CARACaS for USV Swarm Autonomy



JPL Principal Investigator: Dr. Michael Wolf (wolf@jpl.nasa.gov)
ONR Program Manager: Dr. Robert Brizzolara

Jet Propulsion Laboratory
California Institute of Technology

Acknowledgments

Office of Naval Research

Collaboration Team

Spatial Integrated Systems

DH Wagner Associates

NSWC–Carderock, Combatant Craft Division

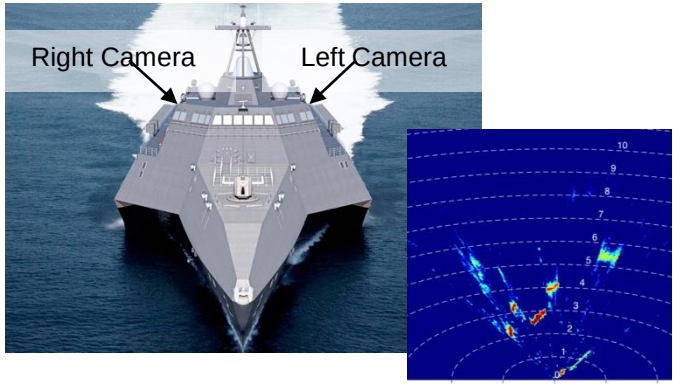
Current Projects in Maritime Autonomy

ASW Continuous Trail Unmanned Vehicle (ACTUV)

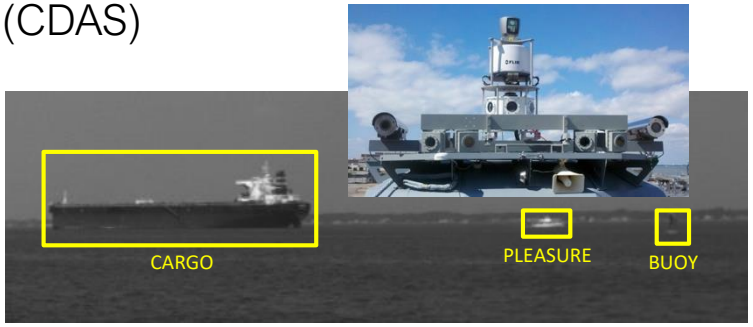
USV Swarm

Unmanned Influence Sweep System (UISS)

Mechanically Uncoupled Stereo EO/IR (MUSE)



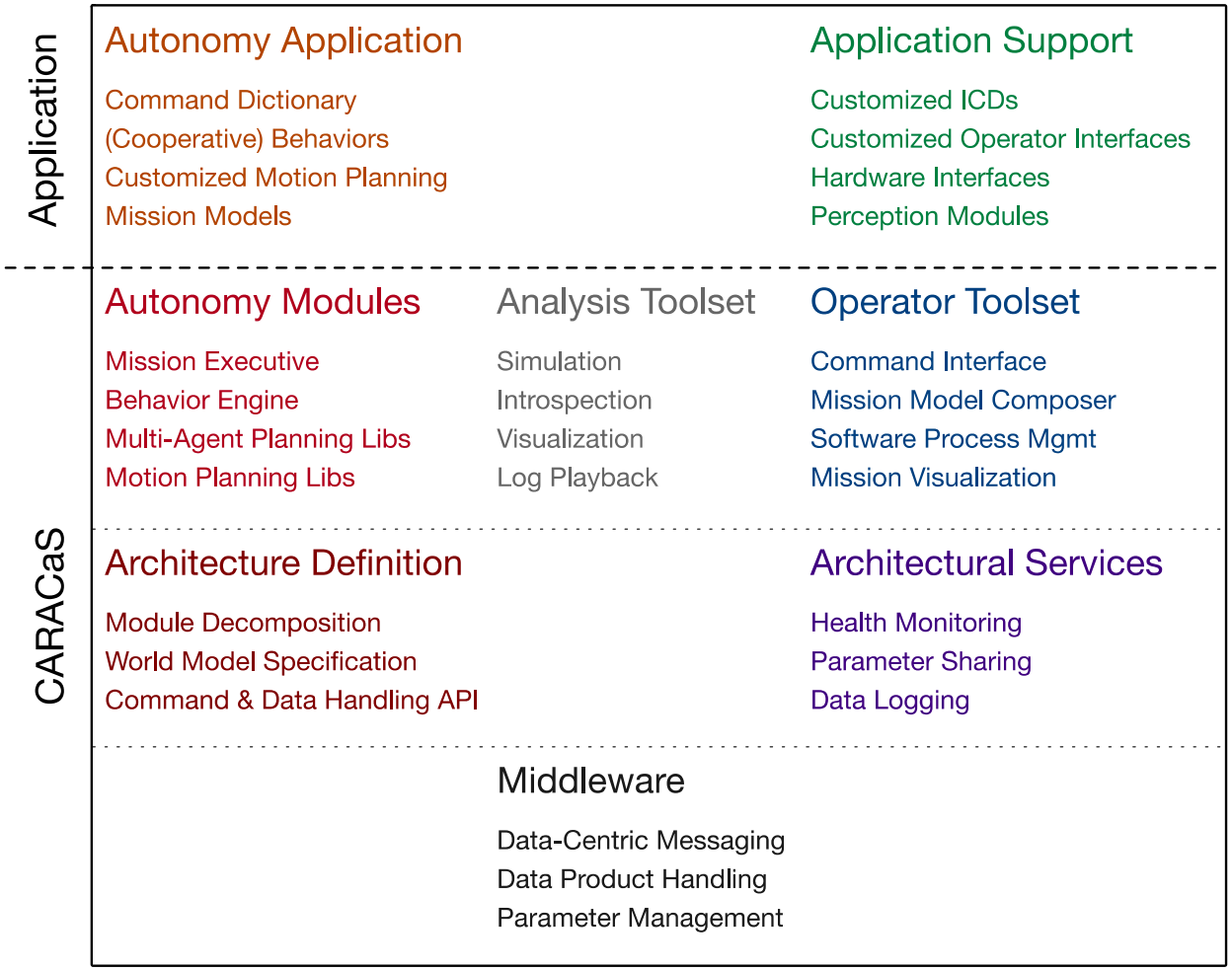
Contact Detection and Analysis System (CDAS)



CARACaS (Control Architecture for Robotic Agent Command and Sensing)

Need: a multi-mission behavior development platform, across a range of vehicles and applications.

