

HARVEST



A Framework & Co-simulation Environment for Analyzing Unmanned Aerial Vehicle Swarms

Chris Augeri¹, Kevin Morris², & Barry Mullins¹

1. Air Force Institute of Technology (AFIT), Wright-Patterson AFB, OH
{chris.augeri, barry.mullins}@afit.edu
2. Air Force Communications Agency (AFCA), Scott AFB, IL
kevin.morris-03@scott.af.mil

Overview



- **HARVEST**
 - Conceptual UAV swarm
 - Cross-layer design?
- **JOCosim I (JOC-I)**
 - Java data exchange
 - Single UAV
- **JOCosim II (JOC-II)**
 - Java control
 - Multiple UAVs

HARVEST



Host of Armed Reconnaissance Vehicles

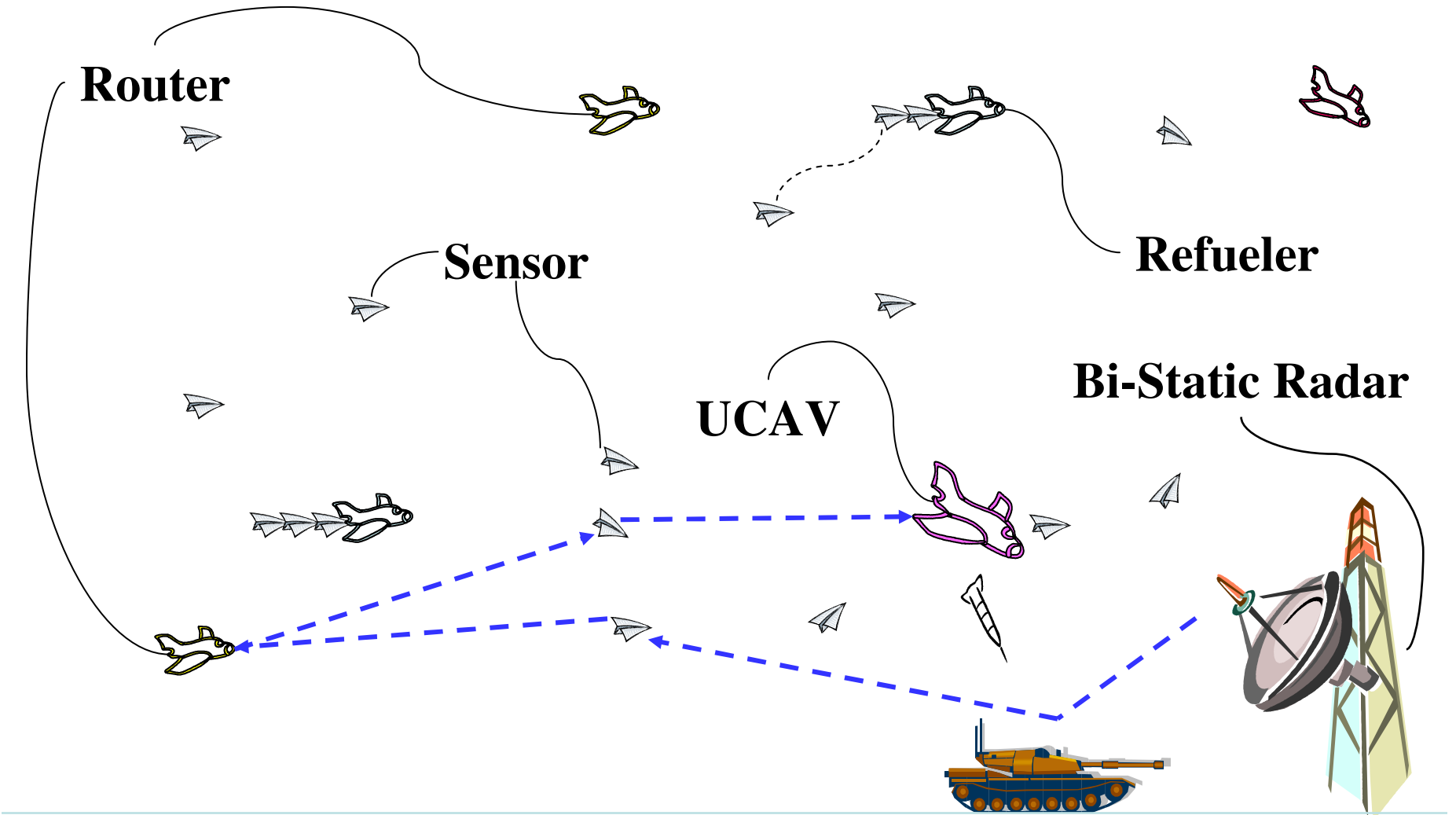
Enableing Surveillance & Targeting

HARVEST: A Prototypical UAV Swarm



- **It *is*:** cooperative UAV swarm concept
- **It *isn't*:** architecture for UAV design (JAUS)
- **Configurations**
 - Micro-UAV for each platoon member
 - Mini-UAV for a security forces element
 - A deployable attack/recon asset
- **Applications**
 - Anti-sniper support
 - CBRN plume/weapon tracking
 - Cooperative search

