High Assurance for Distributed Cyber Physical Systems

Architecting Self-Managing Distributed Systems (ECSA/ASDS) Workshop

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Scott Hissam
September 7, 2015

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM-0002714

DART Driving Vision

DARTs coordinate physical agents in an uncertain and changing physical world.

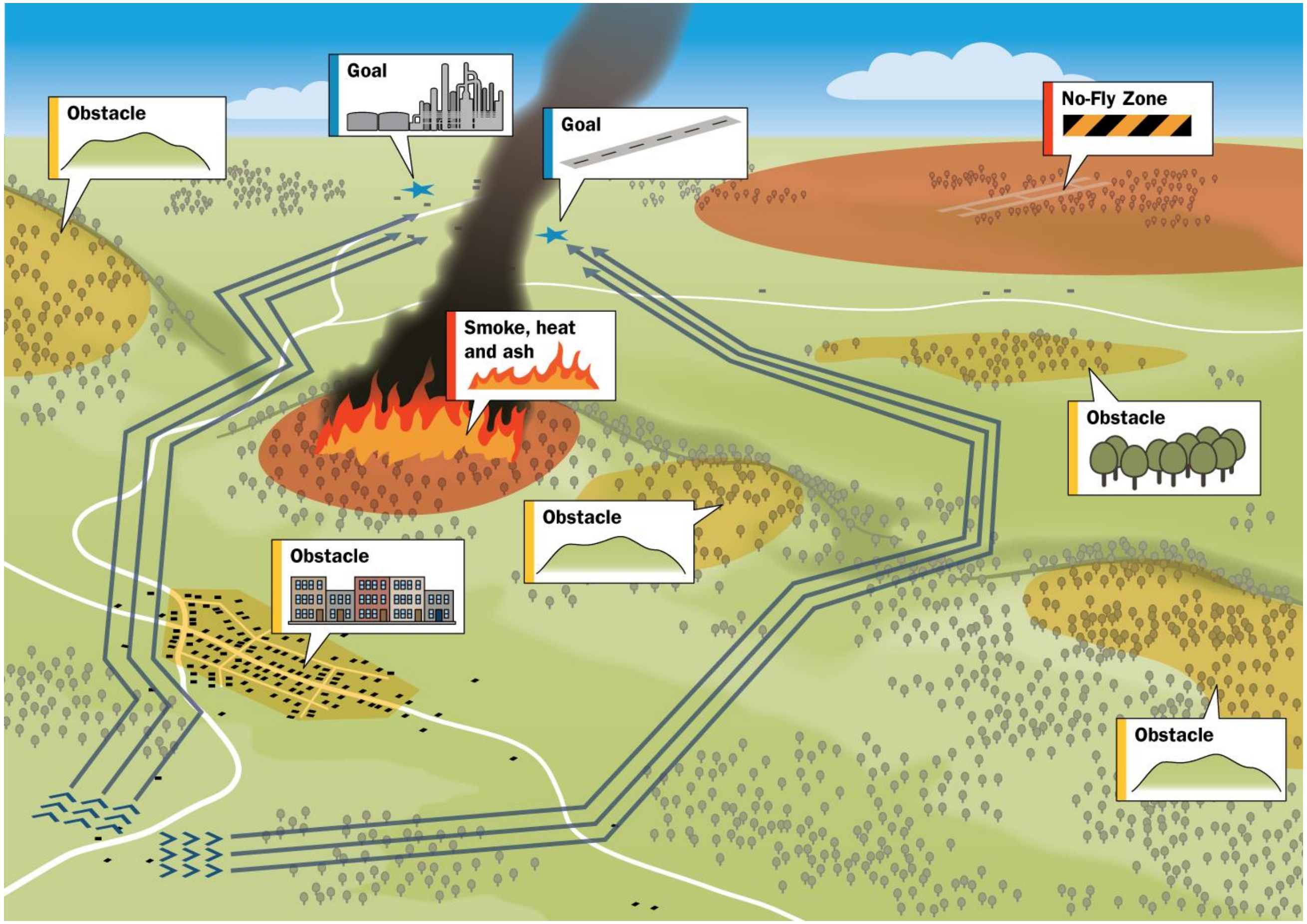
- Coordination – physical agents
- Timeliness – safety critical
- Resource constrained - UAVs
- Sensor rich – sensing physical world
- Intimate cyber physical interactions
- Automated adaptation to physical context
- Computationally complex decisions

Coordination, adaptation, and uncertainty pose key challenges for assuring safety and mission critical behavior of distributed cyber-physical systems.

The DART project uses develops and packages sound techniques and tools for engineering high-assurance distributed CPS.

