



**MIT 2.680**  
UNMANNED MARINE VEHICLE AUTONOMY,  
SENSING, AND COMMUNICATIONS

## Lecture 3: Introduction To MOOS

February 14<sup>th</sup>, 2023

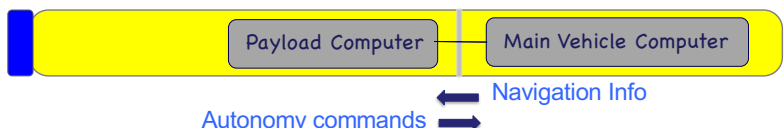


Web: <http://oceanai.mit.edu/2.680>  
Email: [Mike Benjamin, mikerb@mit.edu](mailto:Mike Benjamin, mikerb@mit.edu)

2.680 Spring 2023 – Marine Autonomy – “Programming MOOS Applications”  Photo by Arjan Vermeij, CMRE

1

**Payload UUV Autonomy**



Architecture Principle #1  
Payload Autonomy  
Decouple the Procurement of Hardware and Software

MITMECHE MIT

Three Architectures MOOS Overview MOOS Messages Launching Missions Scoping MOOS Poking MOOS Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

2

MITMECHE MIT

## Payload UUV Autonomy

Payload Computer — Main Vehicle Computer

Payload Computer

MOOS Middleware  
MOOS Applications

Architecture Principle #2  
Autonomy System Middleware

De-couple Software Procurements  
Sensing, Autonomy, Simulation, Comms...