HARVEST: A Conceptual Deployment



HARVEST: Representative Deployment

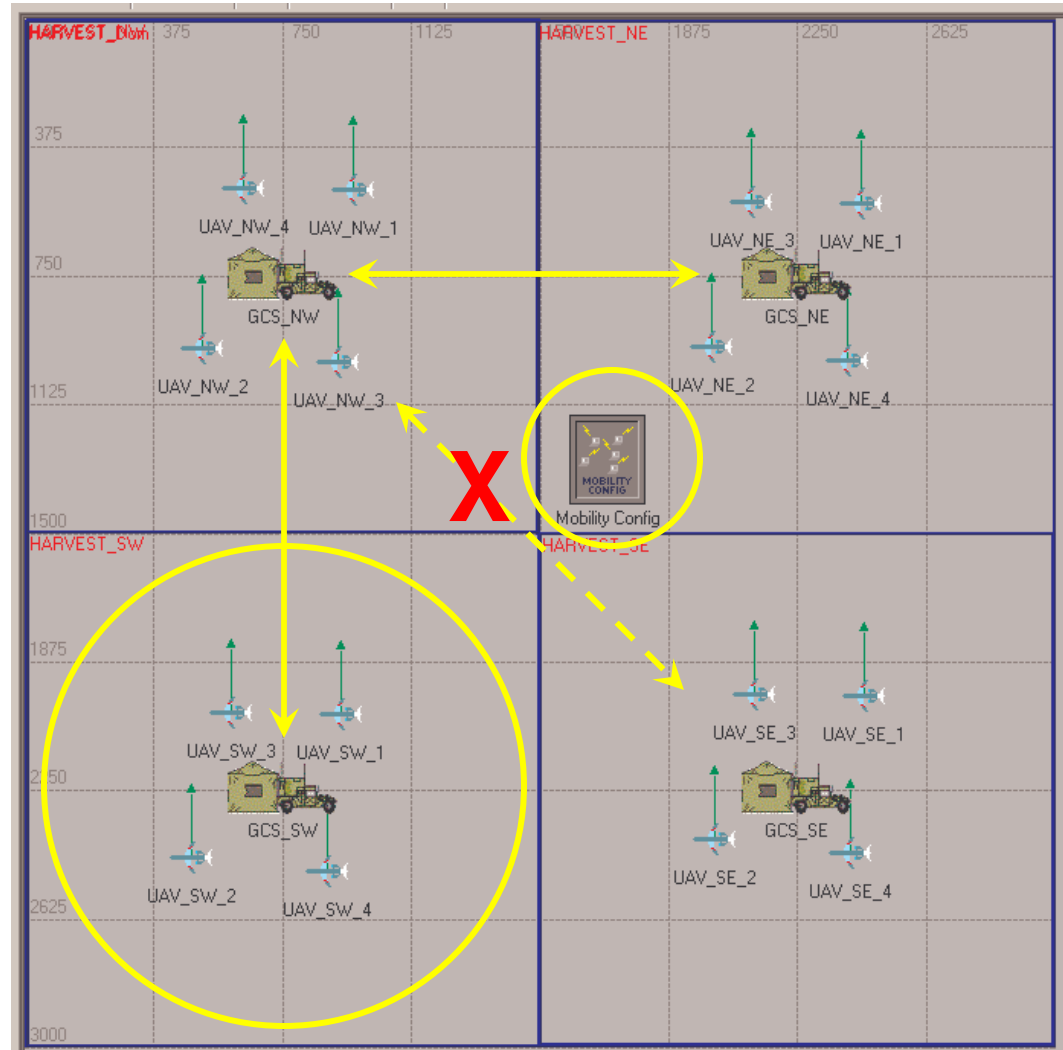


- **Configuration**

- (4) Mobility Domains
- (4) GCSs
- (4) UAVs

- **Operation**

- GCS communicates w/ vert. & horiz. adjacent GCSs, but *not* diagonally adjacent
- UAV assigned to GCS controlling domain
- UAV flies using random waypoint model



HARVEST: Protocol Design



- **Cross-Layer Design**
 - Integrate vertically...
 - Pros: optimization
 - Cons: testing, upgrading
- **Managing Policies¹**
 - Command vertically...
 - Pros: scalable
 - Cons: speed?
 - Suggested policies
 - Preservation
 - Employment

Layers	Services	Policies
Application	User Guidance	Preserve fuel defense
	Swarm Behavior	
	Vehicle Services	
Transport	IPv6	Employ recon target
Network	IPv6	
Link	Time...	

¹I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. "A survey on sensor networks," *IEEE Communications*, vol. 40 (8),2002, pp. 102–114.

HARVEST: Protocol Examples



- **Localization**

- Link-layer timing
- Network & transport layer forward timing
- Application-layer Localization algorithm consumes timing data

- **Power Management**

- Each layer/protocol provides API
- Each protocol adjusts consumption based on UAV status, swarm intel, and user inputs

Layers	Services	Policies
Application	User Guidance	Preserve fuel defense
	Swarm Behavior	
	Vehicle Services	
Transport	IPv6	Employ recon Target
Network	IPv6	
Link	Time...	