*** DART = Distributed Adaptive Real-Time**

Our Current Unified Motivating Scenario



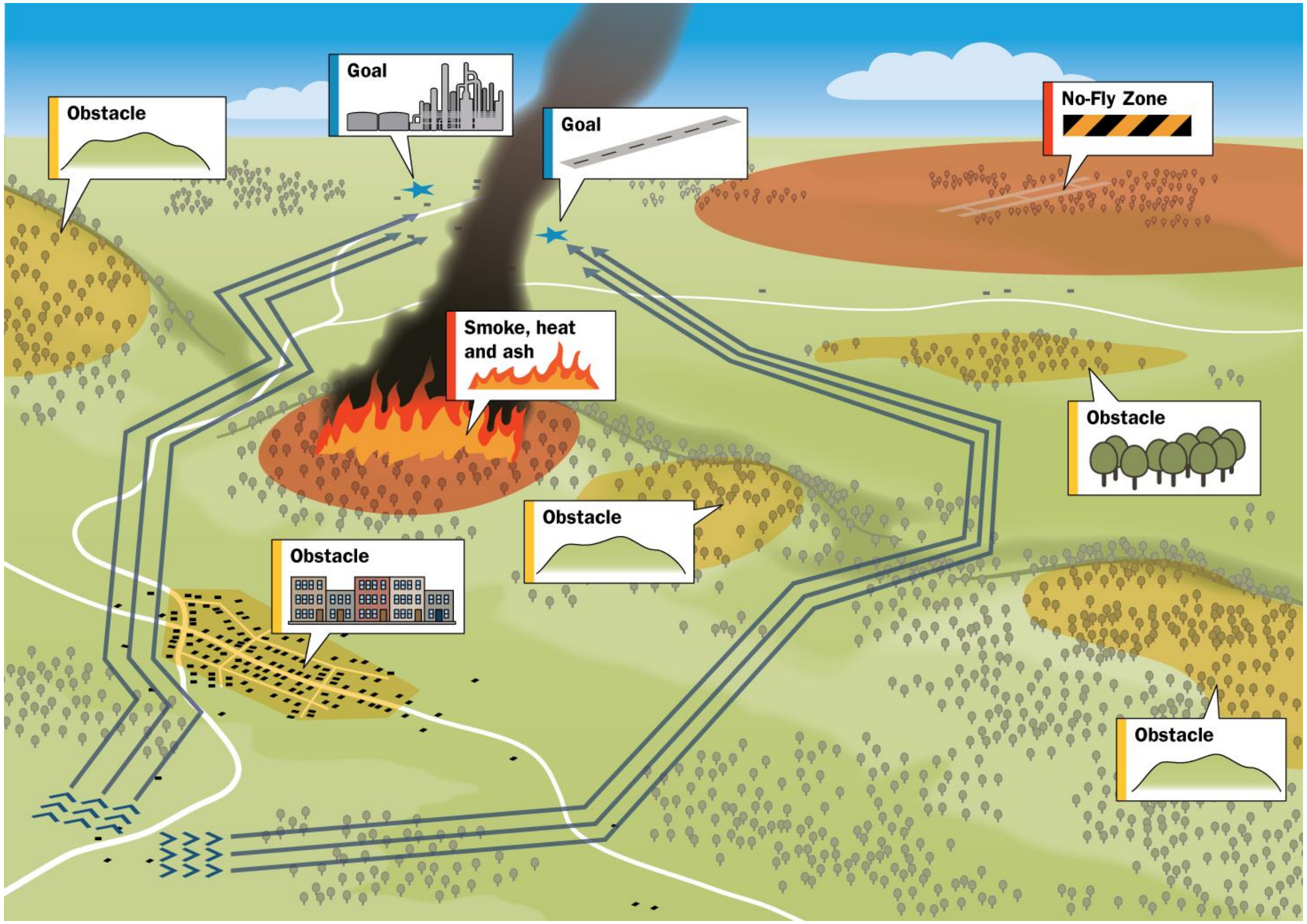
Objectives

- Search and rescue for groups of UAVs
- Protection among important assets by groups of UAVs

Threat Models, Challenges and Features

- Disrupted Communications
- Obstacles
- Emergent threats
- Data, Knowledge and Processing Overload