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS

Poking MOOS

Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

3

MITMECHE MIT

## Payload UUV Autonomy

Payload Computer — Main Vehicle Computer

Payload Computer

MOOS Middleware  
MOOS Applications

IvP Helm

Behavior-Based  
Modular HELM

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

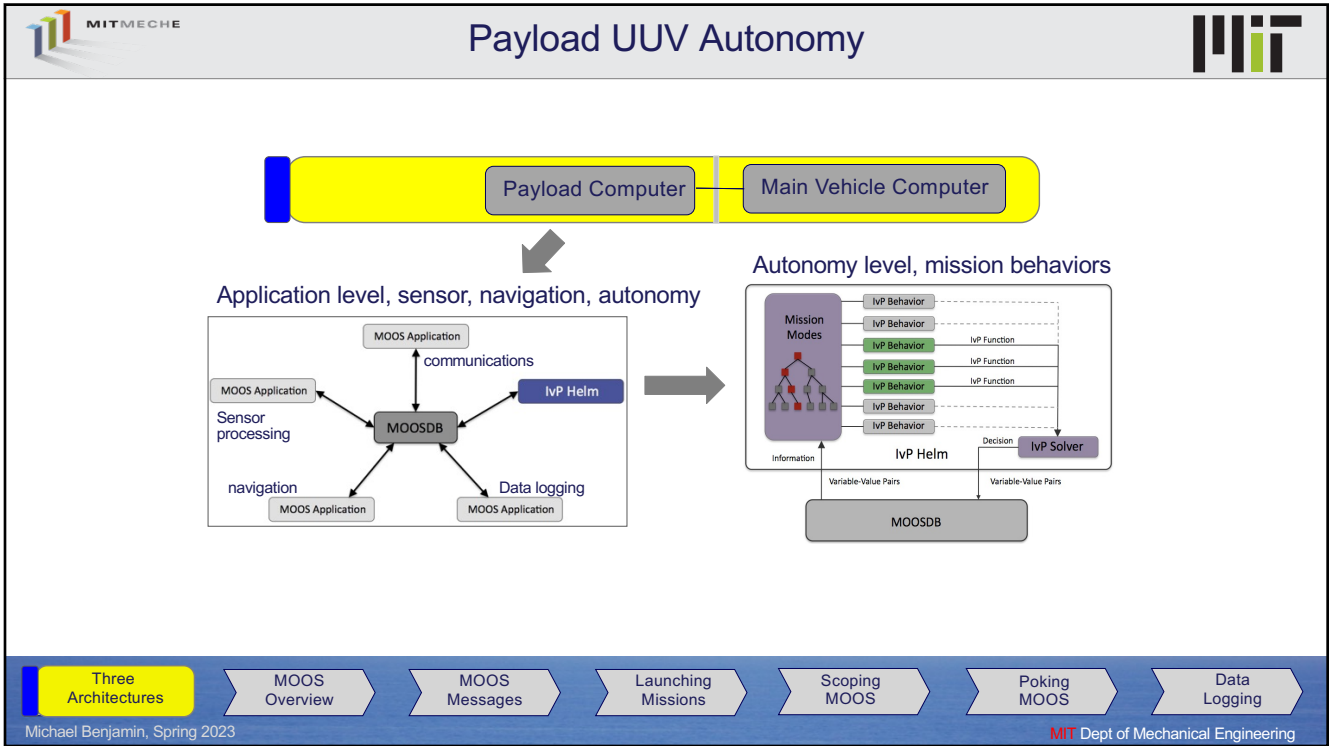
Scoping MOOS

Poking MOOS

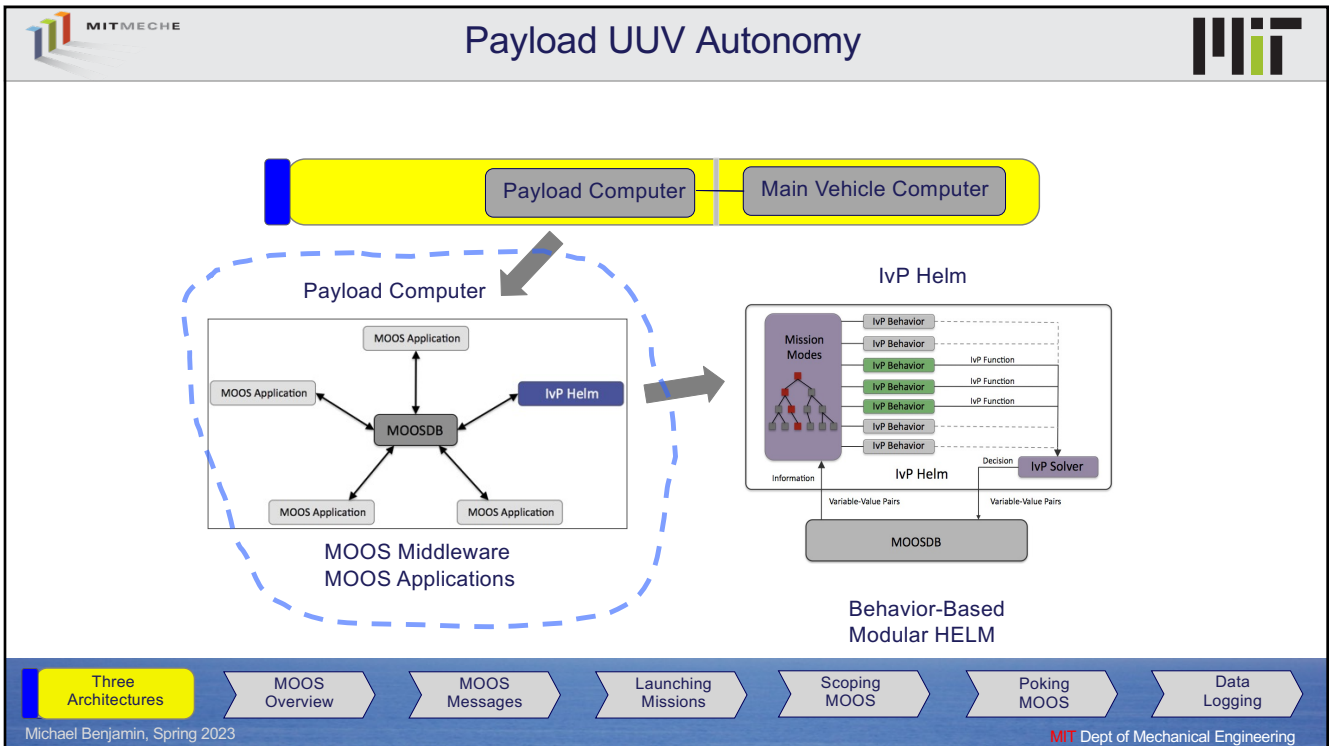
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

4



5




6


MITMECHE MIT

## MOOS Overview

- MOOS is a Robot Middleware
- Developed by Paul Newman, as an MIT post-doc and now Oxford Professor, Oxbotica Founder
- Initial development 2000-2003 on Bluefin Odyssey II UUV owned by MIT



Italy, Summer 2002



Italy, Summer 2002

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS

Poking MOOS

Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

7

MITMECHE MIT

## MOOS Overview








Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

8

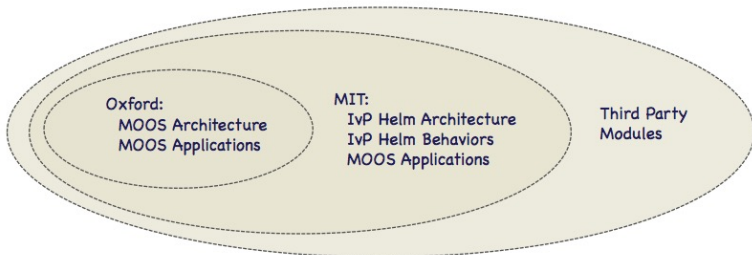




## Nested Repositories



- MOOS, from the Mobile Robotics Group at Oxford  
 - MOOS-IvP, from the Laboratory for Autonomous Marine Sensing Systems at MIT  
 - 3<sup>rd</sup> Party (Your) modules.




The diagram shows three nested ovals representing repository levels:

- Innermost Oval (Oxford):** MOOS Architecture, MOOS Applications
- Middle Oval (MIT):** IvP Helm Architecture, IvP Helm Behaviors, MOOS Applications
- Outermost Oval (Third Party):** Modules


Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

9



## MOOS Modules in the MOOS-IvP Distribution



**The Oxford MOOS tree (11)**

- MOOSDB
- pLogger
- pShare
- pAntler
- iMatlab
- pScheduler
- uMS
- jMOOS
- iRemote
- pyMOOS
- uPlayback

**The moos-ivp tree (57)**

• pHelmIvP	• pMarineViewer	• uFldHazardMgr	• pEvalLoiter
• alogcd	• uMACView	• uFldHazardMetric	• pObstacleMgr
• alogcheck	• uFunctionVis	• uFldNodeBroker	• pMissionEval
• alogclip	• pMarinePID	• uFldShoreBroker	• uCommand
• alogeplot	• pEchoVar	• uFldNodeComms	• uFldObstacleSim
• aloggrep	• pRealm	• uFldPathCheck	• uFldCollObDetect
• aloghelm	• uPlotViewer	• uHelmScope	• uFldCollisionDetect
• alogiter	• nsplug	• uTimerScript	• uFldWrapDetect
• alogpare	• uXMS	• uProcessWatch	• uFunctionVis
• alogrm	• uMAC	• uTermCommand	• uLoadWatch
• alogscan	• iSay	• pContactMgrV20	• uFldMessageHandler
• alogsort	• zaic_hdg	• uQueryDB	• uFldContactRangeSensor
• alogsplit	• zaic_peak	• uPokeDB	• uFldBeaconRangeSensor
• alogview	• zaic_spd	• pHostInfo	• pNodeReporter
	• zaic_vect	• pDeadManPost	• uSimMarineV22