HARVEST: Application Layer



Swarm Services

Aerial
Docking

Data
Indexing

Target
Tracking

Cooperative
Search

Vehicle Services

Collision
Avoidance

Payload
Tasking

Telemetry
Reporting

Information
Sensing

User Interaction

Swarm
Management

Flight
Planning

Query
Generation

Munitions
Deployment



JOCosim I

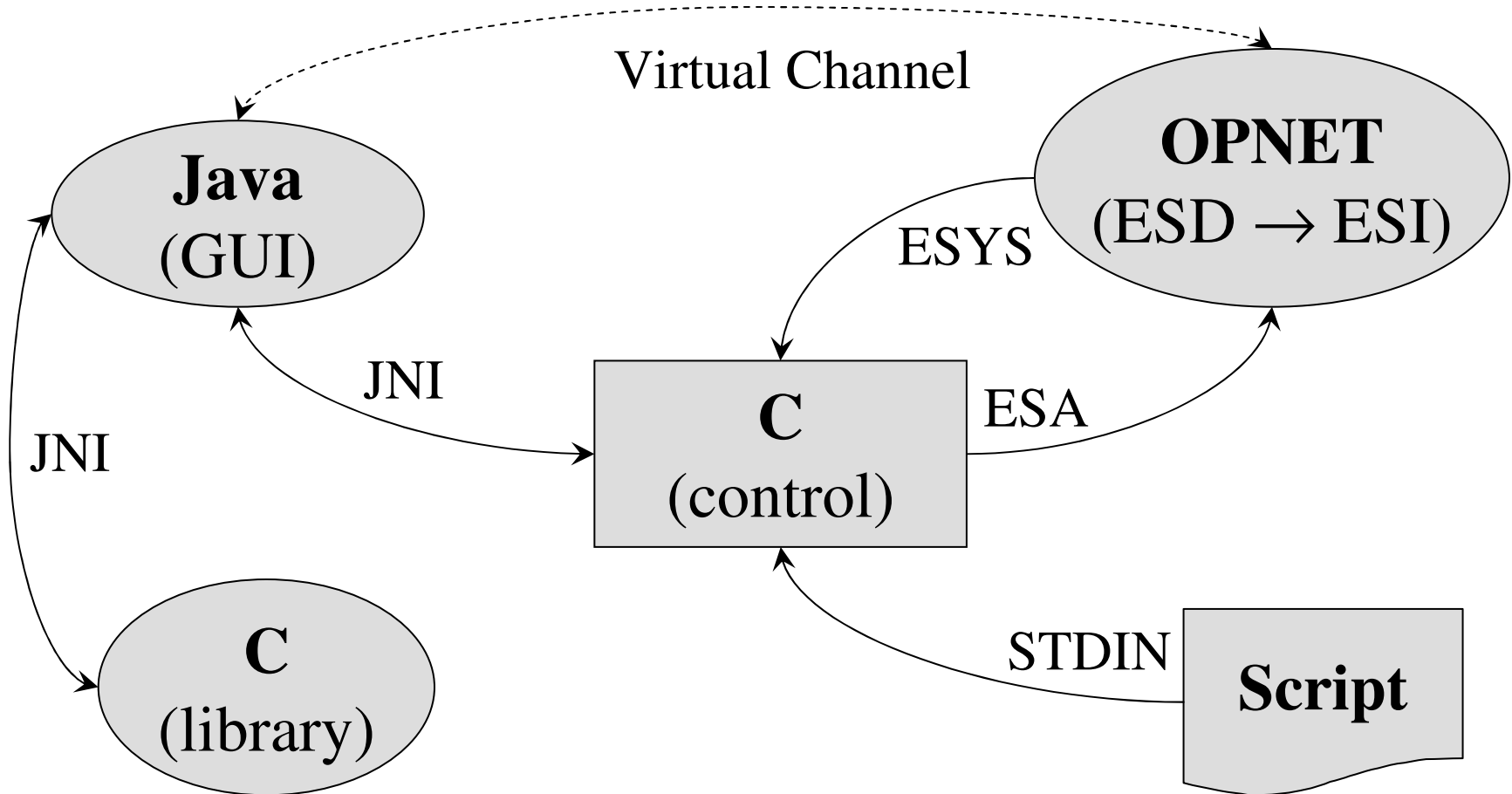
(JOC-I)

JOCosim: A Java, OPNET & C Co-Simulator for a HARVEST



- **What is co-simulation?**
 - Integration of multiple simulation engines
 - Leverages capabilities/investment in multiple simulators
- **Co-simulation goals**
 - Wireless performance: OPNET
 - Application-layer protocols & swarming behavior: Java
- **Co-simulation interfaces?**
 - OPNET: ESI (generic co-simulation interface)
 - OPNET: HLA (from DoD DMSO)
- **Phased development**
 - JOC-I: verify Java can exchange data w/OPNET
 - JOC-II: verify Java can control an OPNET simulation