Application of Research to Motivating Scenario



Design Time Verification

- Guaranteed behavior
- Best-effort behavior

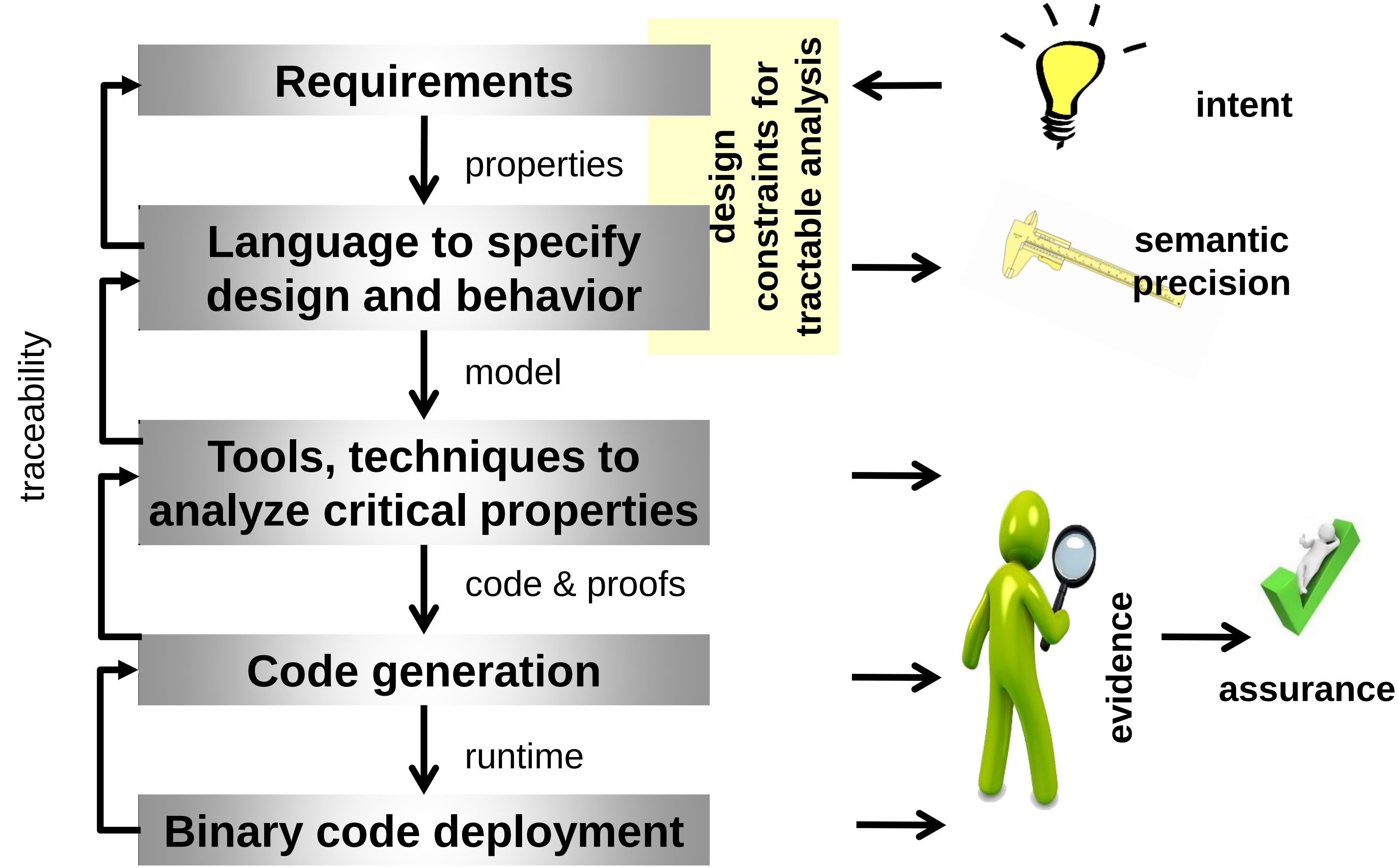
Autonomy Implementation

- Real-time reasoning, timing and control
- Networking
- Quality-of-service

Runtime Assurance

- Critical Timing behavior
- Coordination
- Adaptation

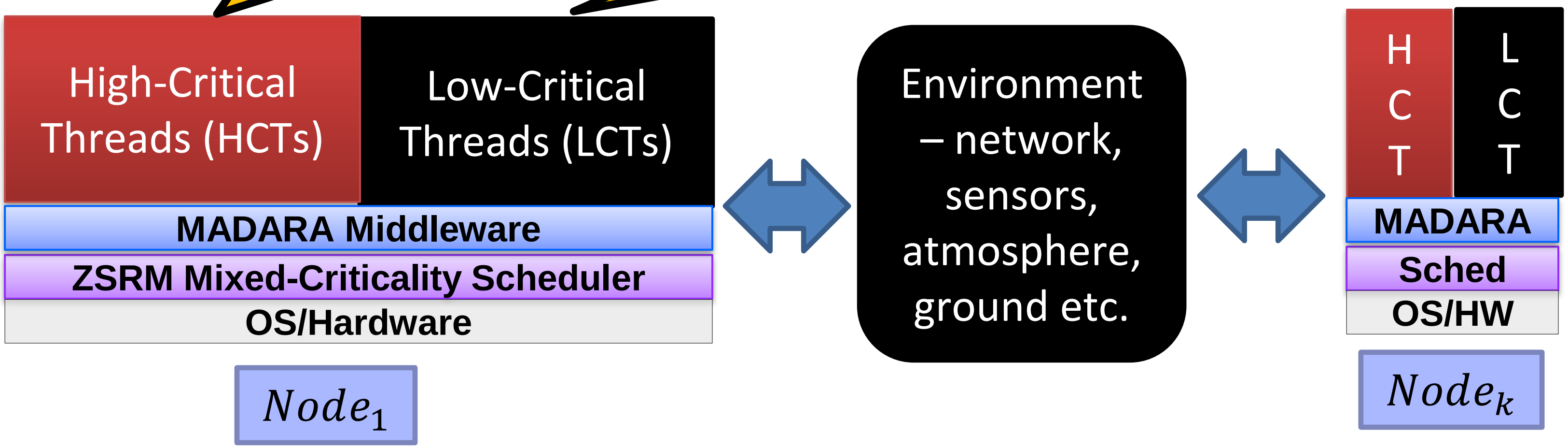
Engineering High Assurance for DART



DART High-Level Architecture

Software for guaranteed requirements, e.g., collision avoidance protocol must ensure absence of collisions

Software for probabilistic requirements, e.g., adaptive path-planner to maximize area coverage within deadline