CARACAS provides foundational software infrastructure, architecture definition, robotic technology modules, and a development environment for new modules and multi-agent mission behaviors.



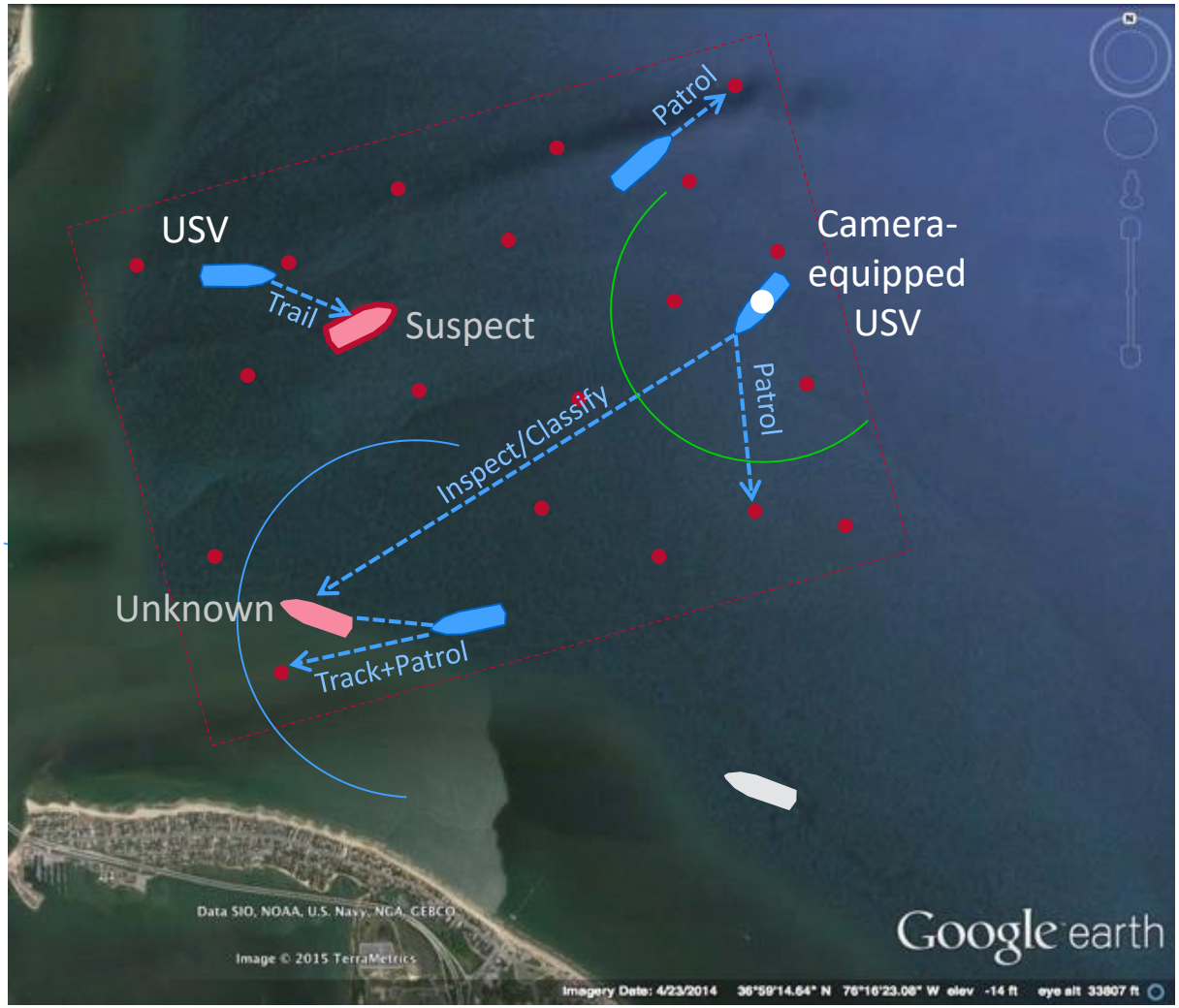
USV Swarm 2 Demonstration (SEP 2016)

Behaviors

- **Patrol** – cooperatively monitor a defined area
- **Track** – keep a vessel in radar range, while patrolling nearby
- **Inspect** – dispatch a USV with cameras to classify a vessel
- **Trail** – closely follow a suspect vessel