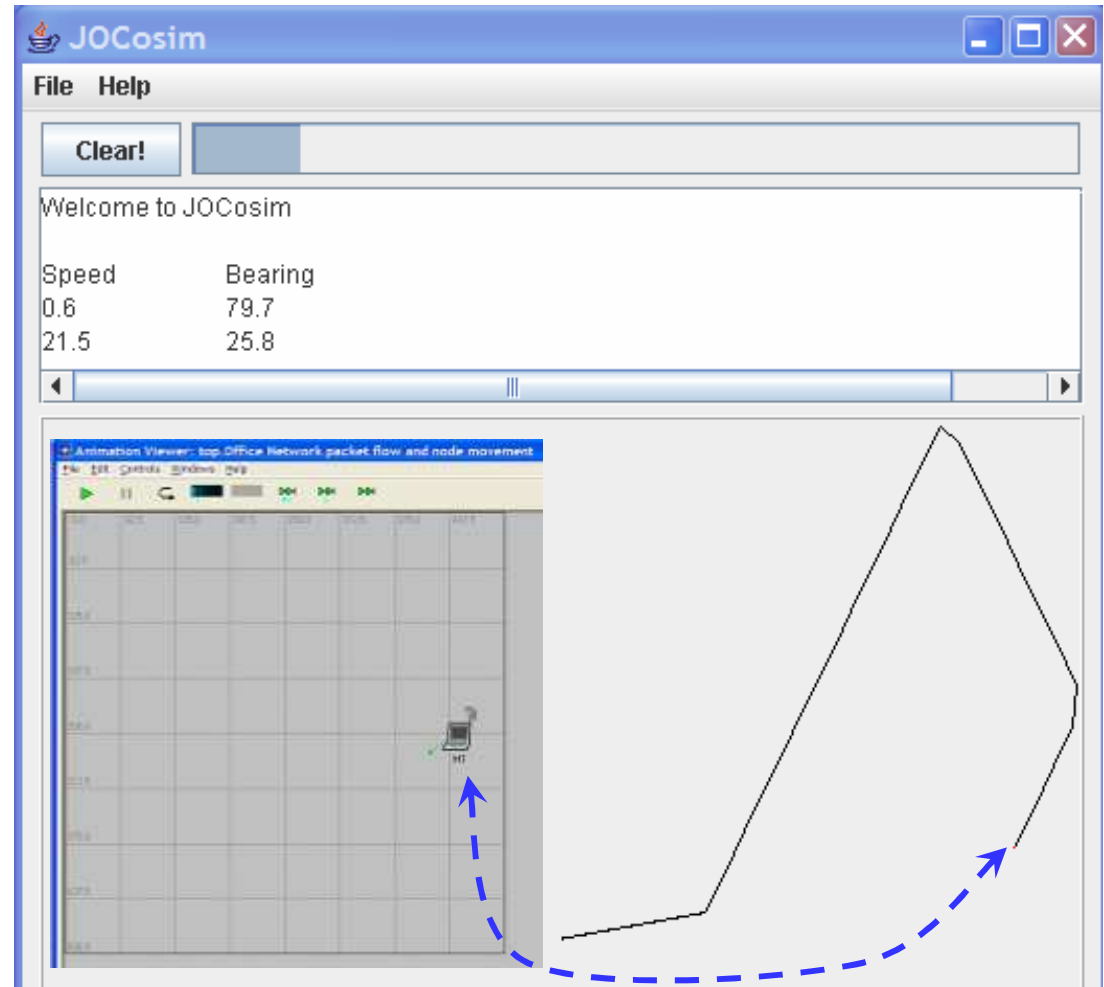
JOC-I: Design Overview



JOC-I: Co-Simulation Results



- **Links Java/OPNET**
 - Script control (C)
 - Uses ESI (C ↔ Java)
- **Single UAV**
 - Position → Java
 - Vector → OPNET
- **GUI**
 - Java: single UAV tracker
 - OPNET: verify in animation viewer





JOCosim II

(JOC-II)

JOC-II: Extensions to JOC-I



1. Move co-simulation control to Java
2. Develop interface cloud for ESYS access
3. Create models w/external model access (EMA)
4. Enable broadcast & routed transmissions (modes)
5. Support node failure and recovery (via power mgmt)
6. *Active statistic collections via probes (global)
7. *Port a node mobility algorithm to Java
8. *Integrate a data indexing protocol
9. *Migrate from OPNET 10.5 to 11.5

* partially completed extensions

JOC-II: Simulation Creation



- User interface to create a simulation
- GUI tailored for cooperative search
- Network created with JOCnet (EMA)
 - Network size
 - Search grid
 - UAVs & cloud
 - ESD

Create simulation...

Basic | Advanced

Model Name: JOC_5_100_100

Scenario Name: JOC_5_100_100_10_10_600_3

Observers: 5

Network Width (m): 100

Network Height (m): 100

Cell width (m): 10

Cell Height (m): 10

Time (sec): 600

Random Seed: 3

Save Animation

View Animation

Probe Model: Global

Use Defaults

Create Scenario!

Run Scenario!

