# Virtual Autonomous System Training (VAST) Project

MSVE Capstone Team 2018-2019

May 10th, 2019

# Purpose of This Briefing

- To signify the final project delivery
- To present project value to client

# Briefing Content

- Project Overview
- Proof-of-Concept
- Conclusion

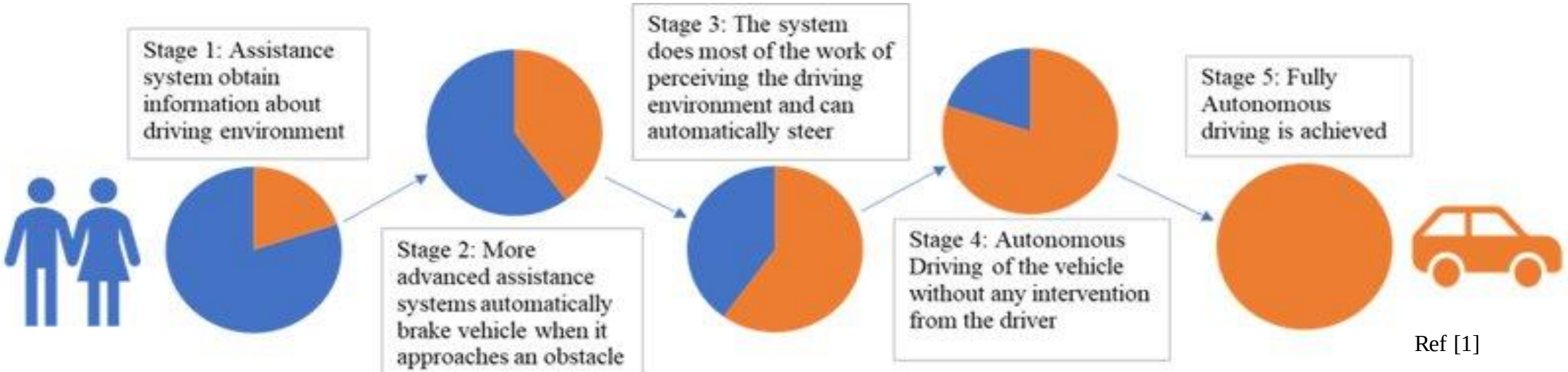
# Participants of the Project

- Students
  - Germaine Badio
  - Elise Beck
  - Zhaohui Hu
  - Peter Marchione
  - Christine Odenwald
  - Jaron Stevenson
  - Travis Sullivan
- Lockheed Martin Corporation (LMC) Mentor
  - Dr. Harry Johnson
- Faculty
  - Dr. Michel Audette
  - Dr. Hong Yang
- Customer Advisor
  - Dr. James Leathrum

# PROJECT OVERVIEW

- Background
- Problem Statement
- Solution Approach
- Architecture

# Autonomous Vehicle (AV) Testing Needs Simulation



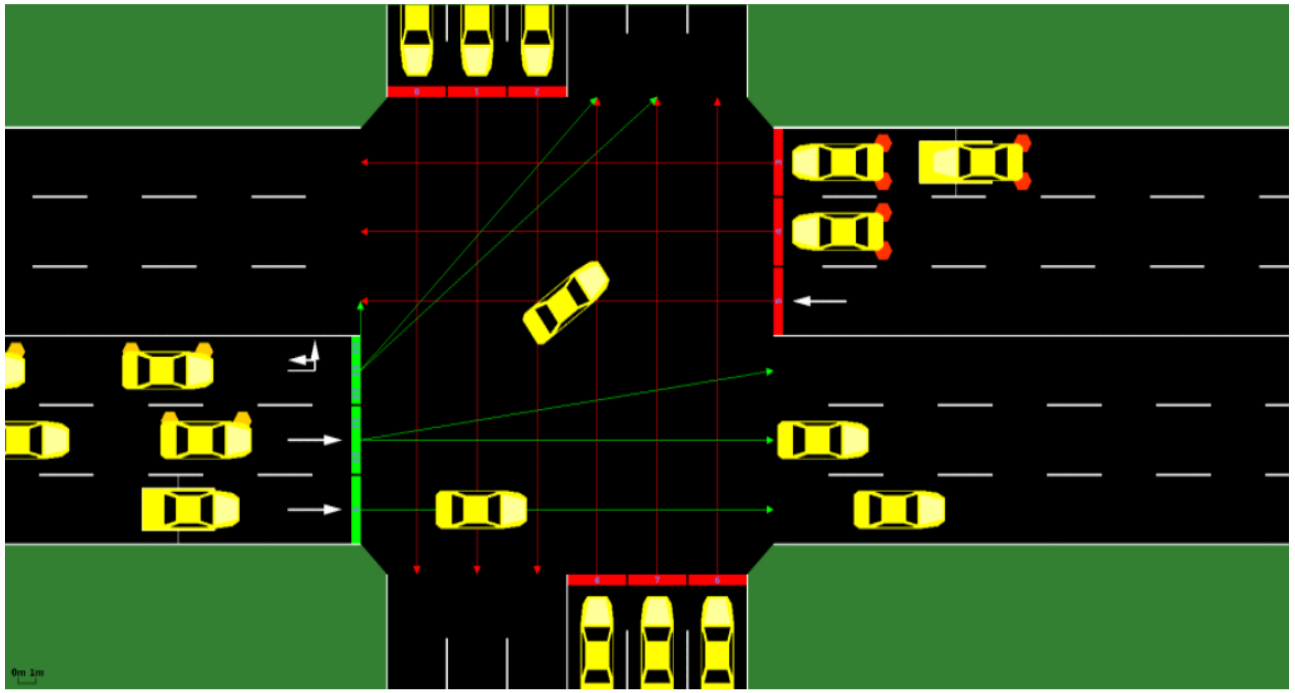

Ref [1]


Ref [2]

