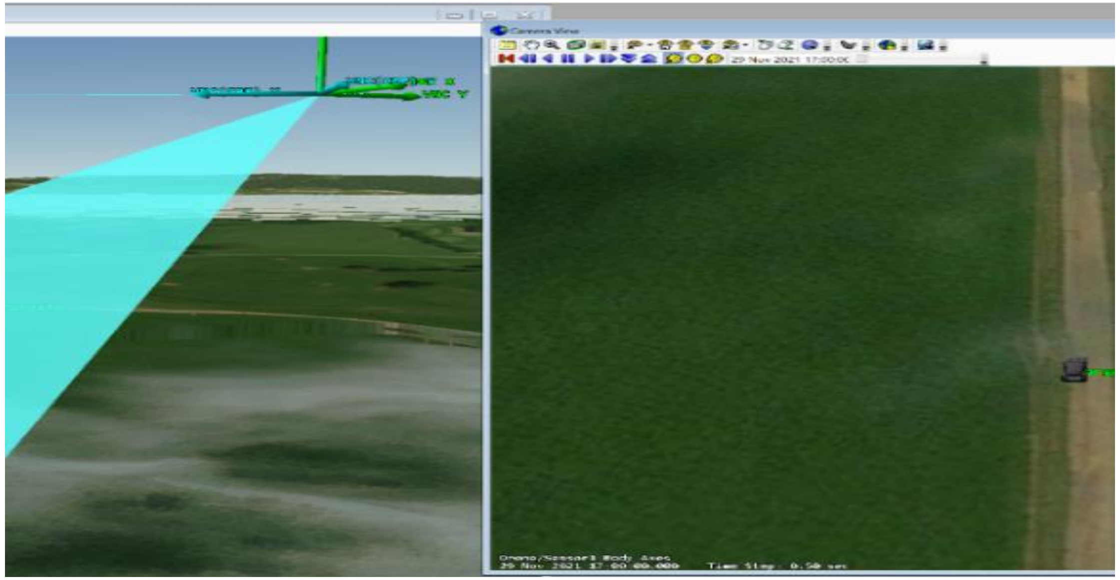
- Use SBT to test the Extractor module **using synthetic video data**
  - Reduces the need to perform full integration testing with the platform
- Primary desired testing: functional testing to measure 'accuracy'
  - Determine how well the Extractor can correctly detect and locate targets
- Alternate metric would be to measure performance/latency

SBT Workflow



An end-to-end SBT workflow for testing the IMAGRS Extractor was proven

Simulation Scenario



- Synthetic environment
  - Created in Systems Tool Kit (STK)
  - Using terrain data and satellite imagery
- Two moving entities simulated
  - Synthetic model of the UAV with FOV
  - Synthetic model of target vehicle
- Used 3D graphics for synthetic video
  - Virtual 'camera view' tethered to boresight of the virtual electro-optical (EO) camera
  - During animation, window used to collect a series of synthetic video frames
- STK report used to capture telemetry for both entities
  - UAV system (position and camera orientation) and target vehicle ('ground truth' data)