JOC-II: Animation Viewer



- **Interface cloud**

- **UAVs**

- Active

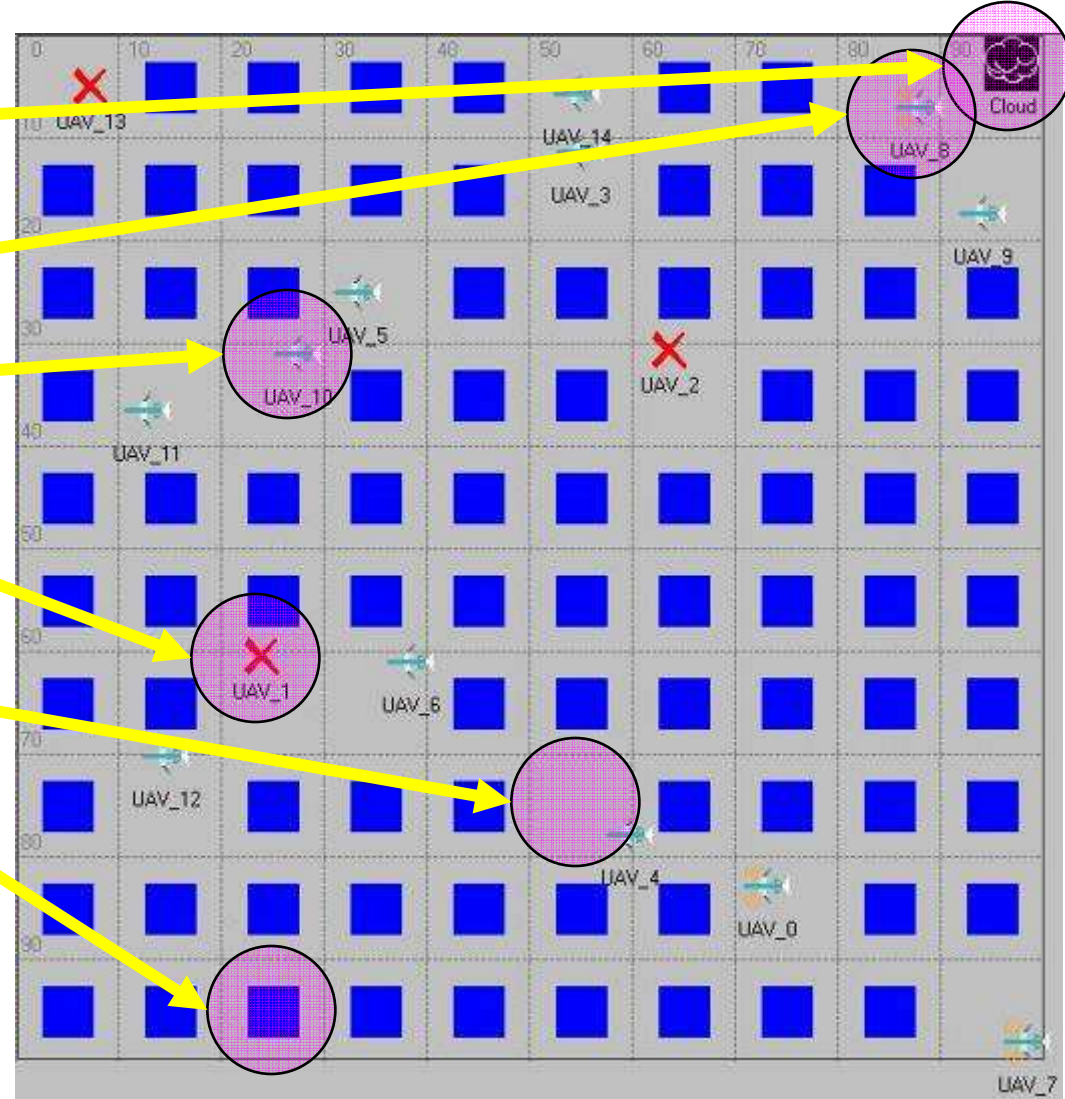
- Reserve

- Destroyed

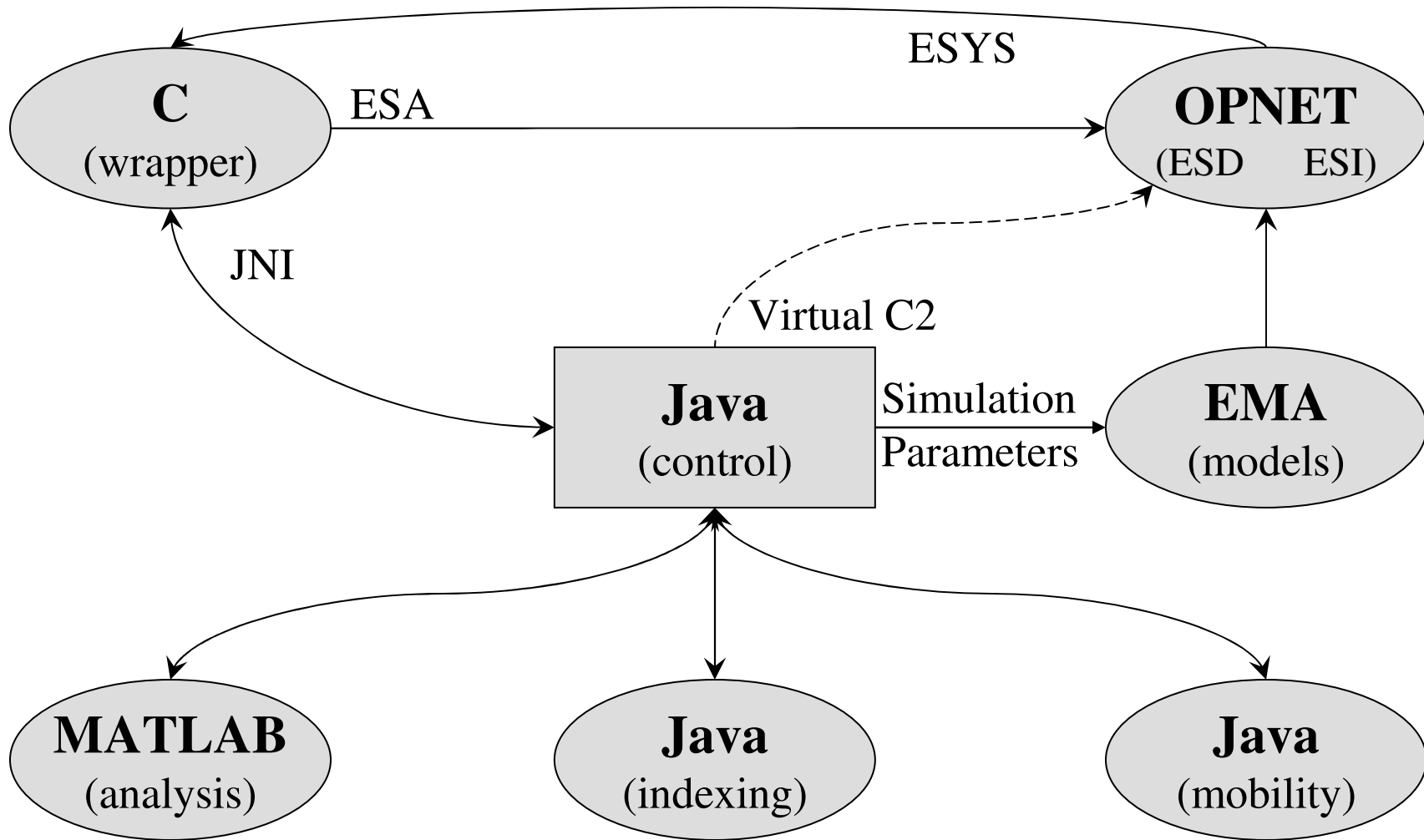
- **Search Areas**

- Searched