# Software Products

- SUMO/TraCI
- Unity
- OpenStreetMap

- OneDrive
- Qt
- Visual Studio
- GitHub
- Tableau

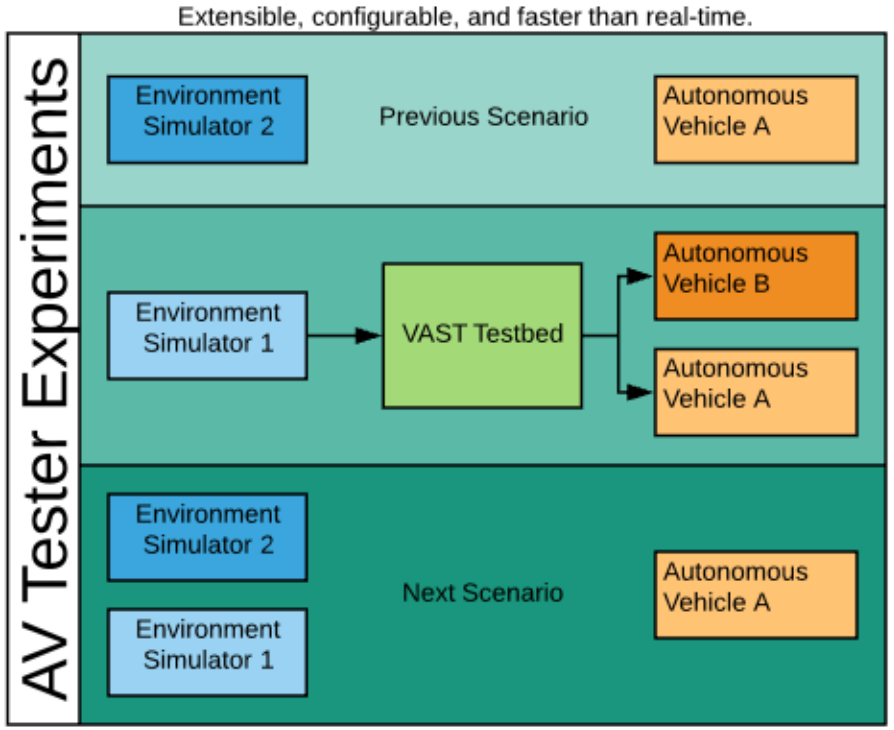


# PROBLEM STATEMENT

At Lockheed Martin Corporation (LMC), the Corporate Engineering, Technology & Operations (CETO) is leading a corporate-wide effort to visualize and understand the impact and implementation approaches for AV technologies. **LMC must verify and validate the potential operational value of land-based AVs by running varies scenarios to ensure the systems are safe, reliable while contributing to mission success.** To rapidly test land-based AV concepts, LMC needs to build a simulation system that allows developers and user to design experiments, test the system, and visualize the scenarios and outcomes, in a cost-effective manner.

# Solution

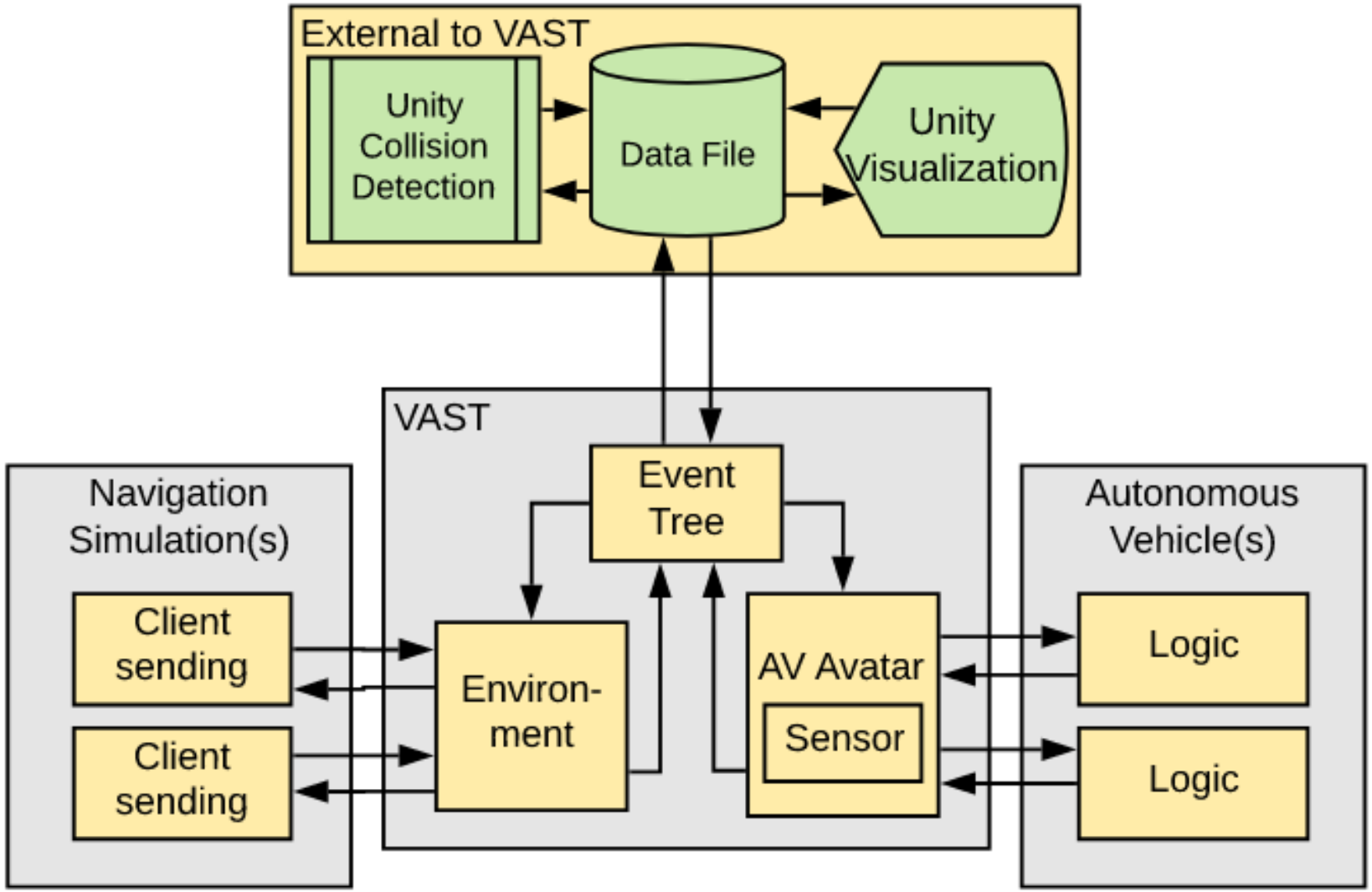
- Rapid scenario builder through User Interface (UI)
- Data collection, metric calculation, and scenario visualization for decision support