Technology Objectives

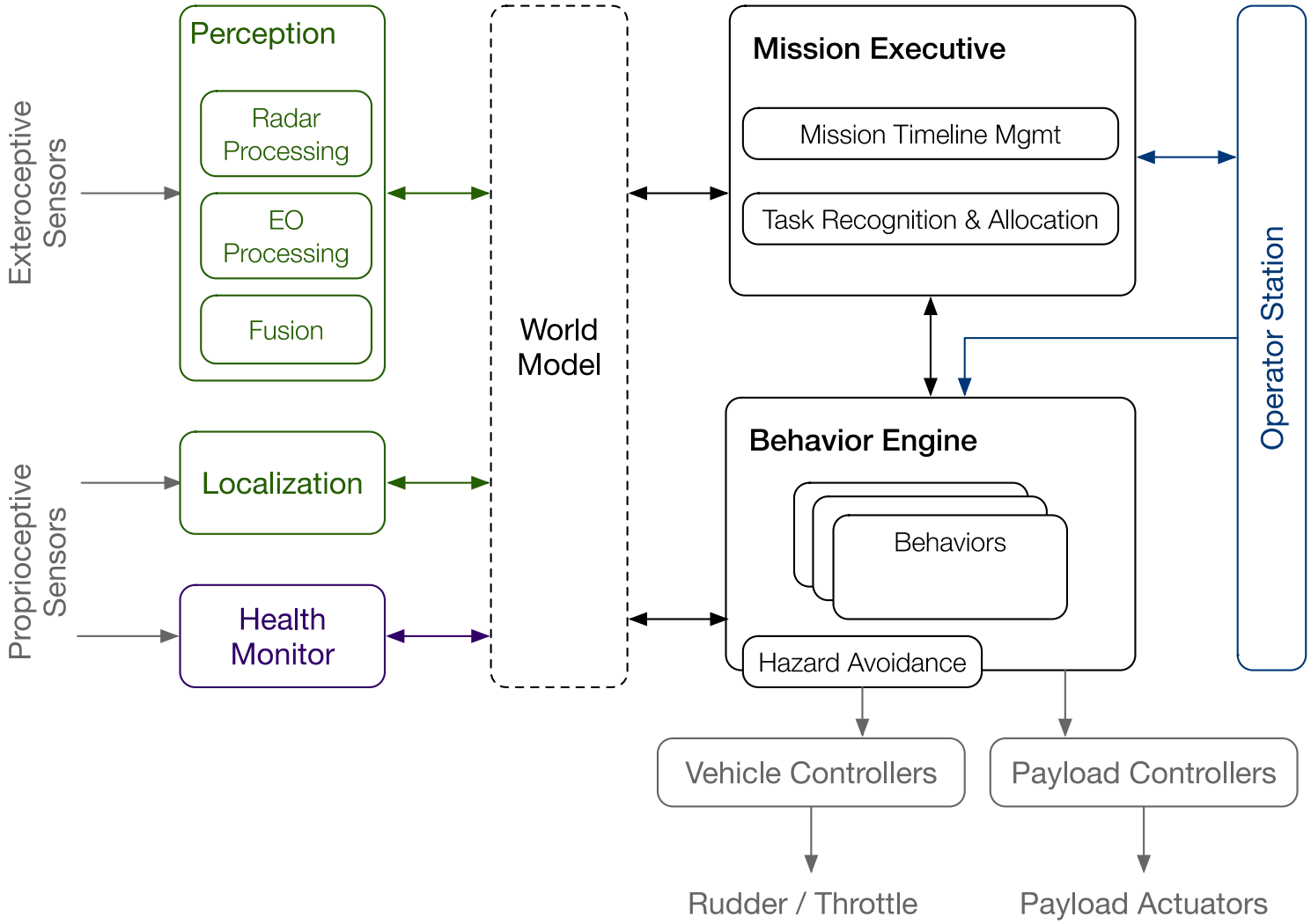
- Higher levels of autonomy / less operator input
- Autonomous task recognition and allocation
- Planning for heterogeneous teams
- Adaptation to attrition and subsystem failures
- Decreased bandwidth requirements



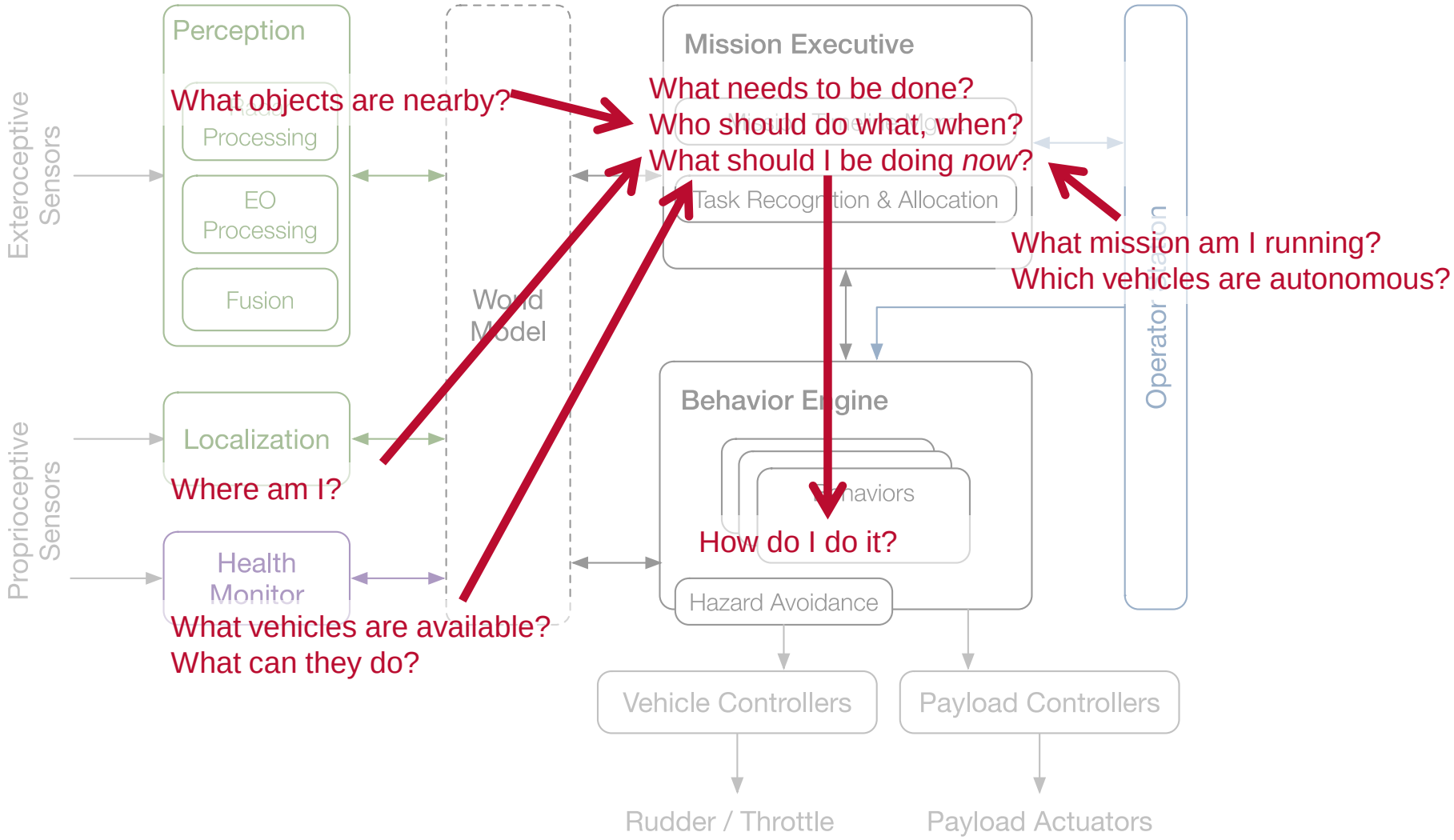
Key Ideas for Swarm 2 Autonomy

- Autonomy must be capable of event-driven operations in highly dynamic environments.
  - Not just sequenced, waypoint-driven missions
  - Swarm needs to *recognize* the tasks required to complete its mission as well as *allocate* those tasks to agents
  - Missions are complex and composed of multiple behaviors
- Focus was multi-mission autonomy architecture with customizable behaviors.
  - Not necessarily optimizing a “patrol agents” system
  - Use a specific, sufficiently complex example to define and exercise a multi-agent autonomy architecture and corresponding cooperative behaviors
- Autonomy systems should be robust and flexible
  - No software component knew any of the following beforehand:
    - how many USVs would be participating
    - how many USVs had camera systems
    - how many contacts might be encountered
  - Discovering conditions in runtime and re-planning accordingly enables both operational flexibility and robustness to health faults / changing conditions