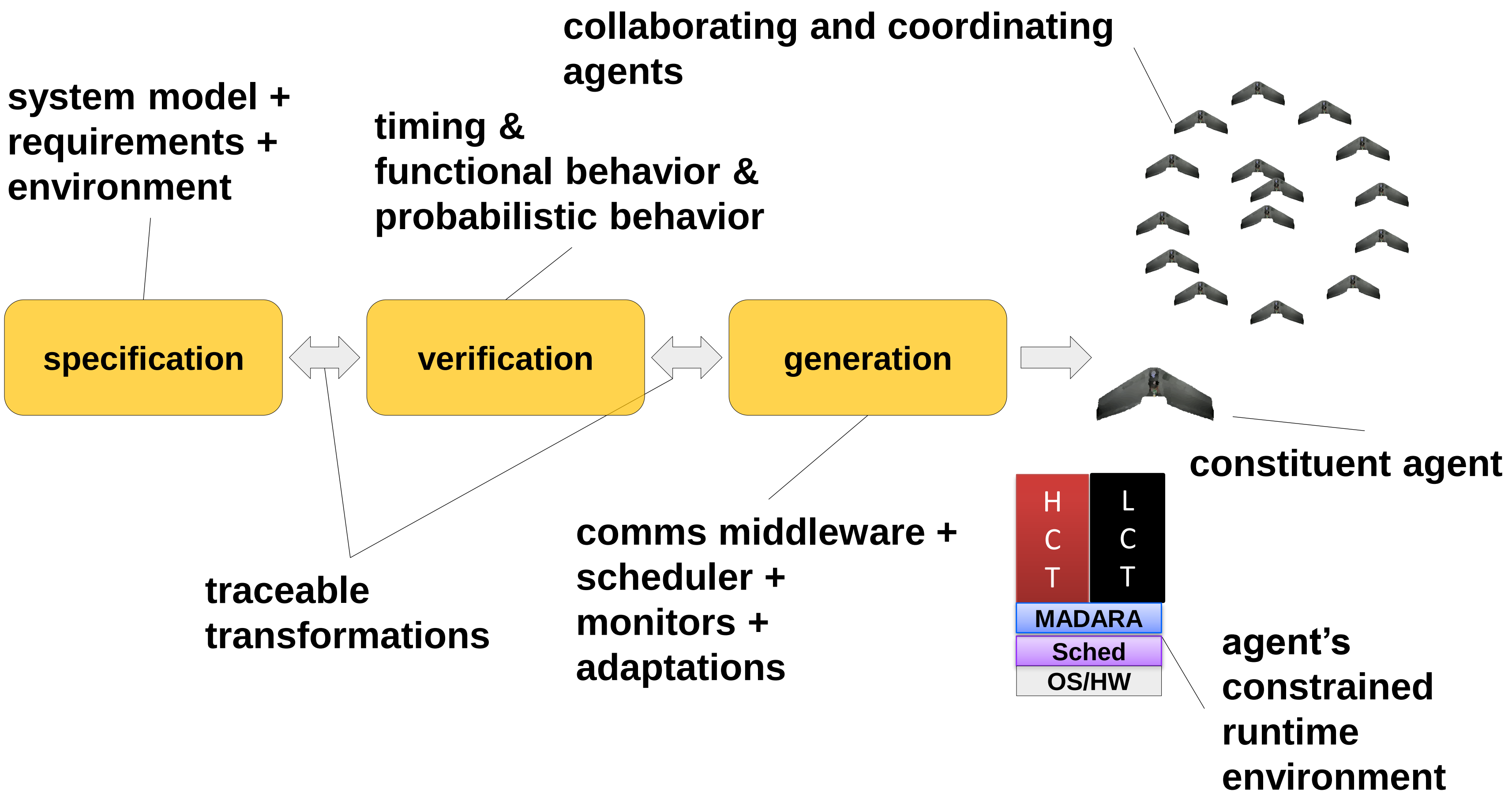
Research Thrusts

- **Proactive Self-Adaptation**
- **Statistical Model Checking**
- **Real-Time Schedulability**
- **Functional Verification**

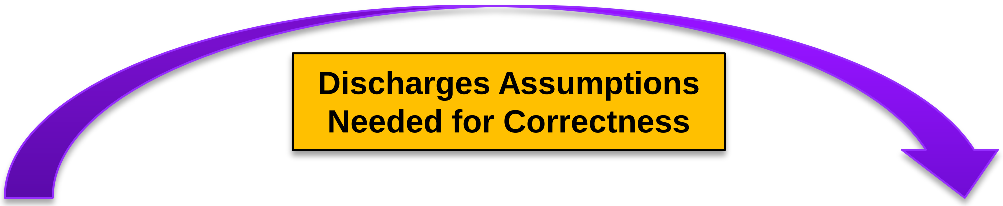
Validation Thrusts

- **Model Problem**
- **Workbench**

DART Tooling and Techniques



Research - Deterministic Behavior



Real-Time Schedulability

- Technique to guarantee deadlines among tasks with different semantic criticalities in a rate-monotonic scheduler.

Challenges:

- Mixed-Criticality Scheduling under I/O
- End-to-end Mixed-Criticality Scheduling

Functional Verification

- Technique to ensure the behavior of a distributed application that satisfies a user-specified safety specification.

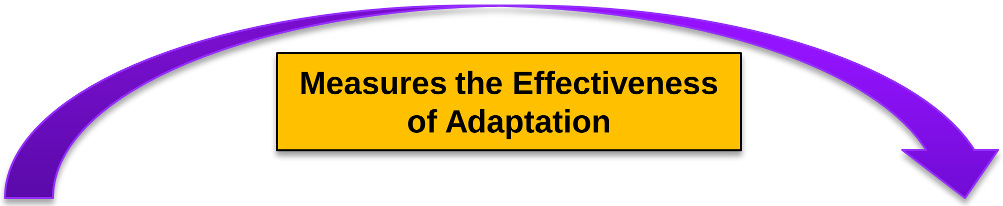
Challenges:

- Unbounded Model Checking Synchronous Software
- Unbounded Model Checking Asynchronous Software

de Niz, D.; Lakshmanan, K.; Rajkumar, R., "On the Scheduling of Mixed-Criticality Real-Time Task Sets," Real-Time Systems Symposium, 2009, RTSS 2009. 30th IEEE, vol., no., pp.291-300, Dec. 2009.

Chaki, S., Edmondson, J., “Model-Driven Verifying Compilation of Synchronous Distributed Applications,” Model-Driven Engineering Languages and Systems, Springer, LNCS, v.8767, pp. 201-217, Oct. 2014.