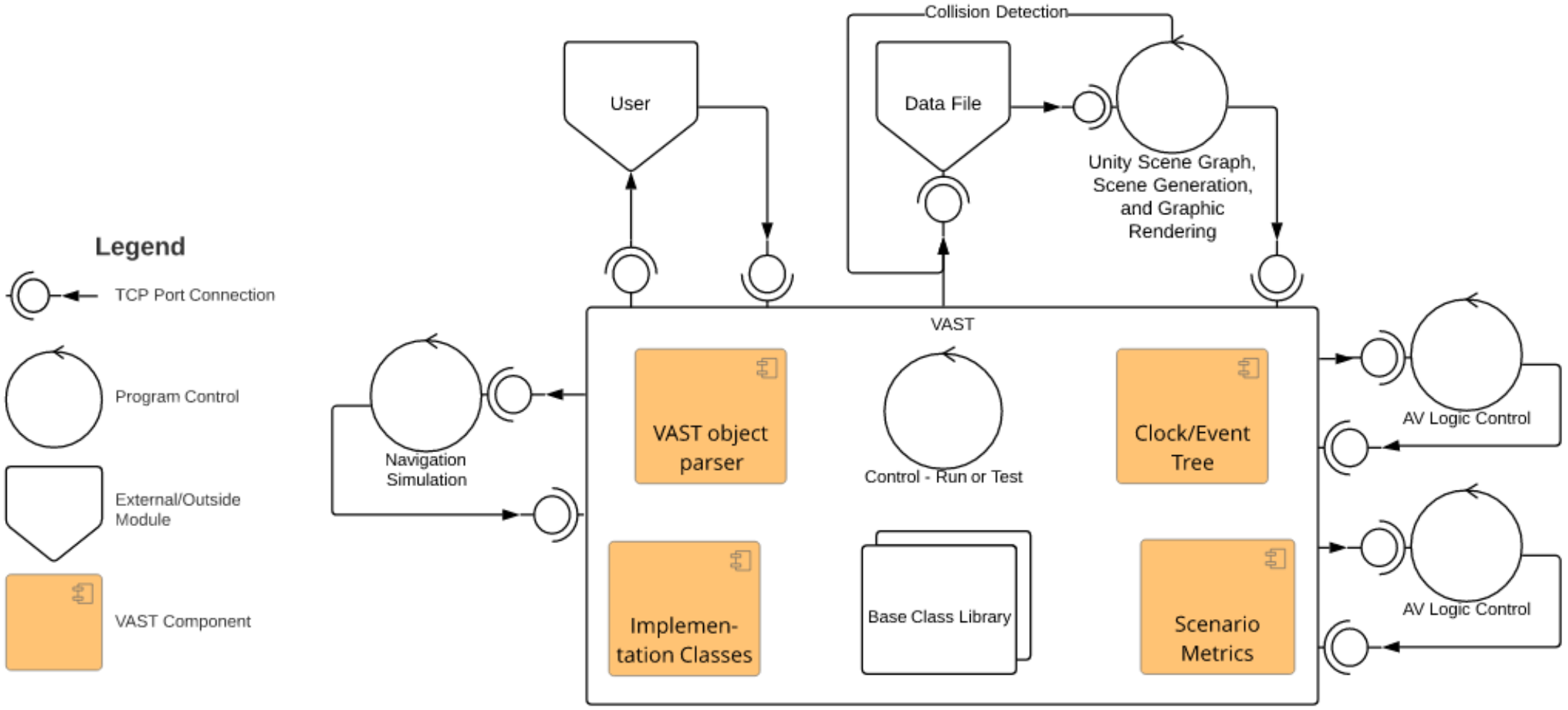
# Project Scope

| Includes | Does not include |
| --- | --- |
| • Processing of inputs and outputs to Navigation Simulation (s) and AV<br>• Post-simulation visualization<br>• User Interface<br>• Simulation for proof of concept | • AV logic<br>• Photorealistic visualization<br>• A mature simulation application |