8 Work years of development effort

2.8 Mb size  
 0 dependencies  
 Download: 2 secs  
 Build: 7 secs


35 Work years of development effort

According to "sloccount"  
[www.dwheeler.com/sloccount](http://www.dwheeler.com/sloccount)


Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging



Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

10








## CMRE (Formerly NURC) MOOS Modules



The **CMRE (formerly NURC)** tree (124 MOOS Applications)

<ul style="list-style-type: none"> <li>• iCompassSocket</li> <li>• iDPCAMOOSSInterface</li> <li>• iFLIRRangerHRC</li> <li>• iFutabaJoyStick</li> <li>• iHardwareHealthMonitor</li> <li>• pRadarTrack2Status</li> <li>• uRangeBearingUUVSim</li> <li>• iMicrostrainCX1</li> <li>• pSonarImageProcessing</li> <li>• iSystemSpcMuscle</li> <li>• pArraySimToNad</li> <li>• pLinearTrajectoryGenerator</li> <li>• pTowedSourceNavEstimator</li> <li>• pCircleTrajectoryGenerator</li> <li>• pMaintainRelativeBearing</li> <li>• pBVBottomTargetDetector</li> <li>• pBacksteppingController</li> </ul>	<ul style="list-style-type: none"> <li>• iTcpClient</li> <li>• iTcpServer</li> <li>• iTelnetClient</li> <li>• iTisVis</li> <li>• i3DMGX1</li> <li>• iAxisM7001</li> <li>• iSentinel</li> <li>• iSerialPort</li> <li>• iPanTilt</li> <li>• iPlayBack</li> <li>• iTps730</li> <li>• iUdpClient</li> <li>• iUdpGatherer</li> <li>• iUdpScatterer</li> <li>• iXBee868</li> <li>• pAdaptiveSurvey</li> <li>• pAisToNmea</li> </ul>	<ul style="list-style-type: none"> <li>• iSidusPositioner</li> <li>• pCommandFilter</li> <li>• pCompassToNmea</li> <li>• pDmhtTracker</li> <li>• pDopplerSim</li> <li>• pDuripToSlita</li> <li>• pFLIRPanTilt</li> <li>• pFSM</li> <li>• pGetSlitaNAD</li> <li>• pGpsNodeReporter</li> <li>• pGuardian</li> <li>• pHelmToNmea</li> <li>• pHiPapUvTracker</li> <li>• plnstroLRF</li> <li>• pJoyStick</li> <li>• pKalmanTracker</li> <li>• pLaserTrack</li> </ul>	<ul style="list-style-type: none"> <li>• pNmeaToXml</li> <li>• pNull</li> <li>• pOctaver</li> <li>• pOENix</li> <li>• pOEX</li> <li>• iJoyStick</li> <li>• iLMSView</li> <li>• pOEXTel2Status</li> <li>• pOnte</li> <li>• iMaxaBeam</li> <li>• iSncZ20P</li> <li>• iRosPositioner</li> <li>• pPersistTracker</li> <li>• pPIDController</li> <li>• pPosToNMEA</li> <li>• pProcessAtlasBB</li> <li>• pProcessSlitaBB</li> </ul>	<ul style="list-style-type: none"> <li>• pREMUSCodec</li> <li>• pReplayAtlas</li> <li>• pScooter</li> <li>• pSpcMaster</li> <li>• pVLC</li> <li>• pWatchDog</li> <li>• pXBee</li> <li>• pXmlSerialiser</li> <li>• pXmlTransformer</li> <li>• pXmlUnpacker</li> <li>• uBcSlitaPlayer</li> <li>• uBcSlitaRcvr</li> <li>• uLRM2500CI</li> <li>• uMaxaBeam</li> <li>• uScooter</li> <li>• uUVRacquire</li> <li>• uUvTracker</li> </ul>	<ul style="list-style-type: none"> <li>• pTrackerUUVSim</li> <li>• pTrackEvaluator</li> <li>• pSurveyPlanner</li> <li>• pCartesianToGeo</li> <li>• iExploreDVL</li> <li>• pUVDeploy</li> <li>• pUVNodeReporter</li> <li>• pTrigger</li> <li>• pBistaticLocator</li> <li>• iBlueView</li> <li>• pNmeaToAis</li> <li>• pMuscle</li> <li>• pMaxaBeam</li> <li>• pUvLink</li> <li>• iLRM2500CI</li> <li>• pTrackBuilder</li> <li>• pIngTimer</li> <li>• pAlarmBox</li> </ul>	<ul style="list-style-type: none"> <li>• iBVLMS</li> <li>• pUvTracker</li> <li>• pVelvet</li> <li>• uWitty</li> <li>• pTgtDetector</li> <li>• pUVRacquire</li> <li>• pRealAIS2Status</li> <li>• pRealAIS2Xml</li> <li>• pHomer</li> <li>• pLaunchTarget</li> <li>• pLRAD</li> <li>• pLuribus</li> <li>• pAntagonist</li> <li>• pAnTiller</li> <li>• pBearingTrack</li> <li>• pActivePassiveManager</li> <li>• pArrayNavEstimator</li> <li>• pBVImgProcessing</li> </ul>
---	---	---	---	--	--	---