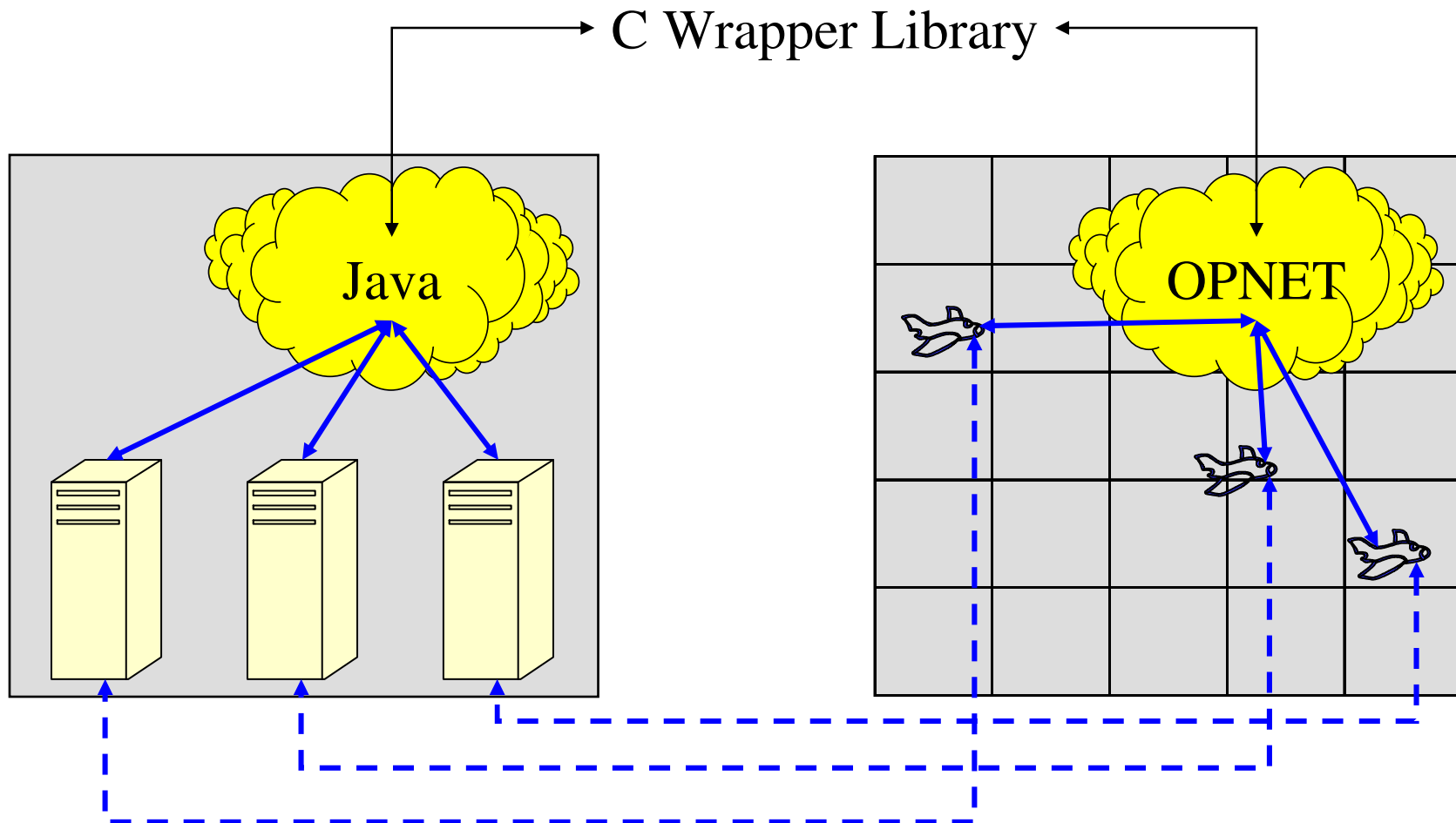
- Not searched



JOC-II: Design View



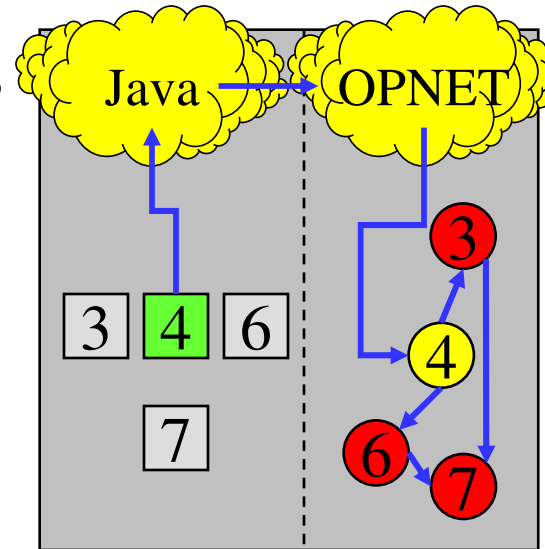
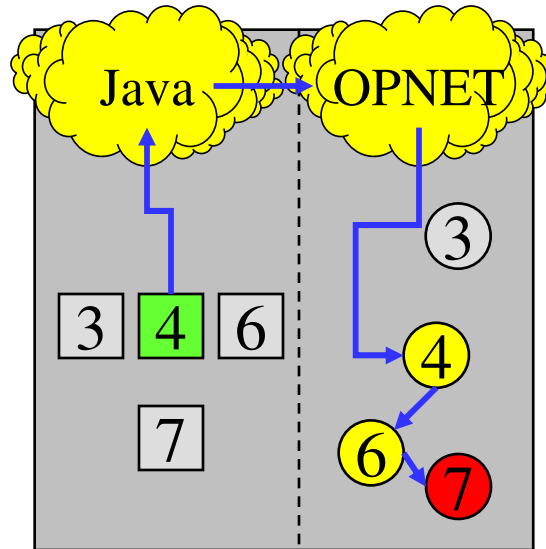
JOC-II: Conceptual Data Flow