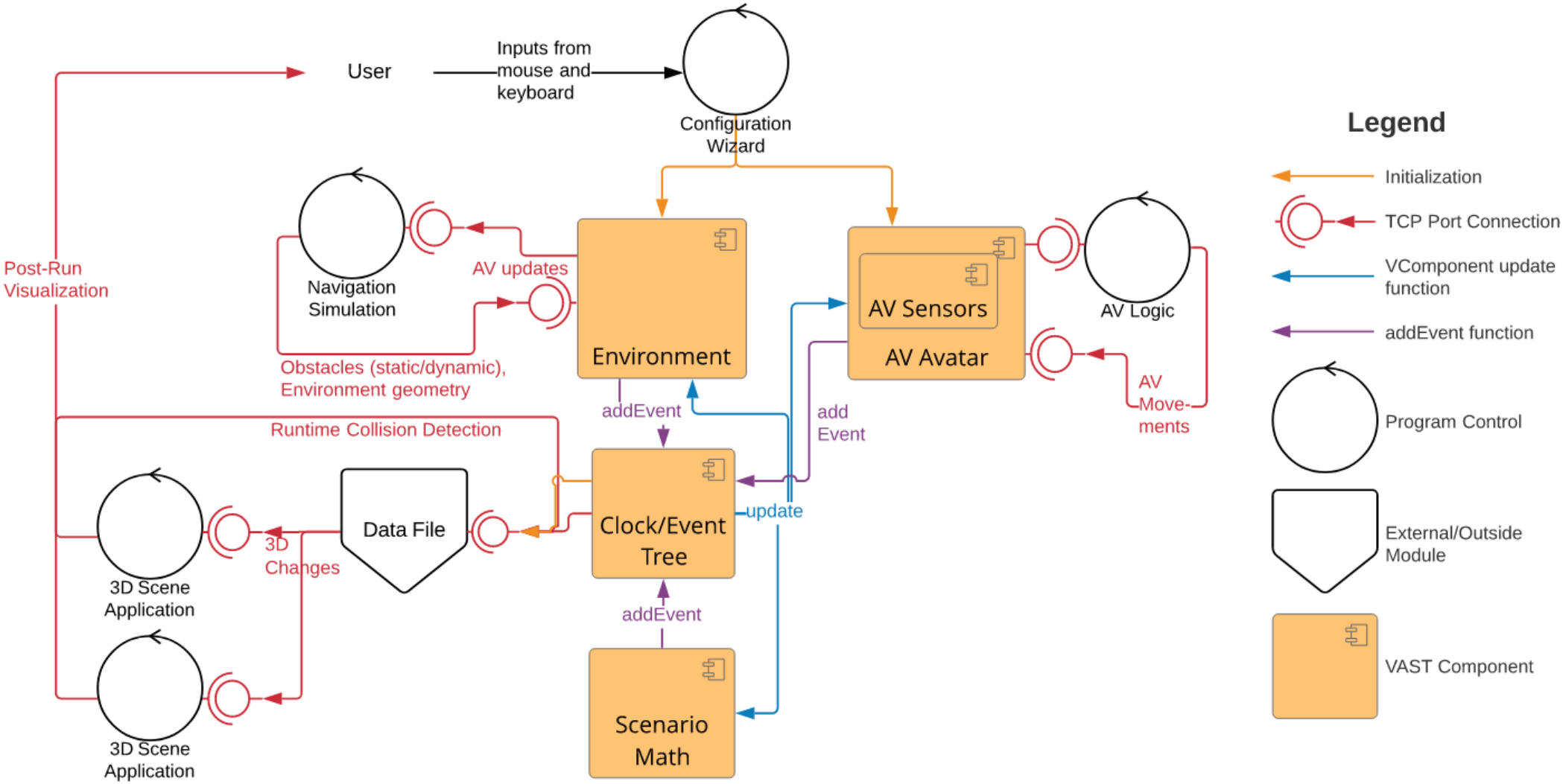
# System Architecture



# VAST System Components

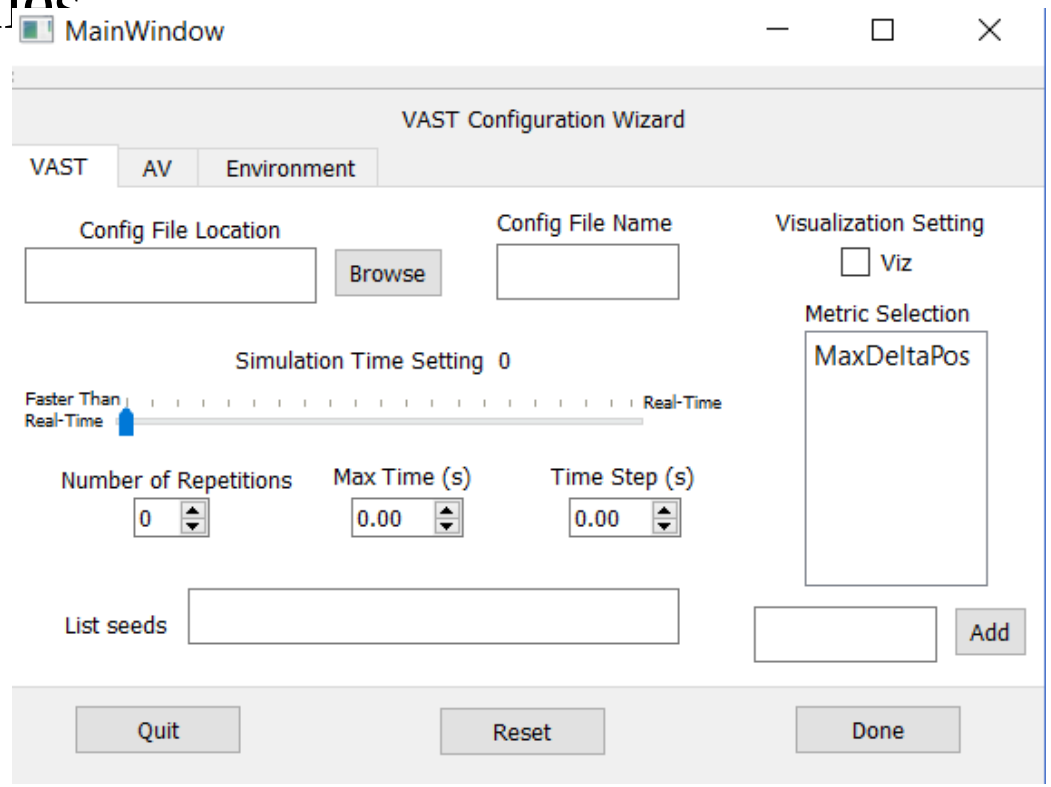


# VAST System Components Flow



# Graphical User Interface (GUI)

- Three tabs, coinciding with primary modules
  - VAST
  - AV
  - Environment
- Persisting Quit, Reset, and Done buttons



# Software Library of Abstract & Basic Classes

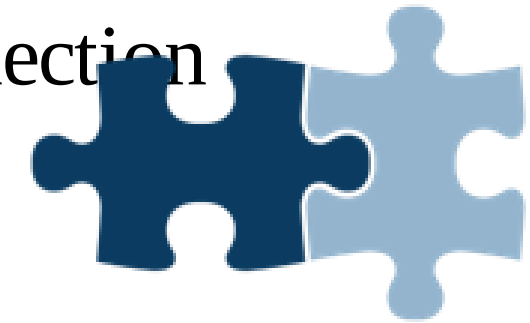
- Purpose: to allow for a multitude of various vehicle and sensor types
- Abstract & extended classes for Scenario Object Library
  - AV class (PythonAV)
  - Sensor class (Proximity)
  - Environment class (SUMOEnvironment)
  - Obstacle class

# PROOF-OF-CONCEPT

- Test Verification
- Demo Description
- Testing Environment

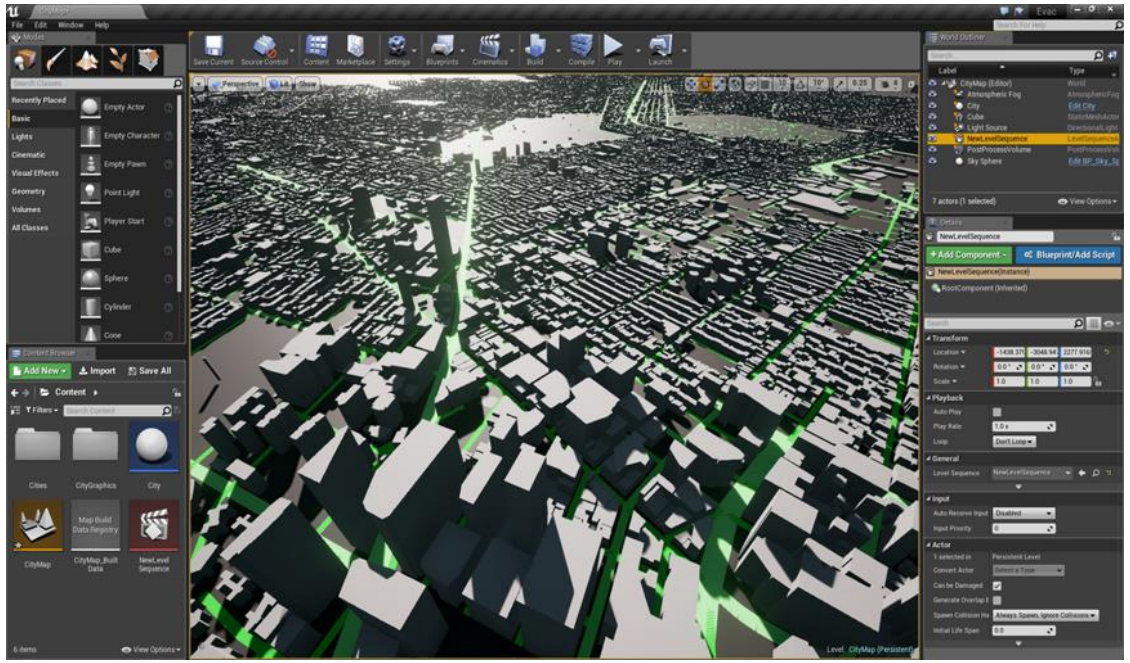
# Verification of System

- Test Plan – follows requirements
  - Verification AND user tool
  - Aligning with the IEEE 829 Standards
- Unit tests – Test-driven development of the code and ability to quickly assess if the functions are working properly
- Integration tests – each component connection