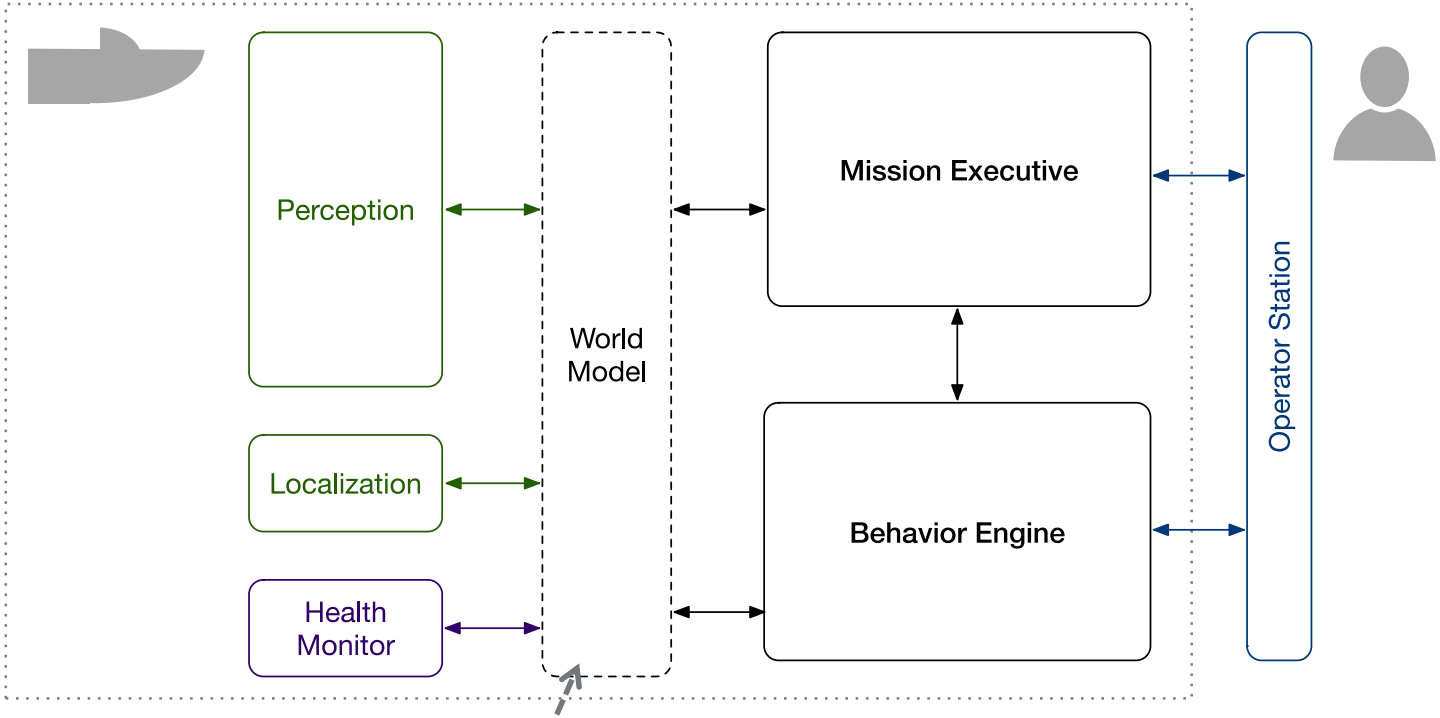
CARACaS: Functional Architecture

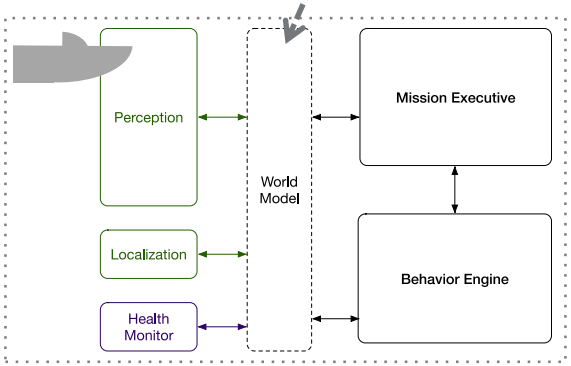


CARACaS: Functional Architecture



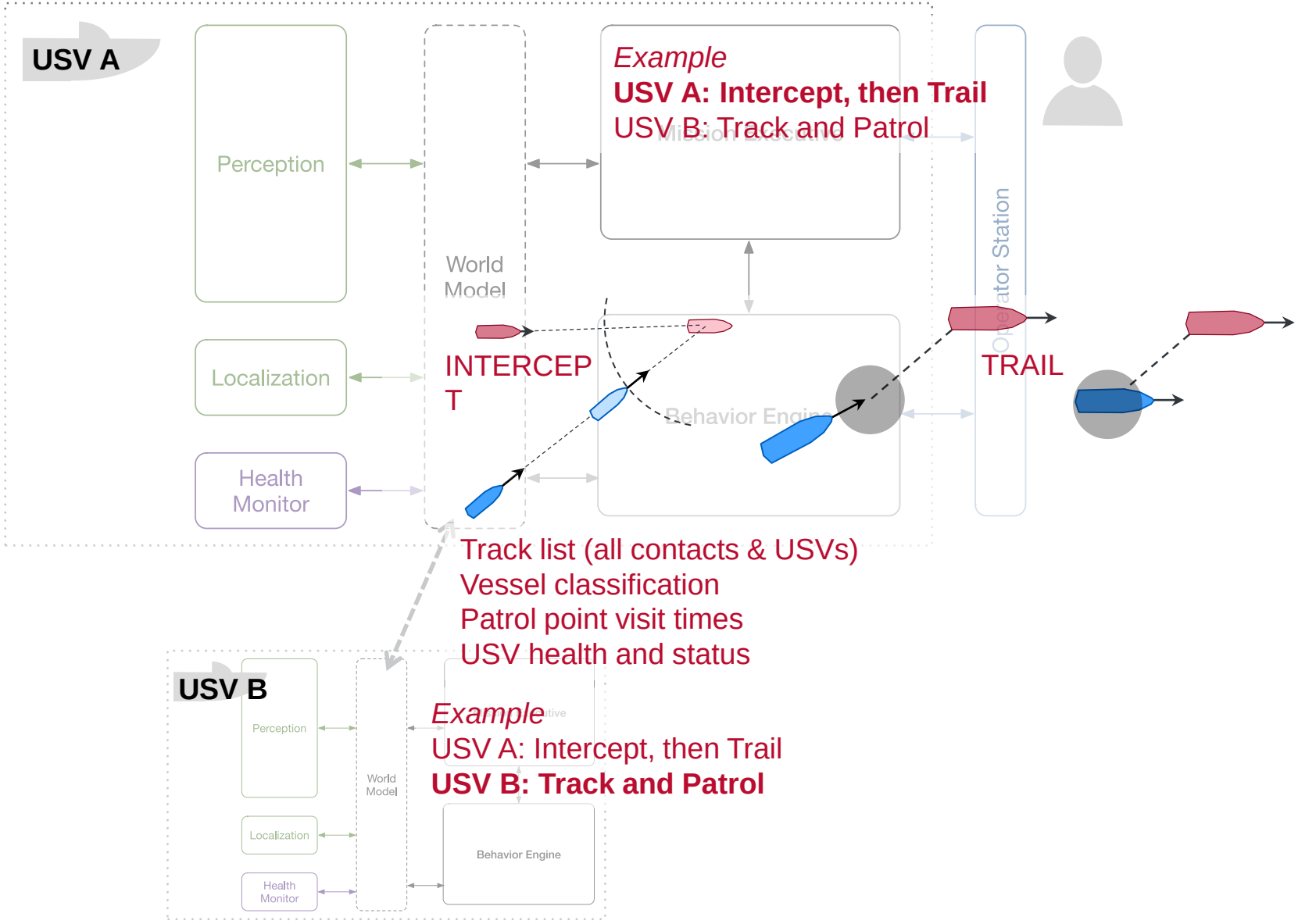
Cooperative Autonomy for USV Swarm



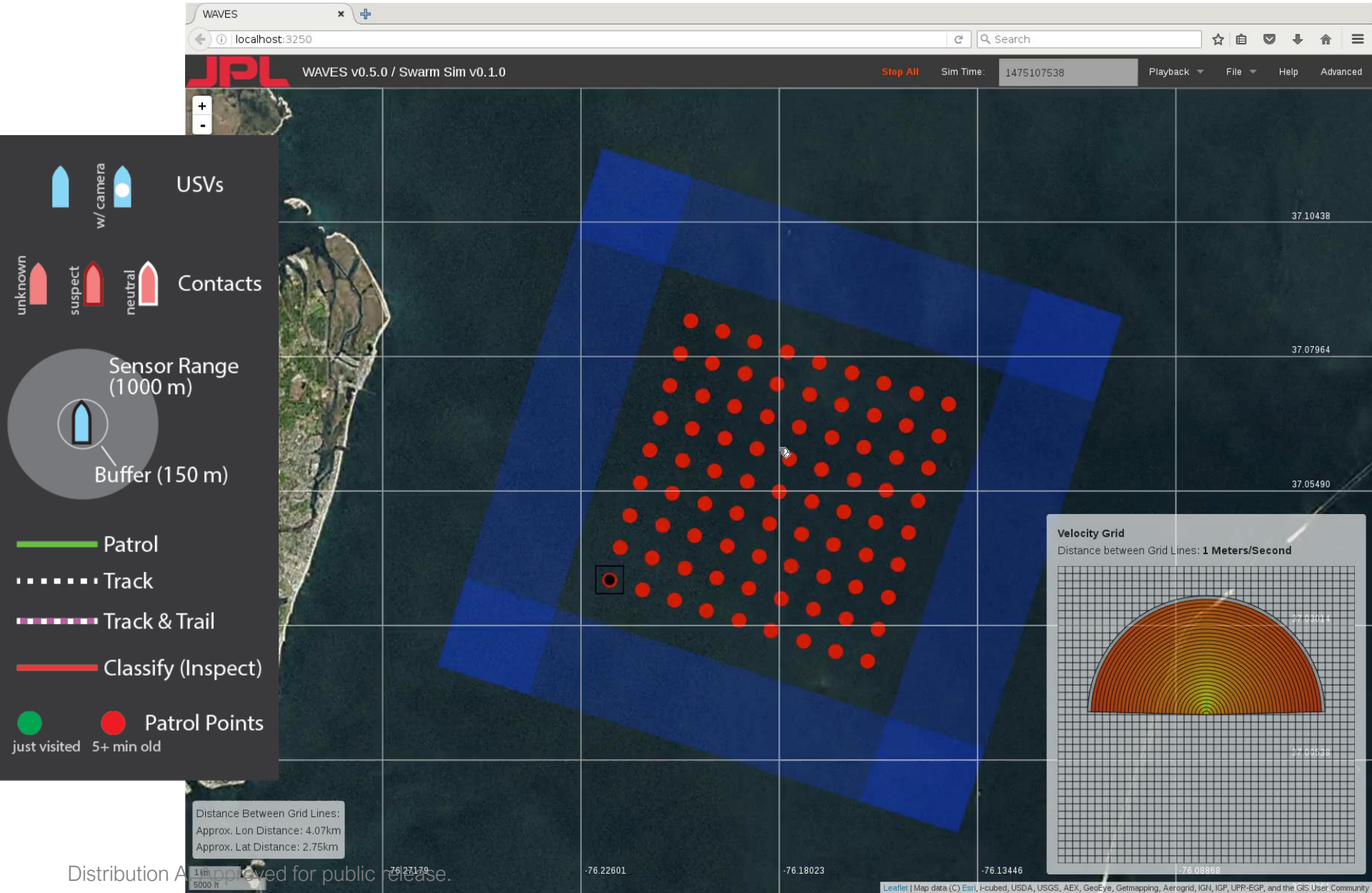


- World Models are synchronized (communications allowing).
- All Planning is decentralized.
- Mission Executive creates a timeline for itself *and* predictions of other agents' plans.
- Agents execute their own local plan; no additional communication needed.