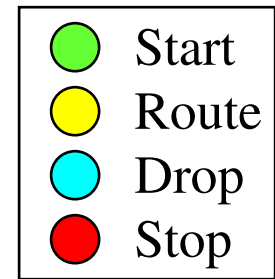
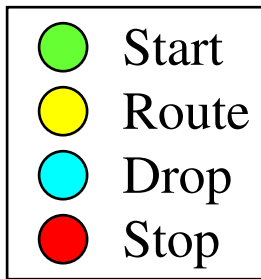
JOC-II: Network Comm



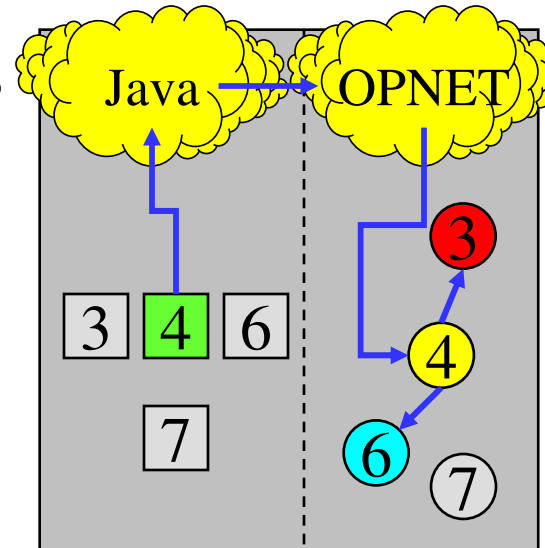
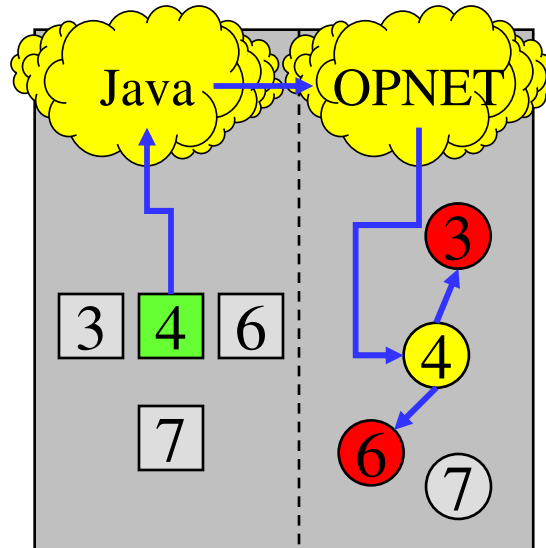
**Route, e.g.,
AODV, DSR
(+ID1, +ID2)**