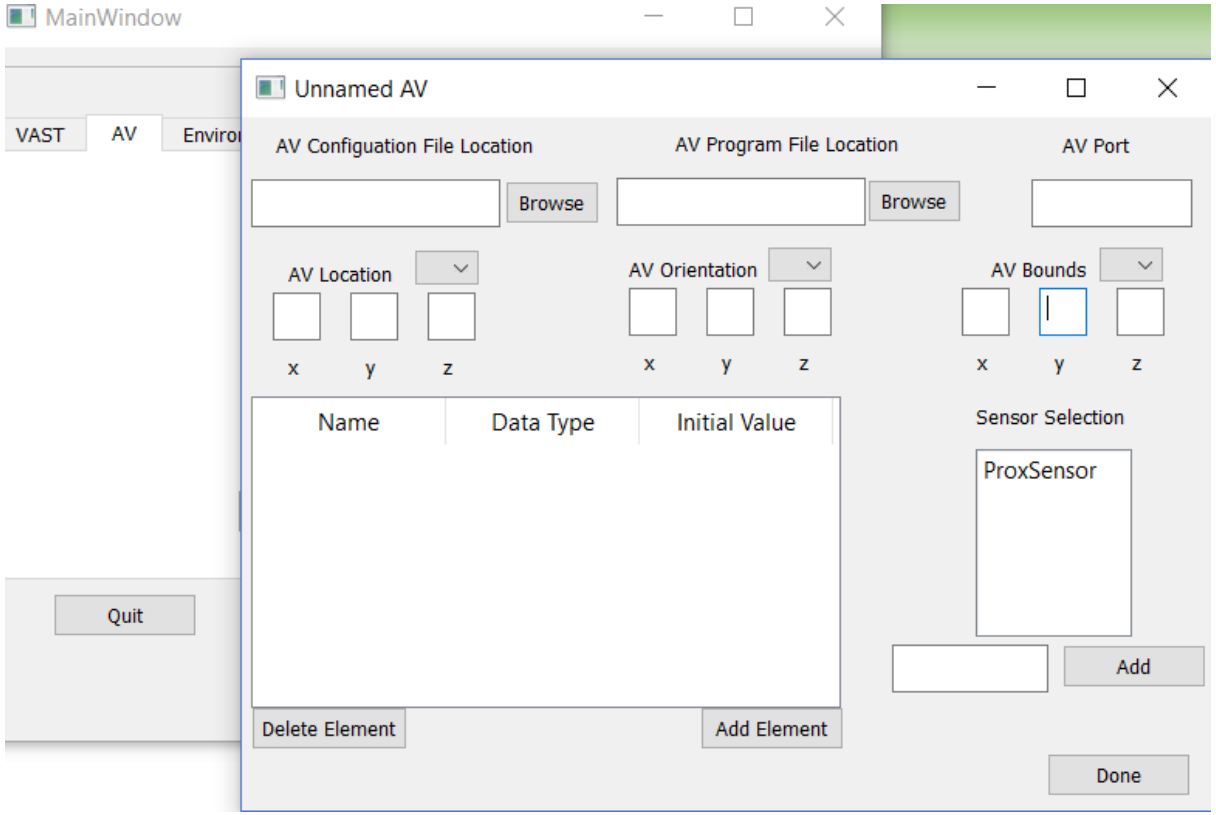
# Proof-of-Concept Demo

- Configuration of VAST
- Replication runs
- Coordination of SUMO and VAST
  - OpenSourceMap
  - TraCI
- Unity Post-Simulation Visualization
- Unity Collision Detection

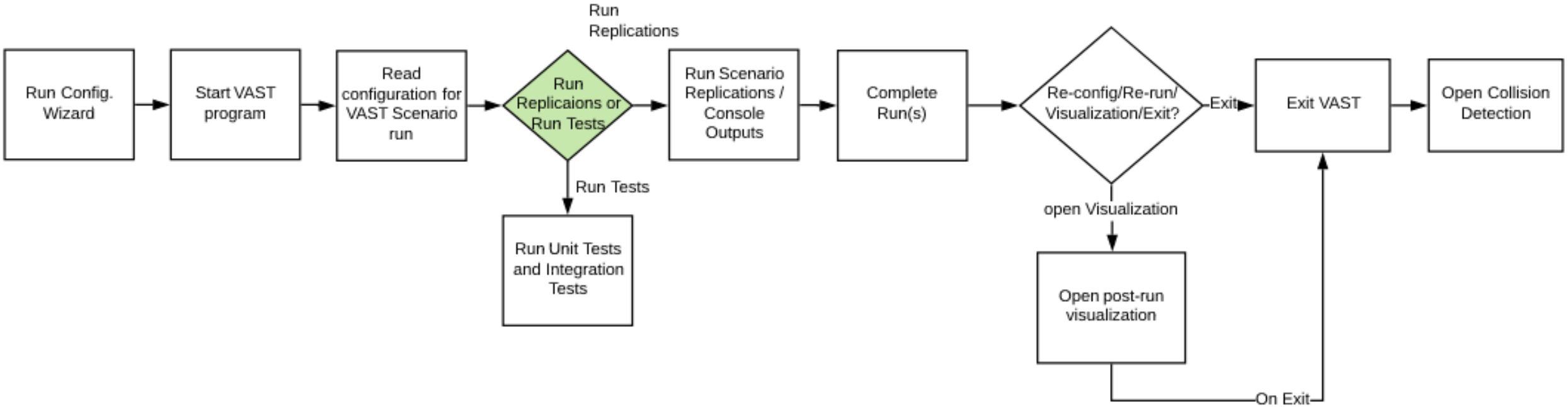


# VAST Configuration - GUI

- Demo
- Executable in VAST folders
  - 3-part menu
  - Mandatory and Optional fields
  - Xml file generated