Cooperative Autonomy for USV Swarm

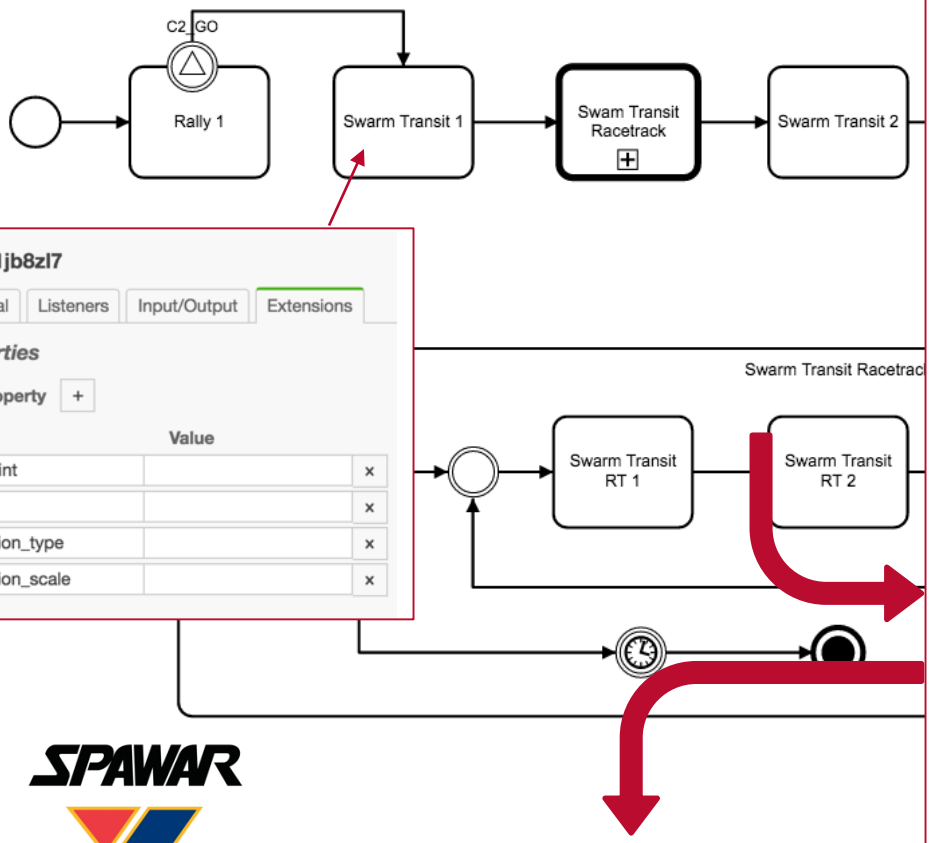


Swarm 2 Mission Video



Human Element: Defining a Mission

CARACaS Mission Executive ingests missions in
Business Process Modeling and Notation (BPMN)