Research - Probabilistic Behavior



Statistical Model Checking

- Technique to compute the bounded probability that a specific event occurs during a stochastic system's execution.

Challenges:

- Importance Sampling with Heterogenous Fault Regions
- Statistical Model Checking for systems with non-determinism

Proactive Self-Adaptation

- Technique for a system to adapt to an upcoming situation given that time is needed to perform the adaptation.

Challenges:

- Adaptation decision under uncertainty
- Integration with Machine Learning Techniques

Hansen, J.P., Wrage, L., Chaki, S., de Niz, D., Klein, M., "Semantic Importance Sampling for Statistical Model Checking," in press, Tools and Algorithms for the Construction and Analysis of Systems (TACAS), Springer, LNCS, Apr. 2015.

Cámara, J., Moreno, G.A., Garlan, D., "Stochastic Game Analysis and Latency Awareness for Proactive Self-adaptation," 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS). ACM, New York, NY, pp. 155-164. May 2014.

Validation Thrusts

Drives Implementation

DART Model Problem

- High assurance multi-agent DART prototype integrating deterministic and probabilistic verification techniques.

Challenges:

- Quantifying mission impact
- Spanning the gap between "the lab" and real world

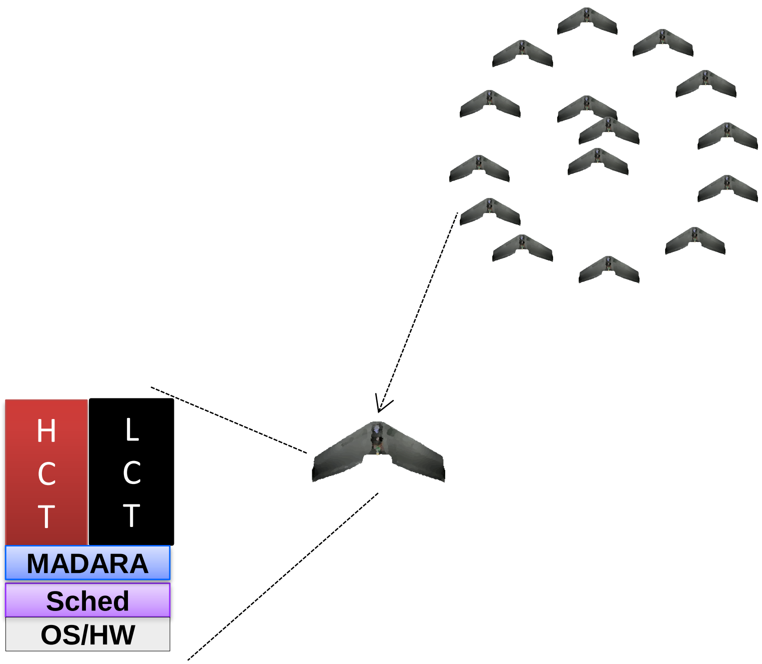
DART Workbench

- Create an integrated engineering approach for developing DART systems through specifications and tooling.

Challenges:

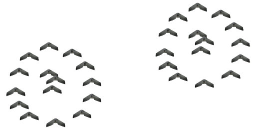
- Semantically precise specifications
- end-to-end traceability

DART Model Problem – Elements of Analysis



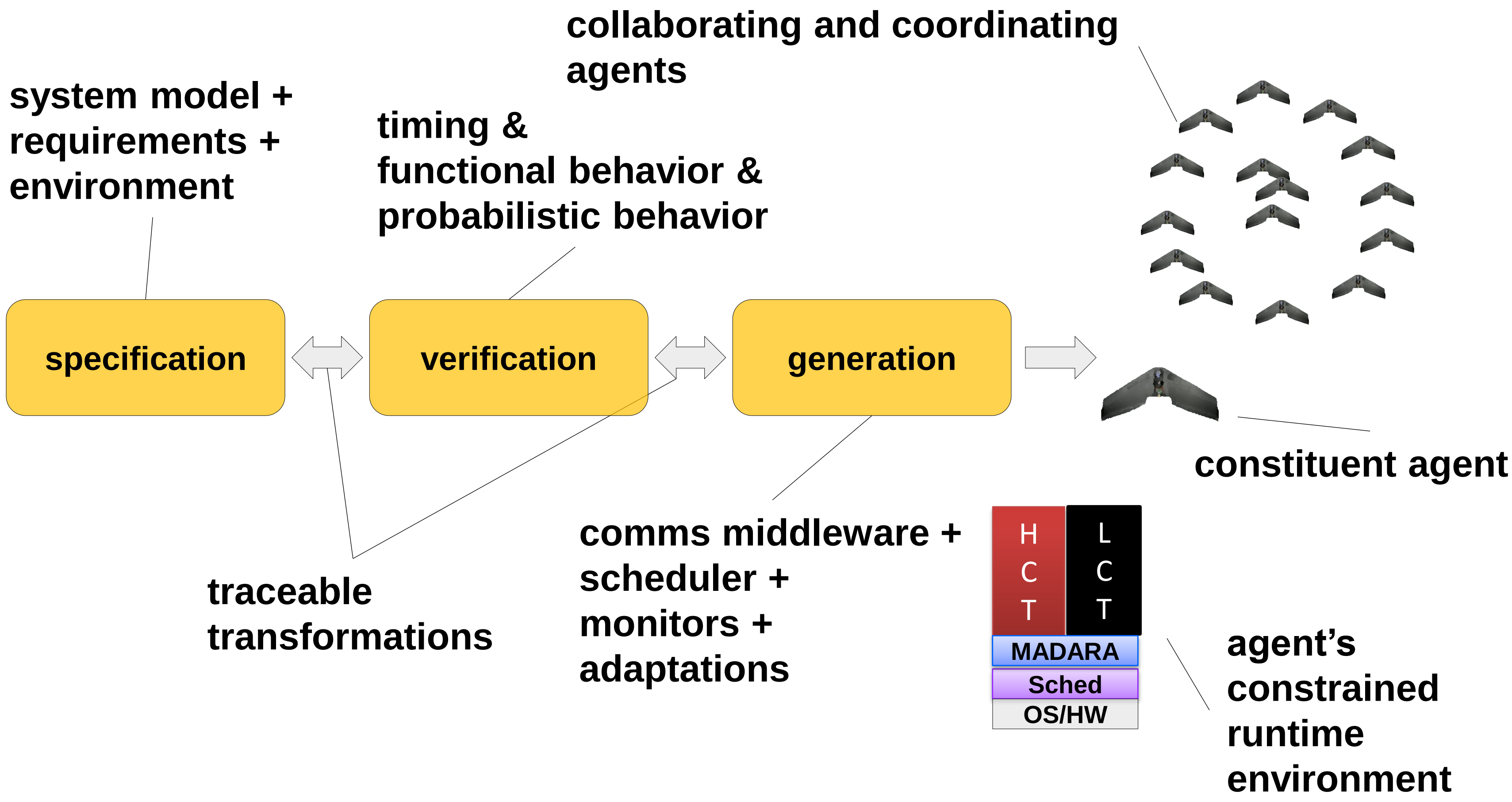
- Guarantee timing behavior of highly critical threads through *mixed-criticality* temporal protection mechanisms