# VAST Run

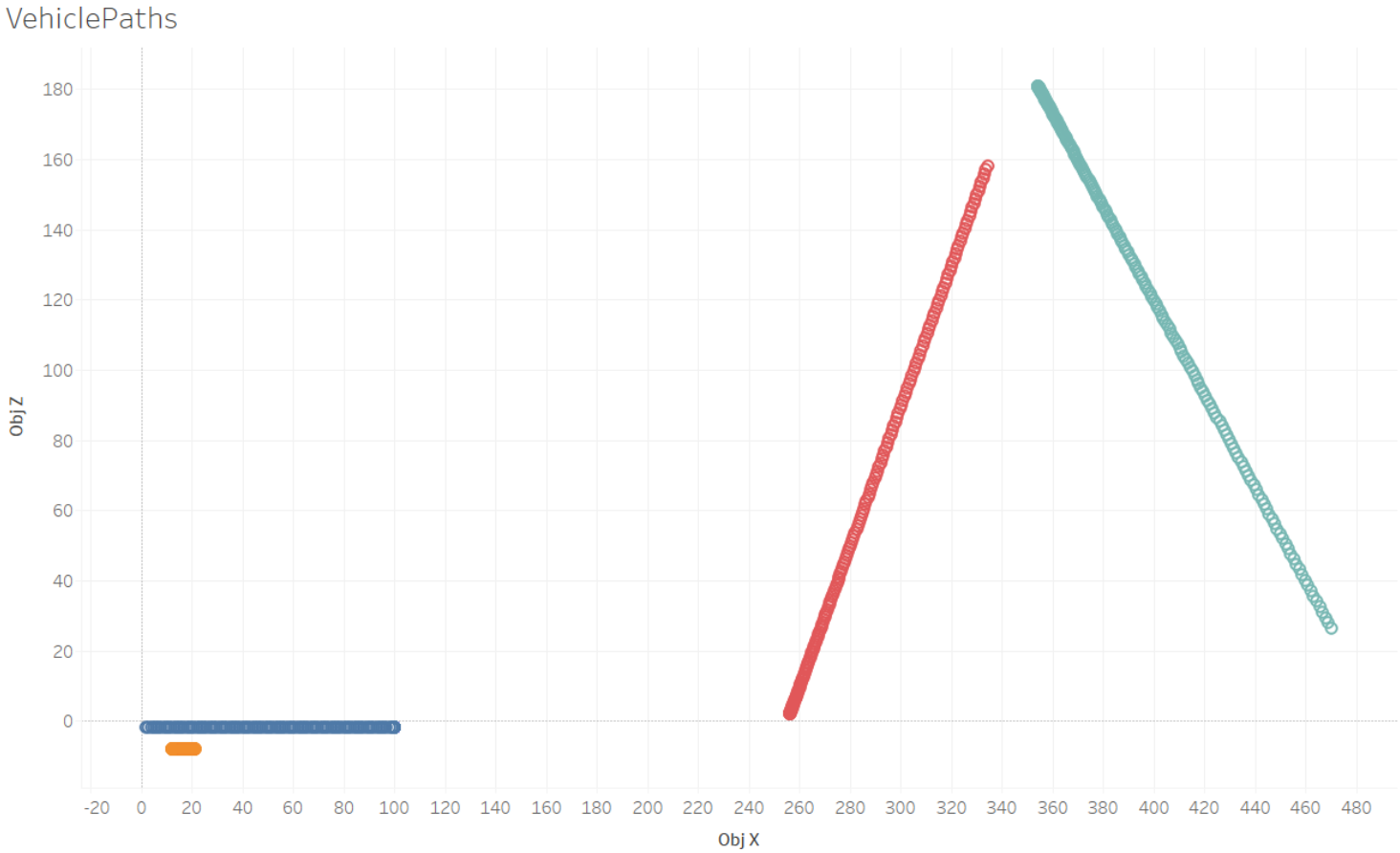

Run Config. Wizard
Start VAST program
Read configuration for VAST Scenario run
Run Replicaions or Run Tests
Run Replications
Run Tests
Run Unit Tests and Integration Tests
Run Scenario Replications / Console Outputs
Complete Run(s)
Re-config/Re-run/ Visualization/Exit?
Exit
Exit VAST
Open Collision Detection
open Visualization
Open post-run visualization
On Exit

# Collision Detection Process

- Demo
- Using Built-in "OnCollisionEnter" Unity Function and Custom Physics System Settings
  - Layer Collision Matrix selections
  - Broadphase types supported
    - Sweep and Prune
    - Multi-Box Pruning
- Different colliders types
  - Box colliders
  - Mesh colliders for AV

# Metric Tableau Visualization

- Allow user to quickly view metrics and understand trends in data



# Conclusions

# Summary

- The proof of concept is a modification on the design
- VAST facilitates the communication between simulation(s) and AV(s)
- VAST reports relevant performance metrics
- VAST has been validated per our test procedures
- VAST is extensible by using multiple abstract classes

# Future Work

- AV travels in various paths (not just a straight line)
- VAST interaction with multiple simulations and AVs
- VAST interaction with various simulation and AV types

# Questions?

# Image Credits

- Integration (Puzzle Pieces): S&T Technologies

# References

[1] Institute of Transportation Systems, "SUMO - Simulation of Urban MObility," [Online]. Available: https://www.dlr.de/ts/en/desktopdefault.aspx/tabid-9883/16931_read-41000/. [Accessed 7 February 2019].
[2] Unity Technologies, "Physics," 17 January 2019. [Online]. Available: https://docs.unity3d.com/Manual/class-PhysicsManager.html. [Accessed 6 February 2019].
[3] "OpenStreetMap," [Online]. Available: https://www.openstreetmap.org/. [Accessed 6 February 2019].
[4] Nvidia, "Nvidia Pro Pipeline," [Online]. Available: https://developer.nvidia.com/nvidia-pro-pipeline. [Accessed 6 February 2019].
[5] "Networks/Import/OpenStreetMap," 26 January 2019. [Online]. Available: http://sumo.dlr.de/wiki/Networks/Import/OpenStreetMap. [Accessed 4 February 2019].
[6] Mapbox, "Maps SDK for Unity," [Online]. Available: https://docs.mapbox.com/unity/maps/overview/. [Accessed 6 February 2019].
[7] "Database (SQLite) Setup for Unity," 8 July 2014. [Online]. Available: https://answers.unity.com/questions/743400/database-sqlite-setup-for-unity.html. [Accessed 6 February 2019].
[8] Unity Technologies, "Continuous Collision Detection (CCD)," 17 January 2019. [Online]. Available: https://docs.unity3d.com/Manual/ContinuousCollisionDetection.html. [Accessed 6 February 2019].
[9] Unity Technologies, "Collision," 17 January 2019. [Online]. Available: https://docs.unity3d.com/ScriptReference/Collision.html. [Accessed 6 February 2019].
[10] Unity Technologies, "Colliders," 17 January 2019. [Online]. Available: https://docs.unity3d.com/Manual/CollidersOverview.html. [Accessed 6 February 2019].
[11] S. Tamilarasan, D. E. Jung and L. Guvenc, "Drive Scenario Generation Based on Metrics for Evaluating an Autonomous Vehicle Controller," in WCX World Congress Experience, 2018.

# References

[12] D. Stanek, E. Huang, R. T. Milam and Y. A. Wang, "Measuring Autonomous Vehicle Impacts on Congested Networks Using Simulation," in Transportation Research Board Annual Meeting, Washington, DC, 2017.
[13] C. Scrapper, S. Balakirsky and B. Weiss, "Autonomous Road Driving Arenas for Performance Evaluation," 24 August 2004. [Online]. Available: https://ws680.nist.gov/publication/get_pdf.cfm?pub_id=822519. [Accessed 6 February 2019].
[14] C. R. Rickarby, "Autonomous Driving Platform Performance Analysis," 2017. [Online]. Available: https://scholars.unh.edu/cgi/viewcontent.cgi?article=1344&context=honors. [Accessed 9 February 2019].
[15] W. G. Phillip J Durst, "Levels of Autonomy and Autonomous System Performance Assessment for Intelligent Unmanned Systems," April 2014. [Online]. Available: https://apps.dtic.mil/dtic/tr/fulltext/u2/a601656.pdf. [Accessed 6 February 2019].
[16] M. Fricker, "Street Map Plugin for UE4," 18 June 2018. [Online]. Available: https://github.com/ue4plugins/StreetMap. [Accessed 6 February 2019].
[17] A. Fox, "The Difference Between a CPU and a GPU," Make Tech Easier, 31 January 2017. [Online]. Available: www.maketecheasier.com/difference-between-cpu-and-gpu. [Accessed 6 February 2019].
[18] S. Cope, "TCP/IP Ports and Sockets Explained," 6 July 2018. [Online]. Available: http://www.steves-internet-guide.com/tcpip-ports-sockets/. [Accessed 6 February 2019].
[19] "Welcome to AirSim," [Online]. Available: https://microsoft.github.io/AirSim/. [Accessed 6 February 2019].
[20] "TraCI/Change Vehicle State," 28 November 2018. [Online]. Available: http://www.sumo.dlr.de/userdoc/TraCI/Change_Vehicle_State.html. [Accessed 6 February 2019].
[21] "Unity Blog," [Online]. Available: https://blogs.unity3d.com/. [Accessed 6 February 2019].
[22] Problem Discovery Affecting OT&E