**Network
Flood
(ID1, +∞)**



**Neighbor
Broadcast
(+ID1, 0)**

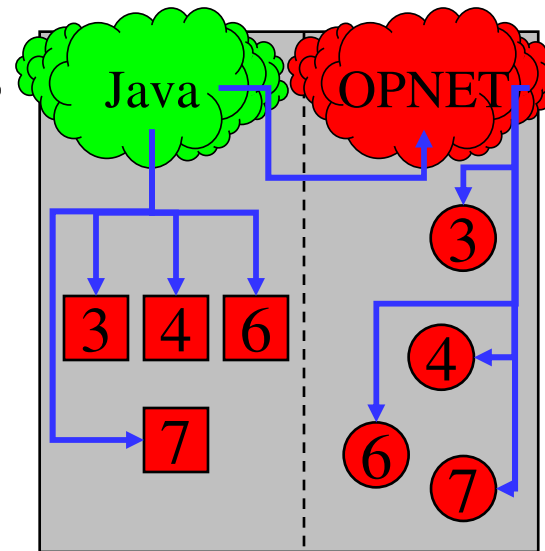
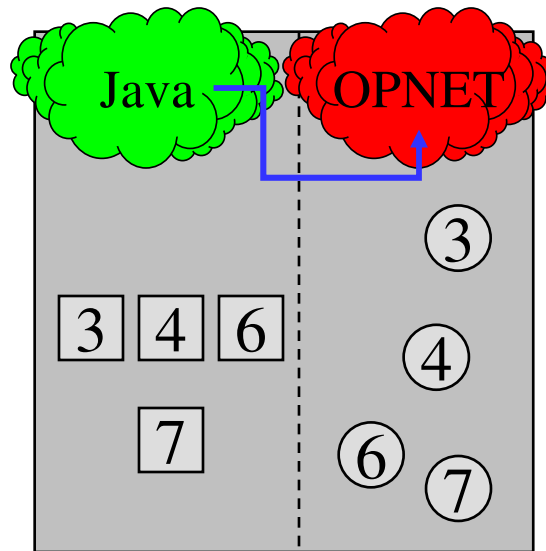
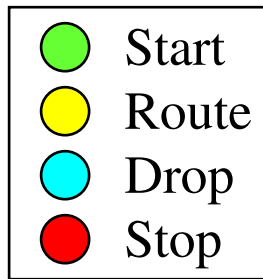


**Multicast
(+ID1, -ID2)**

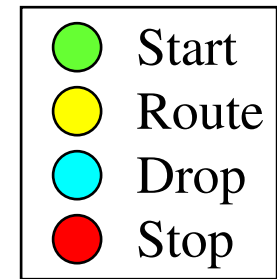
JOC-II: Inter-Process Comm



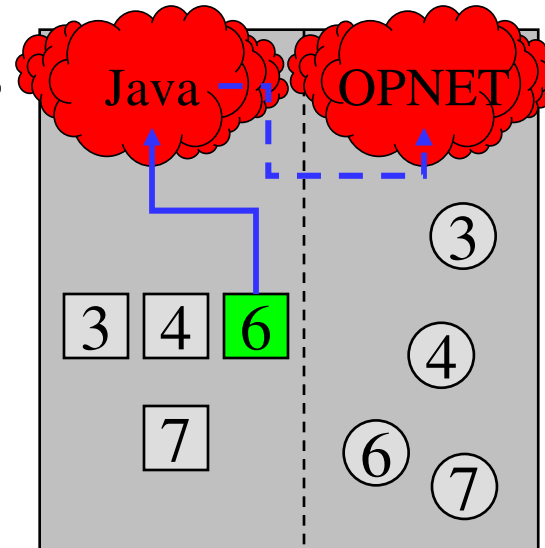
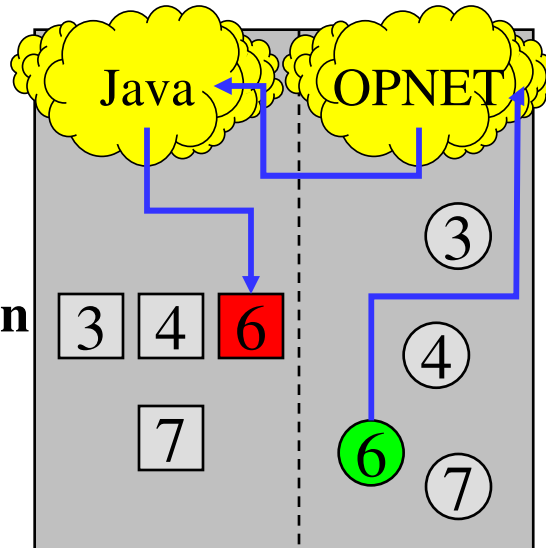
Simulation Management
(0, 0)



Policy Setting
(0, -ID1)



Peer Communication
(-ID1, -ID1)



Status Updates
(-ID1, 0)

Summary



- Discuss an unmanned swarm design framework, HARVEST
- Demonstrates Java & OPNET can successfully be used in a co-simulation
- Shows that Java is able to maintain positive control of an OPNET simulation
- Proposes that centralized cloud management provides a viable co-simulation mechanism



Questions?