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

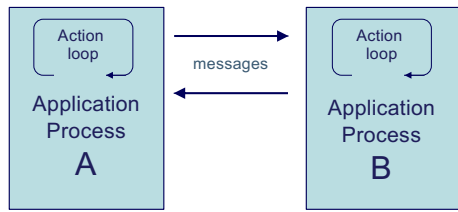
11



## MOOS Does Two Main Things



1. It enables distinct applications to communicate
2. It enables users to control the frequency of each application's action loop



```

    graph LR
      subgraph A [Application Process A]
        direction TB
        AL1[Action loop]
        AL1 --> AL1
      end
      subgraph B [Application Process B]
        direction TB
        AL2[Action loop]
        AL2 --> AL2
      end
      A -- messages --> B
      B -- messages --> A
    
```

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

12



## The Beauty of Separate Processes



Application Process  
**A**

Application Process  
**B**

Application Process  
**C**

On Unix based systems, each process:

- Has a unique Process ID (PID)
- Uses a chunk of computer memory *separate* from all other processes


Advantages:

- A crash in one process will not affect another process
- The OS automatically distributes processes over system CPU cores


Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

13



## MOOSDB is a Process for Communication



- It has its own PID and memory space like any other process
- It maintains a mapping for Variable Names → Values

**MOOSDB**


FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	alpha
WIDTH	86
HOURS	23

Only the most recent value is retained


Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

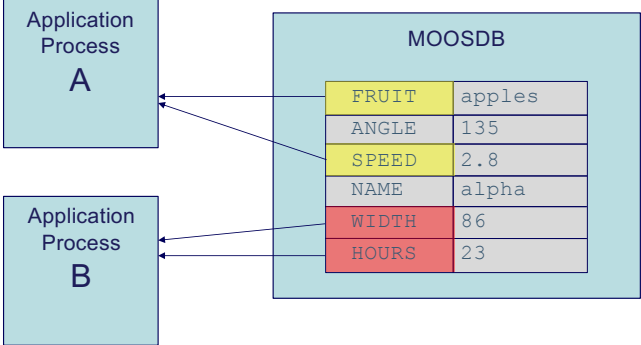
14



## MOOS Apps Subscribe to the MOOSDB



- An App may register (subscribe for) for any variable
- An App may register any time, but typically during startup
- Multiple apps may register for the same variable




When an App first connects, it gets mail for each registered variable.  
(if the variable has ever been written to)


Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

15

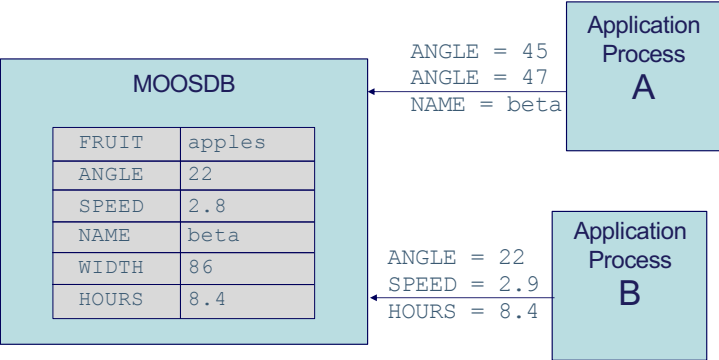


## MOOS Apps Publish to the MOOSDB



- An App may publish to the MOOSDB any time
- No prior arrangement required

Note: Subscribers will get **all** postings – each as a new piece of mail.




Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging


Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

16





## MOOS Apps Publish to the MOOSDB



- An App may publish to the MOOSDB any time
- No prior arrangement required

Application Process C

MOOSDB

FRUIT	apples
ANGLE	22
SPEED	2.8
NAME	beta
WIDTH	86
HOURS	8.4

Application Process A

Time = N

ANGLE = 45  
ANGLE = 47  
NAME = beta

Time = N+1

ANGLE = 22  
SPEED = 2.9  
HOURS = 8.4

Time = N+2

ANGLE = 45  
ANGLE = 47  
ANGLE = 22  
NAME = beta  
SPEED = 2.9

App C subscribes for ANGLE, NAME, SPEED

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

17



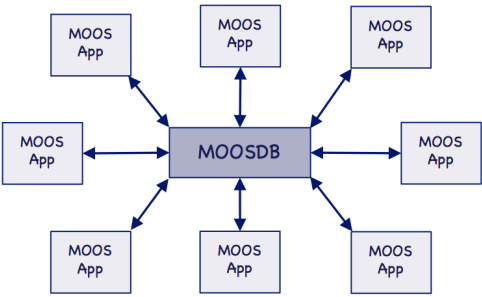
## A MOOS Community



- A MOOS *community* is comprised of one MOOSDB and all connected Apps
- MOOS is described as having a *star* topology.

A community also has a unique

- name
- IP address, Port number



Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

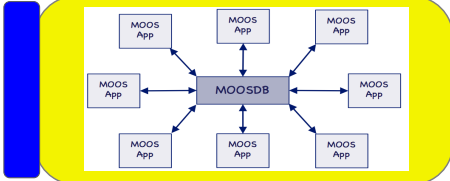
18



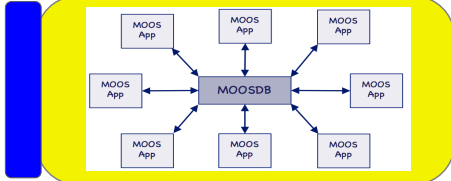
## A MOOS Community Per Robot



- Typically, one community per vehicle/robot
- Sometimes multiple computers on one vehicle, each with a community