# Back-Up Slides

# Verification of VAST

| Requirement | Associated Measures |
|---|---|
| AV Testing Environment shall accurately represent data input to clients. | -Data format reliability<br>-Data transfer success rate |
| AV Testing Environment shall visualize the environment accurately. | -AV position reliability in visualization |
| AV Testing Environment shall be abstracted to handle multiple sensor and vehicle types that can be inputted by the user. These sensor and vehicle types shall have acceptable and realistic parameters. | -Sensor list reliability |
| AV Testing Environment shall receive and store scene simulation information. | -Scene object position reliability<br>-Scene update rate |
| AV Testing Environment shall accurately state the output made by the AV. | -Output reliability |
| AV Testing Environment shall evaluate whether a failure/operational mission failure has occurred and output its result. | -Collision detection rate |
| AV Testing Environment shall report relevant performance metrics. | -Number of Metrics<br>-Correct Metrics |

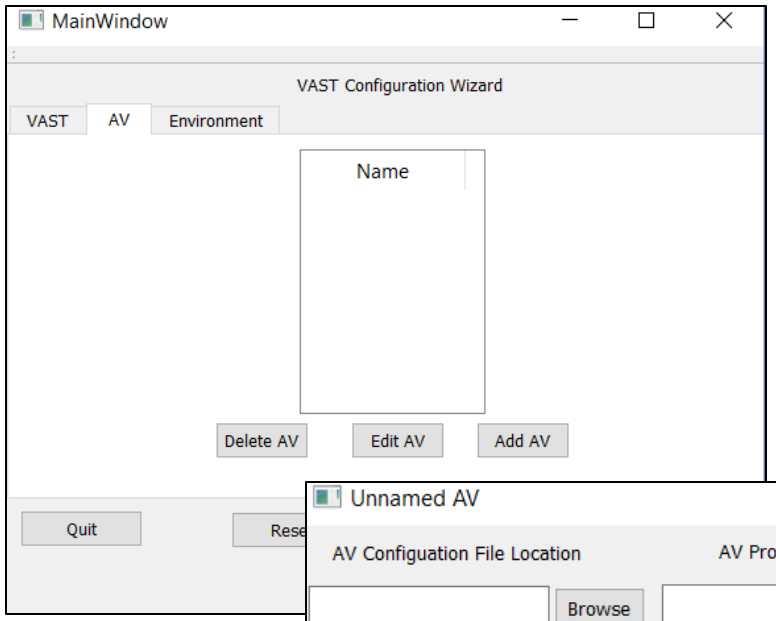
# Testing Limitations

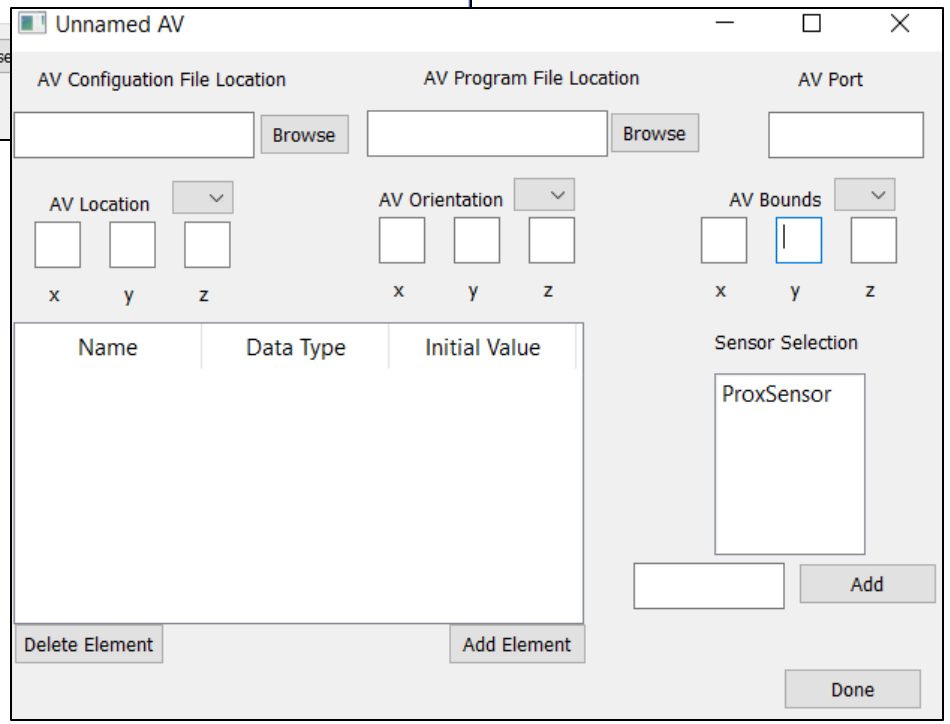
- Does not include hardware-in-the-loop
  - New problems are likely to arise once both hardware and software are tested in conjunction
  - Director, Operational Test and Evaluation (DOT&E) Fiscal Year (FY) 17 Report
    - 34.4% OTs conducted discovered new, critical problems in addition to old problems.
    - 15.7% of OTs discovered new, critical problems.
- Does not include the interaction of the system of systems (SOS) with the system under test (SUT)