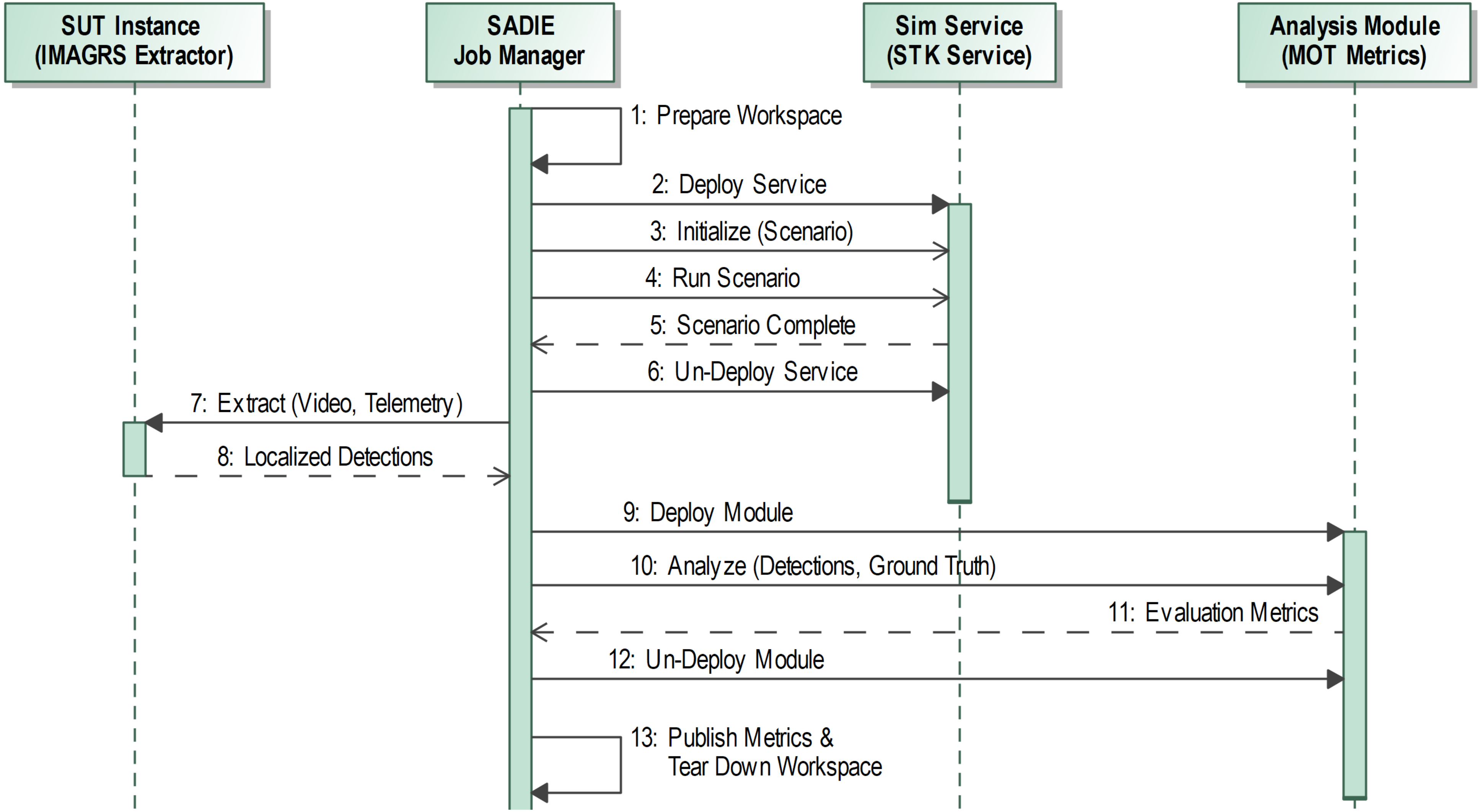
Analysis Module

- Used py-motmetrics (Heindl, 2022)
  - Python library for benchmarking multiple object trackers (MOT)
  - Measures performance using CLEAR-MOT metrics and ID metrics
- Provides appropriate SBT detection metrics for comparison between:
  - 'Ground truth' data from synthetic target vehicle track
  - Localized detections produced by the IMAGRS Extractor

Metrics for object detection and tracking

Name	Description
num_frames	Total number of frames.
num_matches	Total number matches.
num_switches	Total number of track switches.
num_false_positives	Total number of false positives (false-alarms).
num_misses	Total number of misses.
num_detections	Total number of detected objects including matches & switches.
num_objects	Total number of unique object appearances over all frames.
num_predictions	Total number of unique prediction appearances over all frames.
num_unique_objects	Total number of unique object ids encountered.
mostly_tracked	Number of objects tracked for at least 80 percent of lifespan.
partially_tracked	Number of objects tracked between 20 and 80% of lifespan.
mostly_lost	Number of objects tracked less than 20 percent of lifespan.
num_fragmentations	Total number of switches from tracked to not tracked.
motp	Multiple object tracker precision.
mota	Multiple object tracker accuracy.
precision	Number of detected objects over sum of detected & false positives.
recall	Number of detections over number of objects.

SBT Orchestration



Conclusion

Results

- Implementation of the toolset proved feasibility of SBT concepts
- For IMAGRS, SBT provided a unique ability to evaluate the Extractor
- **SBT shows value as an additional technique for automated testing of military software applications**

Future Work

- Improvements to the toolset
  - Scenario Repository, Simulation Services, SBT Job Workflow
- Mature technology readiness with focus on scale, performance & reliability
- Engage stakeholders in the M&S and software testing communities
  - J7 Joint Training Tools
  - Test team for GCCS-J C2 software
- Roadmap with new capabilities expanding into DEaaS