Community 1



Community 2

- Inter-community communications addressed later

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

19

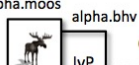


## Alpha Mission



```
$ cd moos-ivp/ivp/missions/s1_alpha
$ ./launch.sh 10
```

**alpha.moos**   **alpha.bhv**



**ivp** → Configure → **pHelmIvP**

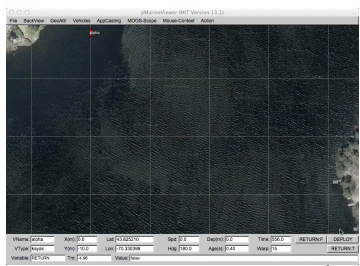
**pHelmIvP** ↔ **MOOSDB** (1)

**MOOSDB** ↔ **pMarinePID** (2)

**MOOSDB** ↔ **uSimMarine** (3)

**MOOSDB** ↔ **pNodeReporter** (4)

**MOOSDB** ↔ **pMarineViewer** (5) ↔ **The User**



Vehicle	Altitude	Heading	Speed	Rudder	Thrust	YawRate	Roll	Pitch	Yaw	RollRate	PitchRate	YawRate	RollRate	PitchRate
Vehicle	1000	100	10	0	100	0	0	0	0	0	0	0	0	0

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

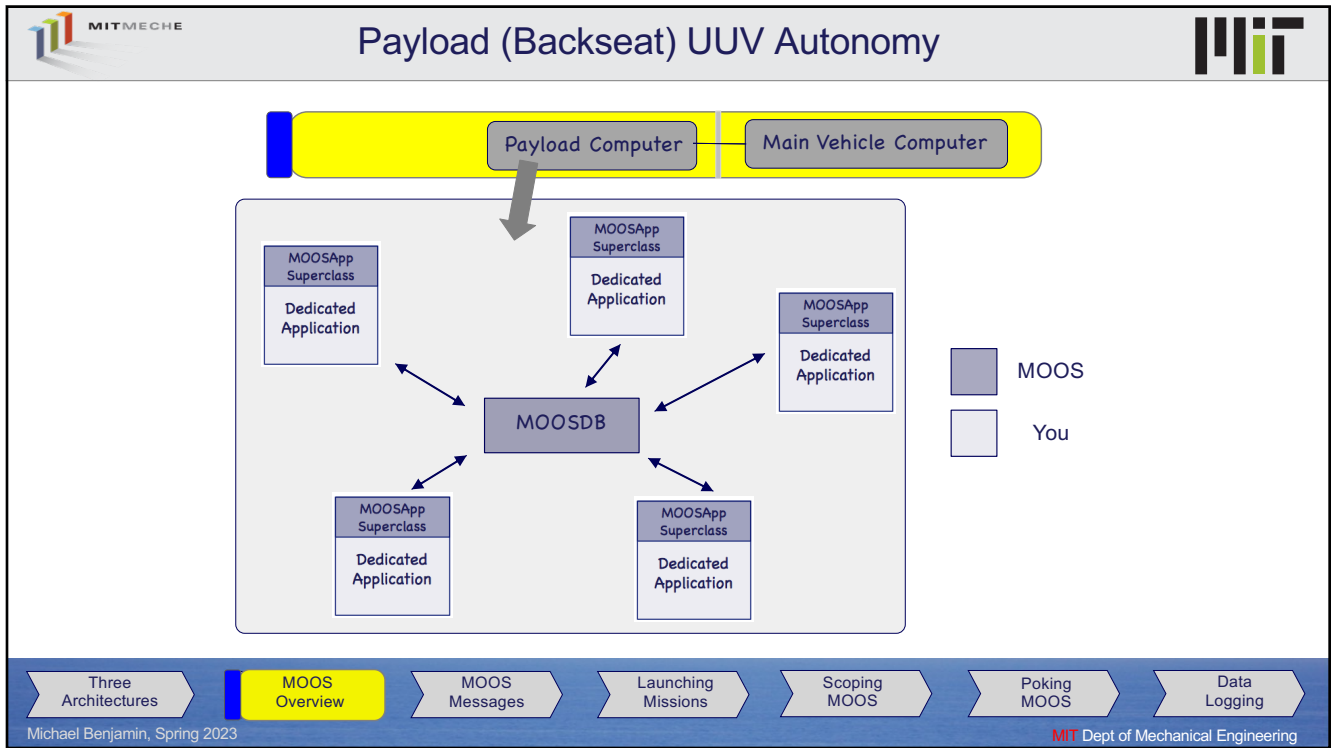
Scoping MOOS

Poking MOOS

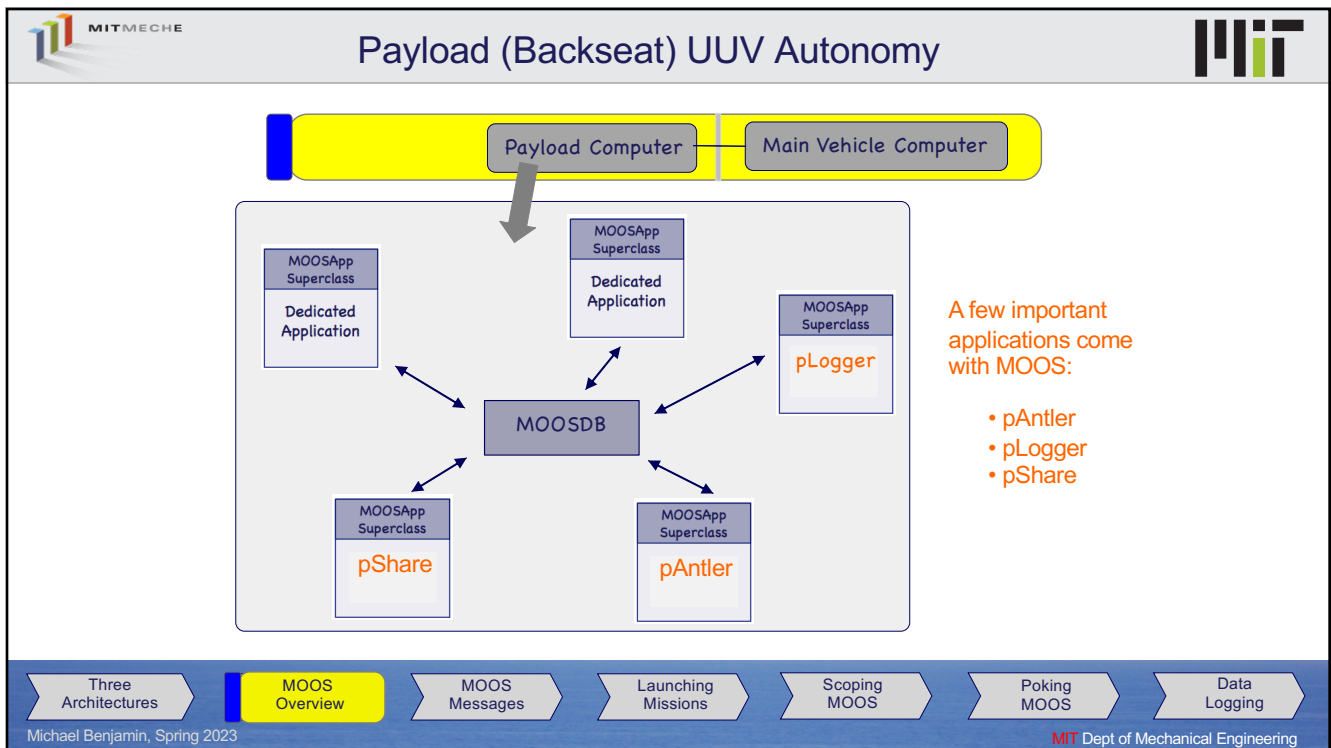
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

20



21



22

**Public Infrastructure – Nested Capabilities**

An autonomy system has components with different capabilities, and distribution access.

- Publicly accessible modules providing infrastructure, basic capabilities
- Restricted-access modules for developers of a particular project.

Autonomy System = Infrastructure + Modules

Three Architectures | **MOOS Overview** | MOOS Messages | Launching Missions | Scoping MOOS | Poking MOOS | Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering


23

**MOOS Messages**


Three Architectures | MOOS Overview | **MOOS Messages** | Launching Missions | Scoping MOOS | Poking MOOS | Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

24



## MOOS Messages



- Two primary message components: **VARIABLE** and **VALUE**
- Two primary message types: **STRING** and **DOUBLE**

MOOSDB	
FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	alpha
WIDTH	86
HOURS	23

Name	The name of the data
StringVal	Data in human-readable string format, or raw binary data