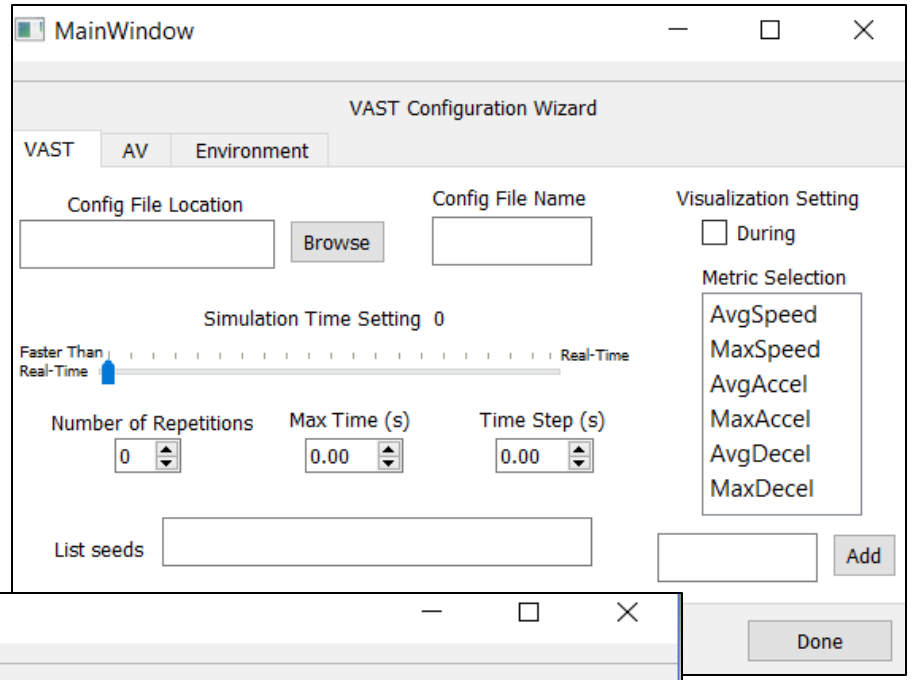
# Configuration Wizard

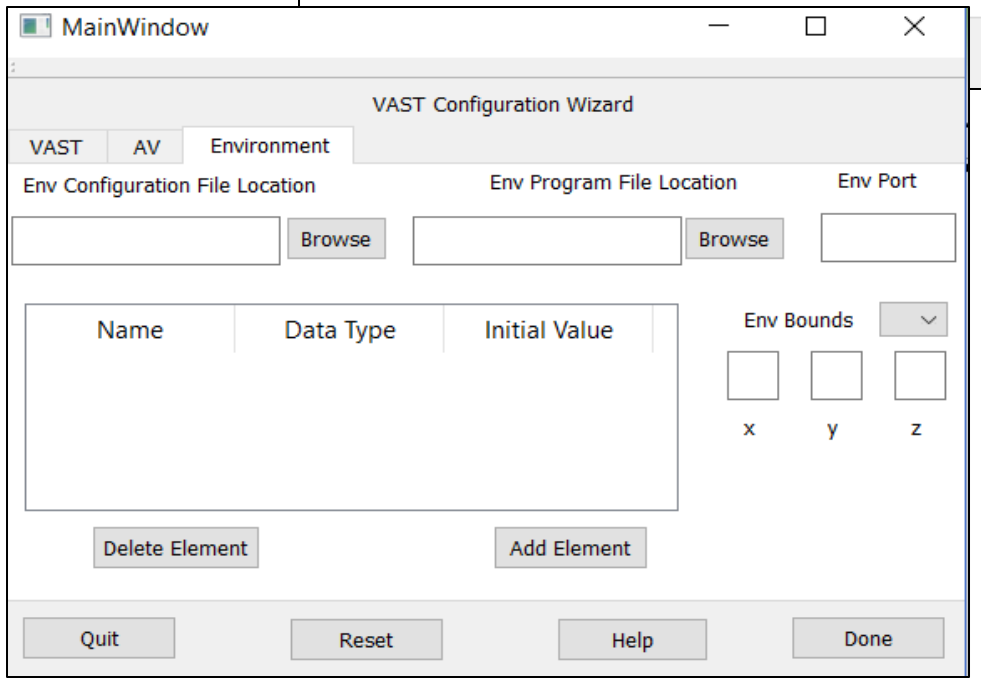
- Requirement:
  - Parser Tests related to instantiating objects.
  - AV Testbed shall accurately represent data input to clients.
- Basic procedure:
  - Creating a test case for a sample xml file format and adding in information
  - Checking the xml file to make sure it matches the expected output

# VAST Configuration - GUI



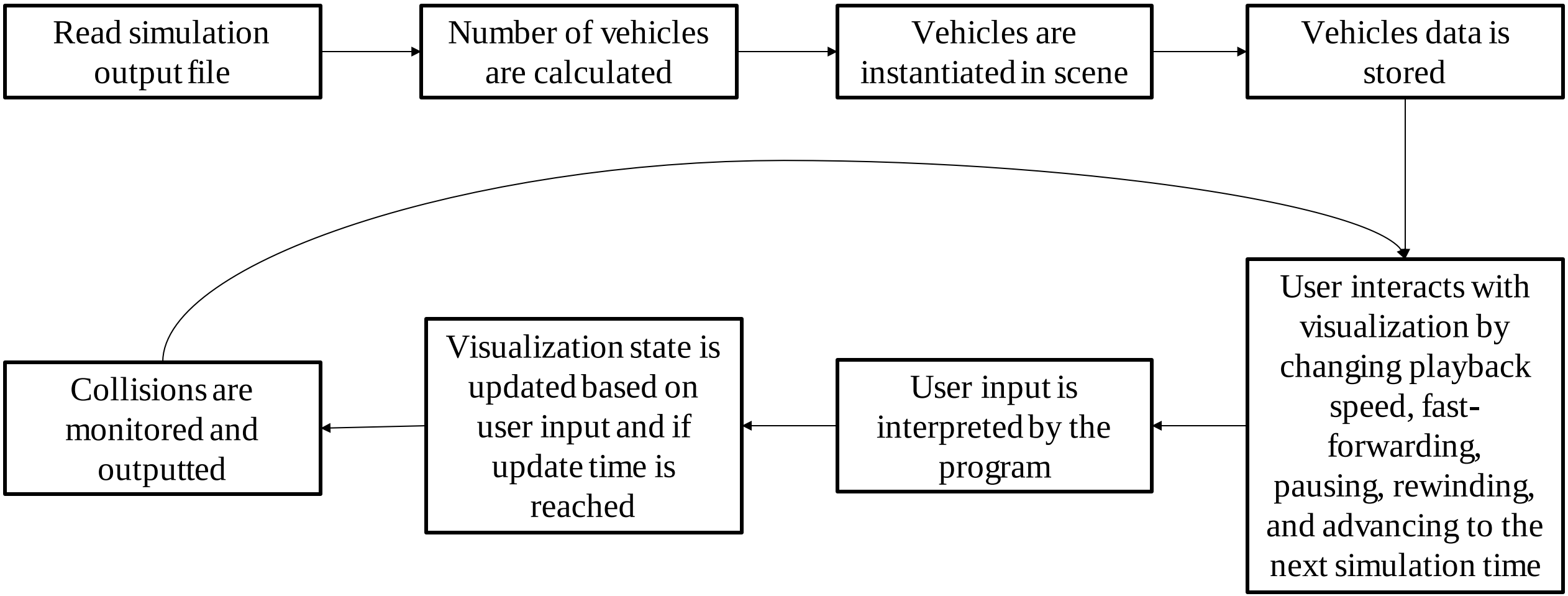



# Collision Detection Testing

- Requirement: the testbed outputs requirement of evaluating whether a failure/operational mission failure has occurred and output its result.
- Should report to the database:
  - Run number
  - AV ID
  - Time of the collision
  - Position in the environment of the collision
  - Object Name AV collided with
- Two versions: post-simulation and faster-than-real-time

# Post-Simulation Visualization Logic



# AV

- Requirement:
  - Information should be passed from the Ground AV class to VAST via the Event Tree
  - AV Testbed shall accurately state the output made by the AV
- Basic procedure:
  - Construct Event Tree and verify registration of VComponents.
  - Check sensor work well
  - Check Unit Test whether to pass

# VAST

- Requirement:
  - Sent data to database
  - Get different value type
- Basic procedure:
  - Construct Database and check functionality of database operation
  - Check VType work well
  - Check VAST library work well
  - Check Event Tree work well

# Environment

- Requirement:
  - Information should be passed from the SUMO Environment class to VAST via the Event Tree
  - AV Testbed shall visualize the environment accurately
  - AV Testbed shall be abstracted to handle multiple sensor and vehicle types that can be inputted by the user. These sensor and vehicle types shall have acceptable and realistic parameters.
- Basic procedure:
  - Set up Traci and connect with SUMO
  - Check Unit Test whether to pass

# Visualization

- Requirement:
  - Visualization needs to be properly launched
  - AV Testbed shall visualize the environment accurately
  - AV Testbed shall receive and store scene simulation information.
- Basic procedure:
  - Reading the database, generated Unity graph based on the simulation and AV interaction.
  - Check the output is same as database.