- Guaranteed separation among multiple agents (MA) through compositional model checking
- Best-effort confidence in adherence to MA formation through statistical model checking
- Increased survivability of MA system from threats through proactive self-adaptation
- Coordination of sensor functions among MA with end-to-end latency (fy16)

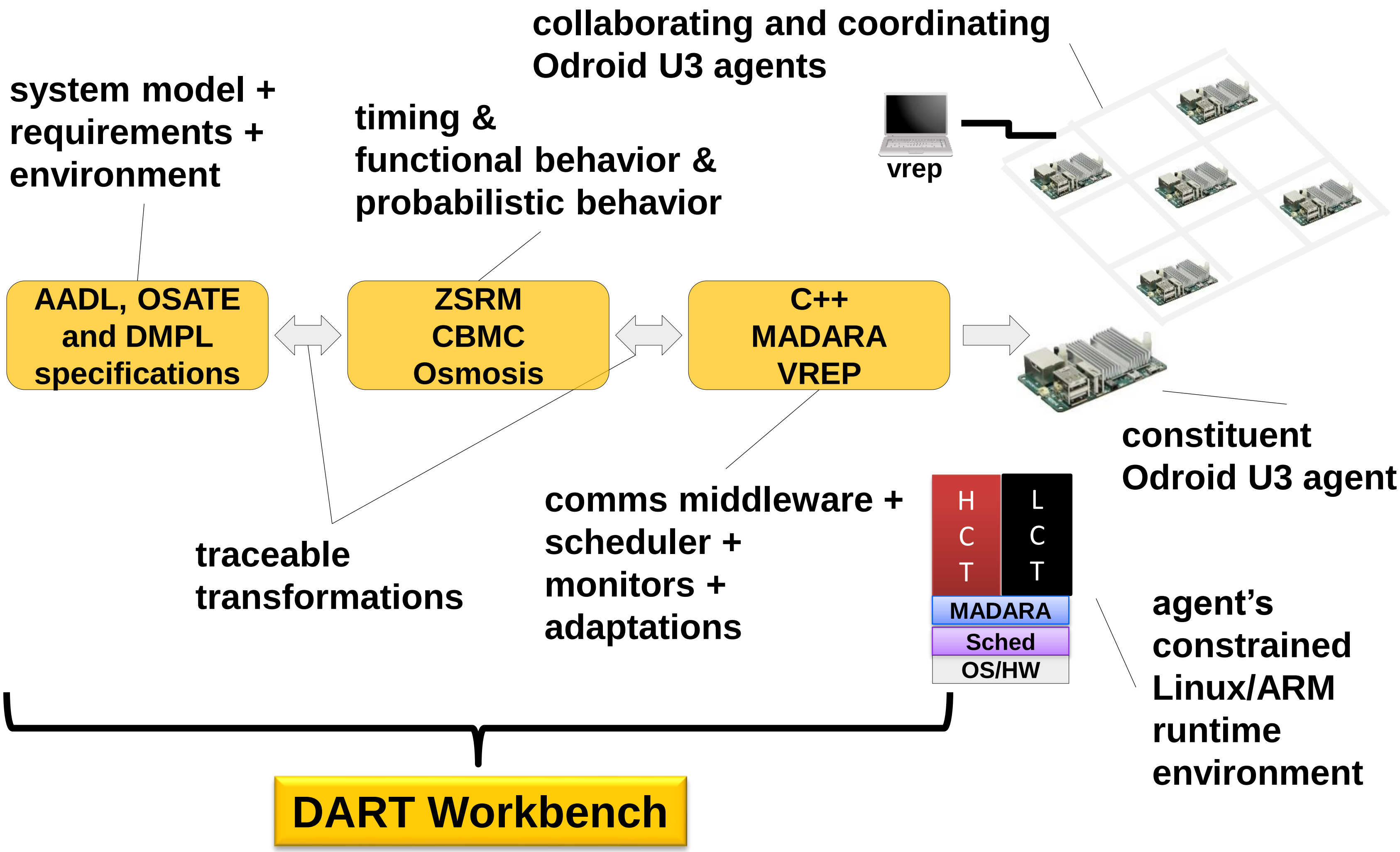


- Coordination of mission objectives between MA systems (fy16)