DoubleVal	Numeric double float data

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

25



## MOOS Message Examples



MOOSDB	
FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	alpha
WIDTH	86
HOURS	23

Name	FRUIT
StringVal	"apples"
DoubleVal	0
DataType	string

Name	WIDTH
StringVal	" "
DoubleVal	86
DataType	double

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

26





## MOOS Messages



Each MOOS Message contains additional useful information:

Name	The name of the data
StringVal	Data in human-readable string format, or raw binary data
DoubleVal	Numeric double float data
DataType	Type of data ( <b>STRING</b> or <b>DOUBLE</b> or <b>BINARY</b> )
Source	Name of client that sent this data to the MOOSDB
SourceAux	Optional additional information about the source client
Time	Time at which the data was written
Community	The community to which the source process belongs

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

27



## Posting MOOS Messages

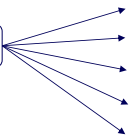


Inside your MOOS application you may post a message with simple line in C++:

```
Notify("FRUIT", "apples");
```

MOOS will automatically fill in the additional fields:

Auto-filled



Name	FRUIT
StringVal	"apples"
DoubleVal	0
DataType	String
Source	pFoobar
SourceAux	
Time	34558.2
Community	alpha

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

28

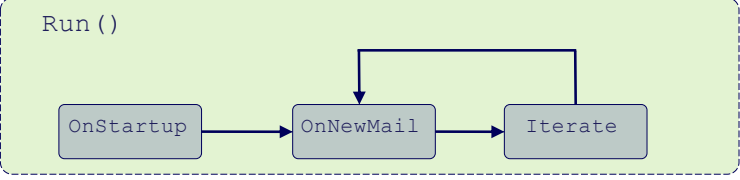


## Reading MOOS Messages



- MOOS Apps read messages inside a mail-handling function
- This function is defined in the MOOSApp superclass for all MOOS Apps

Run ()



```

graph LR
    OnStartup[OnStartup] --> OnNewMail[OnNewMail]
    OnNewMail --> Iterate[Iterate]
    Iterate --> OnNewMail
  
```


The Flow of Control for all MOOS Apps

Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

29

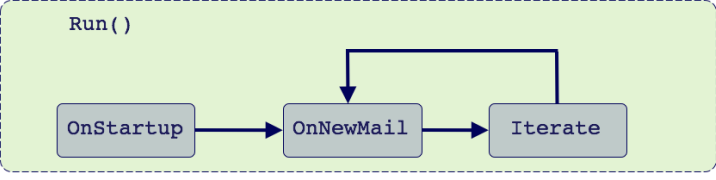


## A Mail Handling Example



- An example OnNewMail() implementation:

Run ()



```

graph LR
    OnStartup[OnStartup] --> OnNewMail[OnNewMail]
    OnNewMail --> Iterate[Iterate]
    Iterate --> OnNewMail
  
```

```

bool MyApp::OnNewMail(MOOSMSG_LIST &NewMail)
{
    MOOSMSG_LIST::iterator p; for(p=NewMail.begin(); p!=NewMail.end(); p++) {
        CMOOSMsg &msg = *p;
        string key = msg.GetKey();


        if(key == "WIDTH")
            updateWidth(msg.getDouble());
    }
    return(true);
}
  
```

Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

30



## Handling a MOOS Message



Other useful functions defined on a MOOS Message:

```

MOOSMsg msg;

string moos_var = msg.GetKey();           // the MOOS variable name

bool is_double = msg.IsDouble();         // true if message content double
bool is_string = msg.IsString();        // true if message content string

double timestamp = msg.GetTime();       // timestamp when message posted

string str_val = msg.GetString();       // the message string content
string dbl_val = msg.GetDouble();      // the message double content

string source = msg.GetSource();        // who (which app) posted message
string src_aux = msg.GetSourceAux();    // further source information

string community = msg.GetCommunity();  // MOOS community who posted

```

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

31



# Launching MOOS



Three Architectures

MOOS Overview

MOOS Messages

Launching MOOS

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

32



## Launching MOOS (Bare Bones)



The MOOSDB may be launched from the command line:

```
$ MOOSDB
```

- The new MOOSDB process is the beginning of a MOOS community
- Recall a community has an **IP Address, Port Number, Community Name**


Terminal output:

```
----- MOOSDB V10 -----
Hosting community           "#1"      Default Community Name
Name look up is             off
Asynchronous support is    on
Connect to this server on port 9000     Default Port Number
-----
network performance data published on localhost:9020  Default IP Address
listen with "nc -u -lk 9020"
```


Three Architectures
MOOS Overview
MOOS Messages
Launching MOOS
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

33



## Launching MOOS (with Mission File)



- The IP Address, Port Number and Community Name may be provided in a mission file.
- The mission file is a command line argument:

```
$ MOOSDB mission.moos
```


```
mission.moos
Community = alpha
ServerPort = 9205
ServerHost = localhost
```

```
----- MOOSDB V10 -----
Hosting community           "alpha"
Name look up is             off
Asynchronous support is    on
Connect to this server on port 9205
-----
network performance data published on localhost:9020
listen with "nc -u -lk 9020"
```


Three Architectures
MOOS Overview
MOOS Messages
Launching MOOS
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

34



## Launching MOOS and Mission Configuration



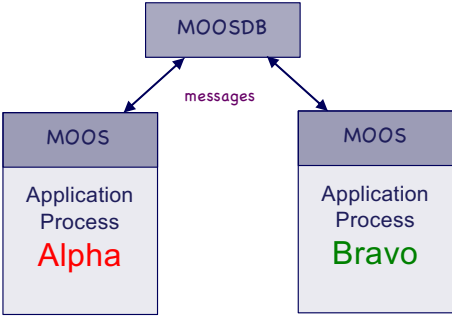
- A mission file may also hold configuration parameters for MOOS apps
- Each application has a dedicated configuration block.

```

mission.moos
Global parameters

ProcessConfig = alpha
{
  alpha parameters
}


ProcessConfig = bravo
{
  alpha parameters
}
            
```




Three Architectures
MOOS Overview
MOOS Messages
Launching MOOS
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

35



## MOOS Mission Configuration



- Mission configuration is through a single “mission file”, with a .moos extension.
- Each application has a dedicated configuration block.

```

mission.moos
Global parameters

ProcessConfig = alpha
{
  alpha parameters
}

ProcessConfig = bravo
{
  alpha parameters
}
            
```

“Global parameters” are accessible to all MOOS applications. They include things like:


- MOOSDB server IP address and port number.
- Local datum (0,0) in lat/lon coordinates.
- Name of the MOOS community.

Three Architectures
MOOS Overview
MOOS Messages
Launching MOOS
Scoping MOOS
Poking MOOS
Data Logging


Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

36





## MOOS Mission Configuration



- Mission configuration is through a single “mission file”, with a .moos extension.
- Each application has a dedicated configuration block.

```

mission.moos
Global parameters

ProcessConfig = alpha
{
  alpha parameters
}

ProcessConfig = bravo
{
  alpha parameters
}
  
```

“Application parameters”  
Accessible only to a particular application.

Application authors implement the handling of parameters upon application startup.

The MOOSApp superclass has a function called OnStartup() where configuration parameters are handled.

Application authors have access to each line in the application’s configuration block to handle as they see fit.

Three Architectures

MOOS Overview

MOOS Messages

Launching MOOS

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

37



## Scoping MOOS



Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

38



## Scoping MOOS



**Scoping** the MOOSDB means examining:

- Current values of variables known to the MOOSDB
- Which processes made the most recent post
- When it was posted
- The community of the application making the post.

MOOSDB

FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	alpha
WIDTH	86
HOURS	23

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

39



## Scoping MOOS



**Scoping** the MOOSDB means examining:

- Current values of variables known to the MOOSDB
- Which processes made the most recent post
- When it was posted
- The community of the application making the post.