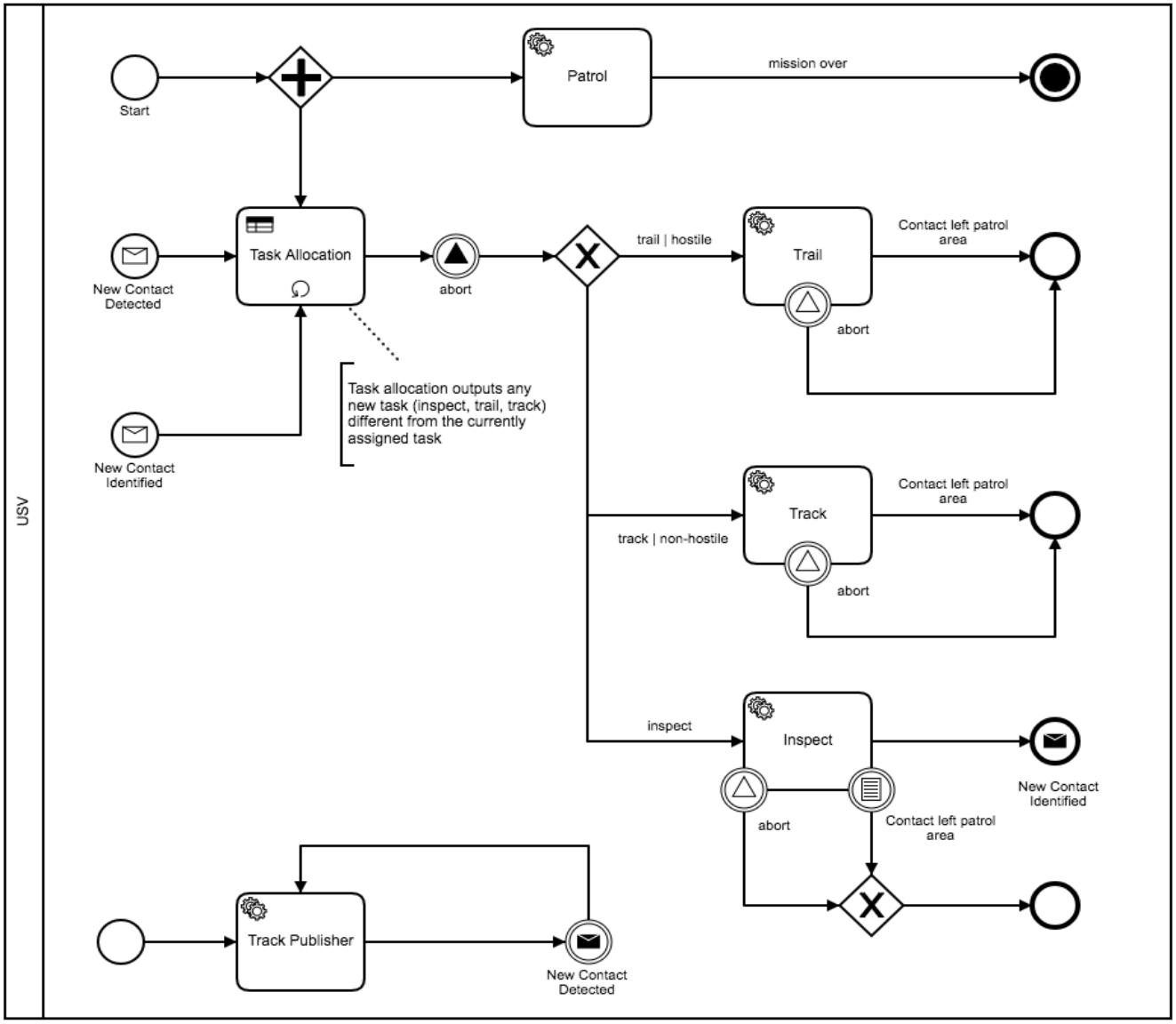



```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <bpmn:definitions xmlns:bpmn="http://www.omg.org/spec/BPMN/20100524/MODEL" xmlns:bpmndi="http://www.omg.org/spec/BPMN/20100524/DI" xmlns:di="http://www.omg.org/spec/DD/20100524/DI" xmlns:dc="http://www.omg.org/spec/DD/20100524/DC" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:camunda="http://camunda.org/schema/1.0/bpmn" id="Definitions_1" targetNamespace="http://bpmn.io/schema/bpmn" exporter="Camunda Modeler" exporterVersion="1.2.2">
3.    <bpmn:process id="Process_1" isExecutable="false">
4.      <bpmn:task id="Task_1qh8xc8" name="Rally 1">
5.        <bpmn:extensionElements>
6.          <camunda:properties>
7.            <camunda:property name="waypoint" value="[lat, lon]" />
8.            <camunda:property name="approach_speed" value="(default:  0.7*V_max)" />
9.            <camunda:property name="capture_radius" value="(default:  2Km)" />
10.         </camunda:properties>
11.       </bpmn:extensionElements>
12.       <bpmn:incoming>SequenceFlow_0l647uu</bpmn:incoming>
13.     </bpmn:task>
14.     <bpmn:sequenceFlow id="SequenceFlow_0l647uu" sourceRef="StartEvent_0430kc2" targetRef="Task_1qh8xc8" />
15.     <bpmn:task id="Task_1jb8zl7" name="Swarm Transit 1">
16.       <bpmn:extensionElements>
17.         <camunda:properties>
18.           <camunda:property name="waypoint" />
19.           <camunda:property name="speed" />
20.           <camunda:property name="formation_type" />
21.           <camunda:property name="formation_scale" />
22.         </camunda:properties>
23.       </bpmn:extensionElements>
24.       <bpmn:incoming>SequenceFlow_1o12s9v</bpmn:incoming>
25.       <bpmn:outgoing>SequenceFlow_131ezto</bpmn:outgoing>
26.     </bpmn:task>
27.     <bpmn:sequenceFlow id="SequenceFlow_1o12s9v" sourceRef="BoundaryEvent_017f9od" targetRef="Task_1jb8zl7" />
28.     <bpmn:sequenceFlow id="SequenceFlow_131ezto" sourceRef="Task_1jb8zl7" targetRef="CallActivity_0oyap06" />
29.     <bpmn:callActivity id="CallActivity_0oyap06" name="Swam Transit Racetrack">
30.       <bpmn:incoming>SequenceFlow_131ezto</bpmn:incoming>
31.       <bpmn:outgoing>SequenceFlow_03yh811</bpmn:outgoing>
32.     </bpmn:callActivity>
33.     <bpmn:subProcess id="SubProcess_0nrbbhh" name="Swarm Transit Racetrack">
34.       <bpmn:startEvent id="StartEvent_08jhtzh">
35.         <bpmn:outgoing>SequenceFlow_1tlpfxh</bpmn:outgoing>
36.       </bpmn:startEvent>
37.       <bpmn:task id="Task_0v5zecc" name="Racetrack Waypoint 1">
```

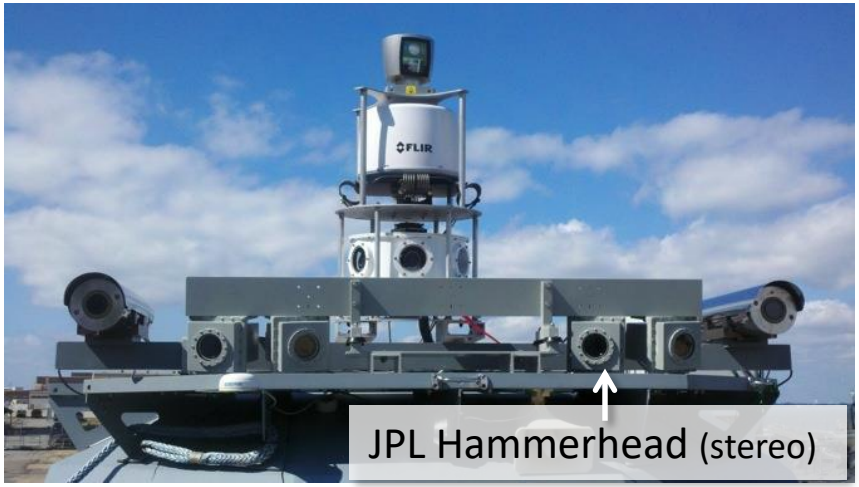
SPAWAR
Systems Center PACIFIC

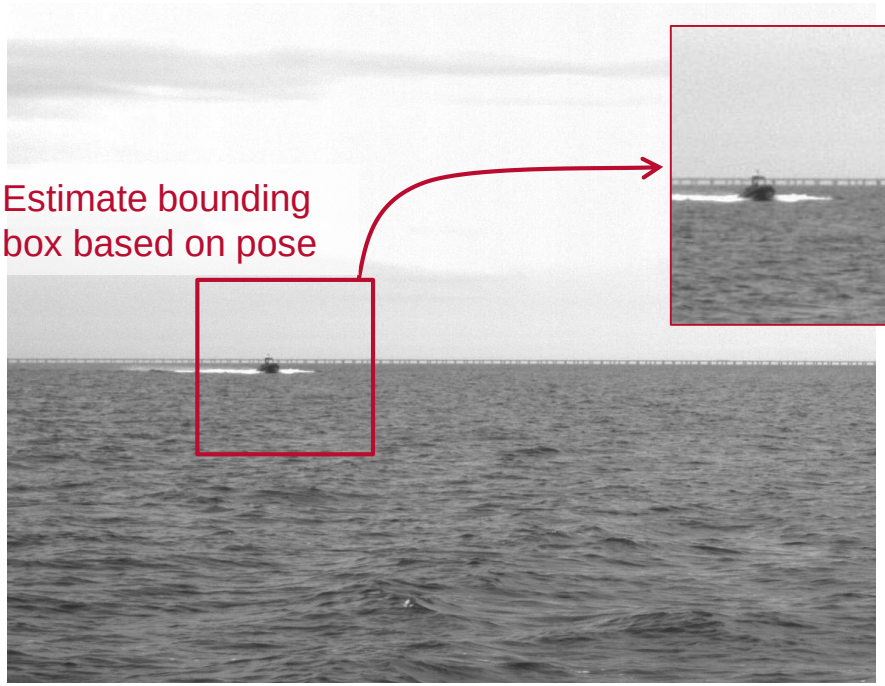
Model Verification
Interpretation by Autonomy
Mission Status Updates