DART Workbench Overview



DART Workbench Details



DART Workbench Usage

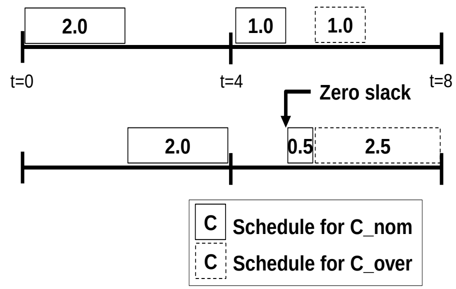
Node Specification in a DSL

```
@HERTZ(8)
@CRITICALITY(HIGH)
@WCET_NOMINAL(2.5)
@WCET_OVERLOAD(5.0)
@BARRIER_SYNC
...
void collision_avoid() {
  // Operates on X & Y
}
...
require(FORALL_NODE_PAIR
  (id1, id2,
   x@id1 != x@id2 ||
   y@id1 != y@id2));

require(InBounds(X,Y);
...
@AT_LEAST(0.8)
expect(COVER() >= 0.9)
else {
   // Adapt
 };
...
```

Analysis & Verification

ZSRM timing

```
T: <other>
Period: 4
C_nom:  2.0
C_over: 2.0
Crit:  Low
Pri:   High

T: coll_av
Period: 8
C_nom:  2.5
C_over: 5.0
Crit:  High
Pri:   Low
```



CBMC model

- read shared context
- `ASSUME` (local constraints)
- do collision_avoid()
- `ASSERT` (local changes)
- write shared context

VREP model*

- log (COVER() variables)
- Do collision_avoid()