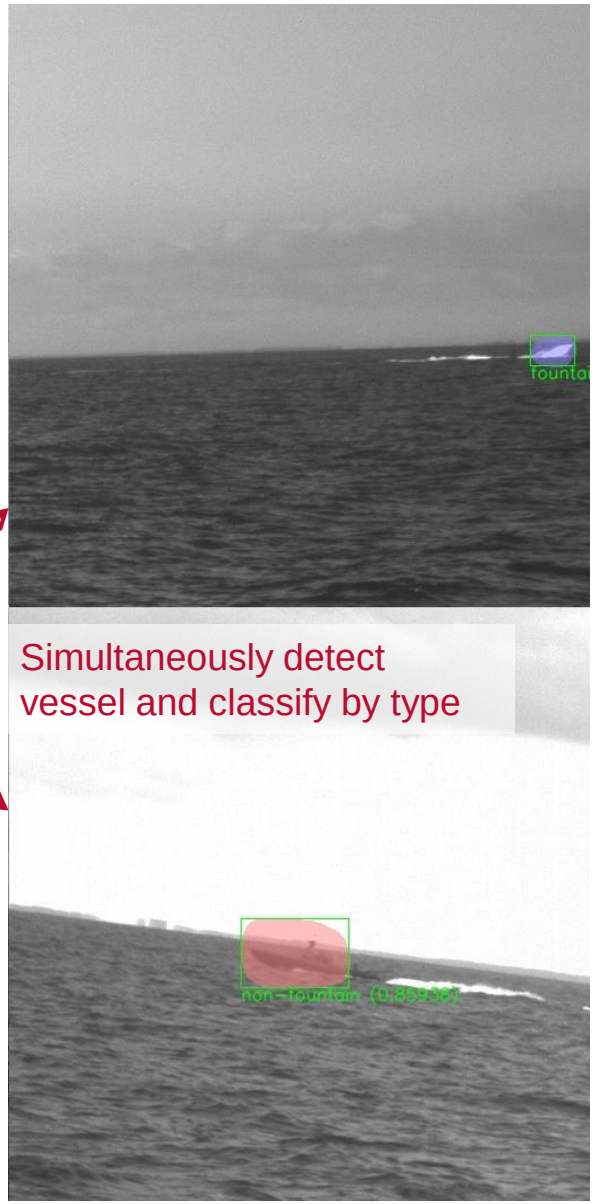
Swarm 2 Mission Model Diagram



CDAS EO Processing (Contact Detection and Analysis System)







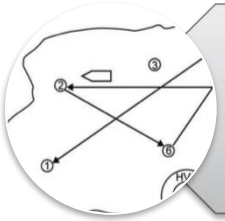
Conclusion

- Summary
  - CARACaS provides an autonomy software framework for multi-agent coordination and control.
  - The Swarm 2 mission demonstrated performance in dynamic, event-driven missions requiring multiple behaviors and shared world modeling.
  - Autonomy has been designed for user-friendliness, robustness to failures, and flexibility for operational variables.
- Next Steps
  - Maturing situational awareness / robustness to perception noise
  - Further demonstrating “building block” approach to missions through behavior composition
  - More complex teaming scenarios (e.g., USV subgroups with closely coupled behaviors)

BACKUP

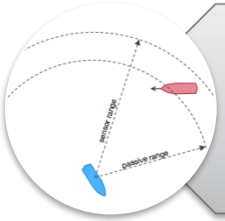
Example Mission Behaviors

Harbor defense: USVs cooperatively patrol harbor, classify unknown contacts, and respond to hostile contacts.



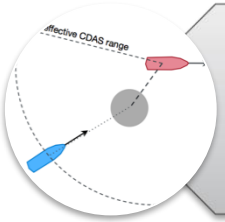
Patrol

- Search area of interest for COI



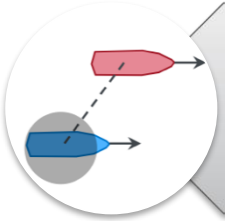
Track

- Keep COI in sensor range



Classify

- Bring COI in CDAS range and classify it



Trail

- Follow COI with prescribed standoff

Swarm Performance Test (SPT) — Metrics

	# COI	COI Type	COI Order	Simulated COI Noise	COI Speed	COI CPA	COI Separation	COI path	Health Fault
Variations	1	Simulated	Simultaneous	Low	Low (5kts)	None	3 km	Orbit	No
	2	Real	One by one	High	High (10Kts)	1 km	1.5 km	Transit	Yes
	3					300m	300m	Diagonal	
								Sinusoid	

How should we measure performance of cooperative autonomy in a general way? (across multiple missions)

Behavior	Metrics
Task Recognition and Allocation	1) Correctness: a. new tasks are recognized b. correct USV is assigned.
	2) Convergence of allocation: time from WM update to showing a task in USV's plan.
	3) Timeliness of allocation: time from WM update to beginning of execution of a task
	4) Swarm plan agreement: % time USV plans are similar (all CASPER solutions have the same Swarm assignments).

Behavior	Metrics
Intercept	Time to intercept (from receiving the command to reaching the desired range to COI / starting the subsequent Trail or Classify behavior)
Patrol	Max time between consecutive visit of any waypoint during a trial.
Classify	Percentage of Classify attempts that result in (successful CDAS WM update -or- maintaining COI in the camera FOV for 50+% of the time between CDAS cue and CDAS failure)
Trail	Range error: RMS of the difference between the desired and actual distance between the USV and COI over the duration of the Trail execution (after intercept)
Track	Percentage of true COIs that have a track maintained (alive in the CSAP) for the full duration (starting at detection+TBD margin) while in Op Area.

Approach: Autonomy Architecture