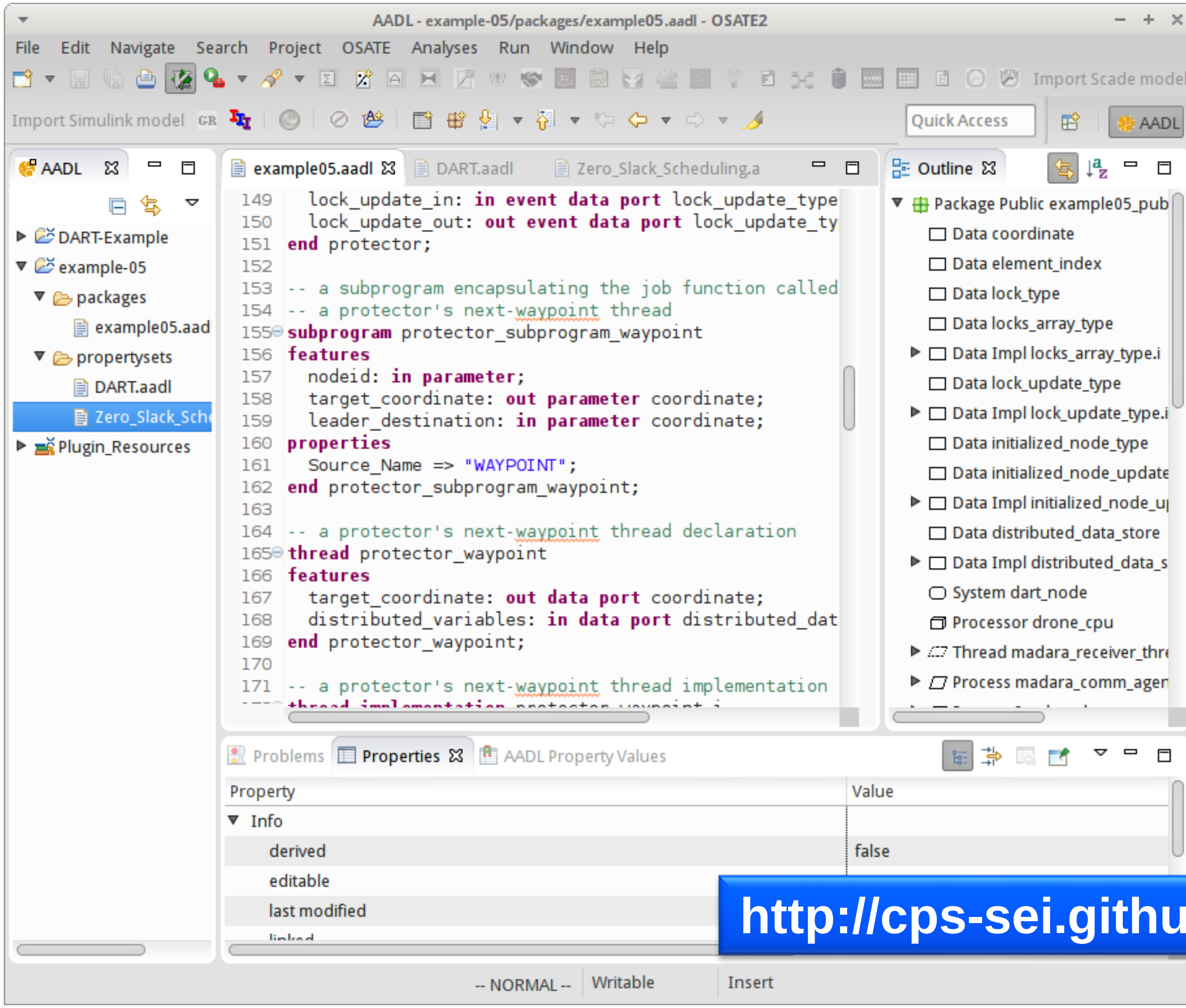
*multiple repetitions

→ Perform offline statistical analysis of logged data

Target Code Gen.

```
attr.period_msec = 125;
attr.Cmon_msec = 2.5;
attr.Cover_msec = 5.0;
attr.criticality = HIGH;
attr.zs_instant_nsec =
   zsinst["coll_avoid"];
zs_reserve(&attr);

int loc_X = ShrRead(X);
int loc_Y = ShrRead(Y);

// Do coll_avoid logic

if(!(InBounds(loc_X,
              loc_Y))
  // Handle Fault

AdaptManager(COVER());

ShrWrite(X).set(loc_X);
ShrWrite(Y).set(loc_Y);
```

DART Workbench Screenshot



http://cps-sei.github.io/dart/

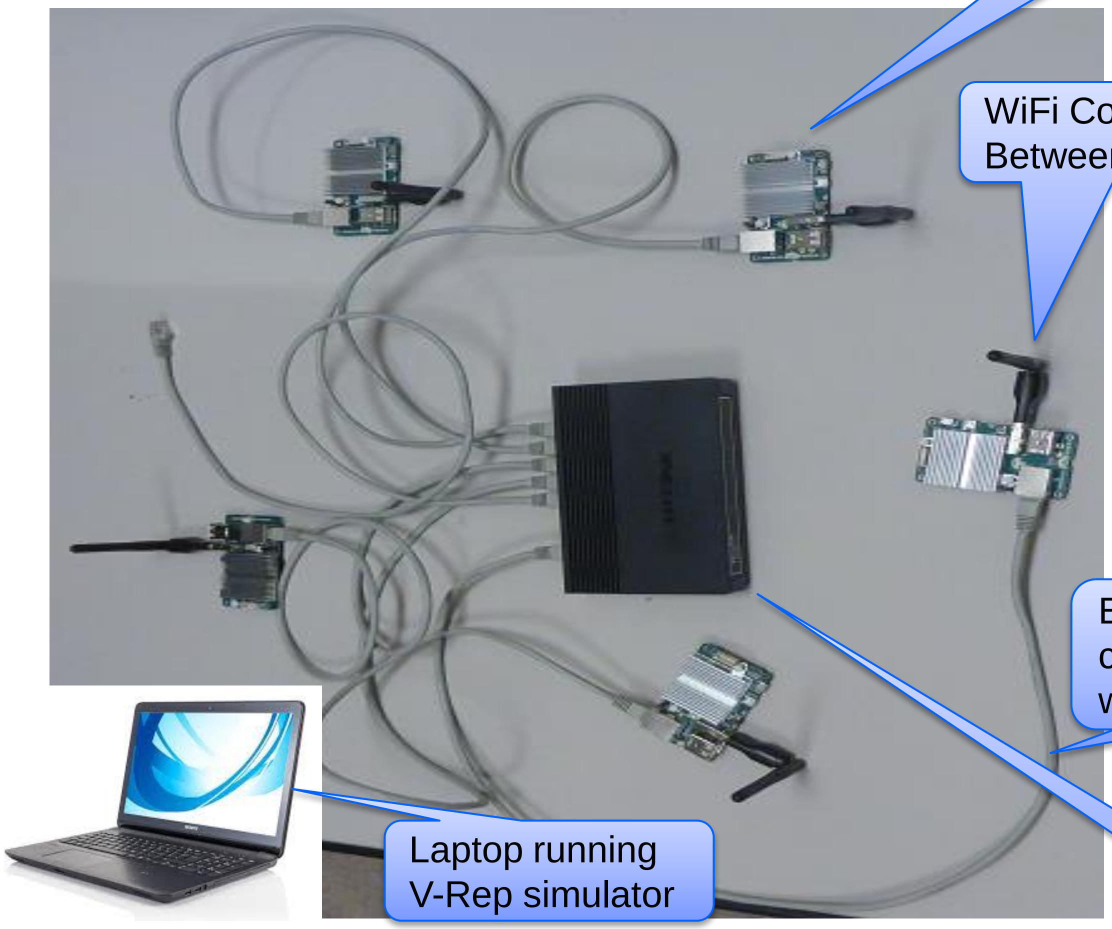
DART Demonstration Infrastructure

Odroid U3s

- Linux-RK/ARM
- ZSRM
- WiFi

Laptop

- VREP
- Logging
- Ethernet



Team

Scott Hissam

Bjorn Andersson

Mark Klein

Bud Hammons

Arie Gurfinkel

Dionisio de Niz

Gabriel Moreno

David Kyle

Jeff Hansen

James Edmondson

Sagar Chaki

Contact Information Slide Format

Presenter / Point of Contact

Scott Hissam

Telephone: +1 412-268-6526

Email: shissam@sei.cmu.edu

U.S. Mail

Software Engineering Institute

Customer Relations

4500 Fifth Avenue

Pittsburgh, PA 15213-2612

USA

Web

www.sei.cmu.edu

www.sei.cmu.edu/contact.cfm

Customer Relations

Email: info@sei.cmu.edu

Telephone: +1 412-268-5800

SEI Phone: +1 412-268-5800

SEI Fax: +1 412-268-6257