MOOSDB

VarName	Source	Community	Time	VarValue
FRUIT	pFruit	alpha	143.21	apples
ANGLE	uMeasure	alpha	1873.24	135
SPEED	uMeasure	alpha	62.11	2.8
NAME	pIdentity	gamma	3.91	alpha
WIDTH	uMeasure	alpha	1873.24	86
HOURS	uMeasure	alpha	1873.25	23

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

40



## Scoping MOOS



**Scoping** the MOOSDB means examining:

- Current values of variables known to the MOOSDB
- Which processes made the most recent post
- When it was posted
- The community of the application making the post.

**MOOSDB**

VarName	Source	Community	Time	VarValue
FRUIT	pFruit	alpha	143.21	apples
ANGLE	uMeasure	alpha	1873.24	135
SPEED	uMeasure	alpha	62.11	2.8
NAME	pIdentity	gamma	3.91	alpha
WIDTH	uMeasure	alpha	1873.24	86
HOURS	uMeasure	alpha	1873.25	23

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

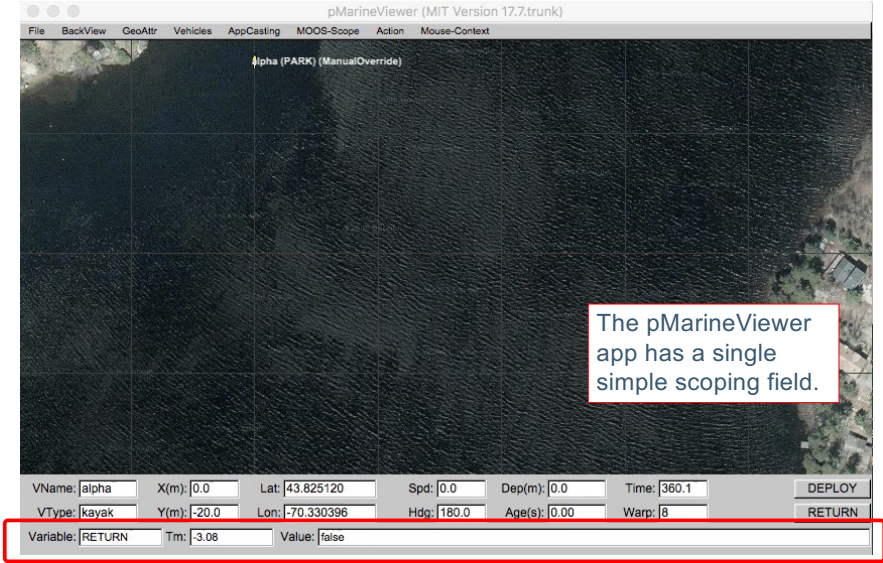
Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

41



## A Simple Single Scope in pMarineViewer





The pMarineViewer app has a single simple scoping field.

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS

Poking MOOS

Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

42

**Changing the Scope Variable in pMarineViewer**

The scope variable may be changed:  
Hit CTRL-'A'  
Enter a new variable.

Variable: RETURN Tm: 548.72 Value: false

Three Architectures | MOOS Overview | MOOS Messages | Launching Missions | **Scoping MOOS** | Poking MOOS | Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

43

**The uXMS Scope List**


uXMS is a simple scoping utility launched from the command line

```
$ uXMS mission.moos --all
```


Three Architectures | MOOS Overview | MOOS Messages | Launching Missions | **Scoping MOOS** | Poking MOOS | Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

44



## The uXMS Scope List



uXMS is a simple scoping utility launched from the command line

```
$ uXMS mission.moos --all
```

- To scope on a MOOSDB, uXMS must connect to the MOOSDB.
- Where is the server?
- It could be anywhere on the internet.
- Exactly where? This is determined by the IP address and the port number, for the MOOSDB server.

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

45



## The uXMS Scope List



uXMS is a simple scoping utility launched from the command line

```
$ uXMS mission.moos --all
```

- To scope on a MOOSDB, uXMS must connect to the MOOSDB.
- Where is the server?
- It could be anywhere on the internet.
- Exactly where? This is determined by the IP address and the port number, for the MOOSDB server.

```
mission.moos
ServerHost = localhost
ServerPort = 9005
```

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS

Poking MOOS


Data Logging

MIT Dept of Mechanical Engineering


Michael Benjamin, Spring 2023

46





## The uXMS Scope List



uXMS

is a simple scoping utility launched from the command line

```
$ uXMS mission.moos --all
```

- To scope on a MOOSDB, uXMS must connect to the MOOSDB.
- Where is the server?
- It could be anywhere on the internet.
- Exactly where? This is determined by the IP address and the port number, for the MOOSDB server.

```
mission.moos
ServerHost = localhost
ServerPort = 9005
```


The same information may also be passed on the command line as arguments to uXMS

```
$ uXMS --all --serverhost=localhost --serverport=9005
```


Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

47



## The uXMS Scope List



uXMS

is a simple scoping utility launched from the command line

```
$ uXMS mission.moos --all
```

By default, the screen will refresh whenever one variable value changes

VarName	(S)ource	(T)ime	(C)ommunity	VarValue
DB_CLIENTS	MOOSDB_alpha	106.2	alpha	"uXMS,DBWebServer,"
DB_TIME	MOOSDB_alpha	107.2	alpha	1325701208.08963
DB_UPTIME	MOOSDB_alpha	107.2	alpha	107.20791
FRUIT	pFruit	143.21	alpha	"apples"
ANGLE	uMeasure	107.2	alpha	135
SPEED	uMeasure	107.2	alpha	2.8
NAME	pIdentity	3.91	gamma	"alpha"
WIDTH	uMeasure	1873.24	alpha	86
HOURS	uMeasure	1873.25	alpha	23
-- displaying all variables --				

All scoped variables

Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

48

MITMECHE MIT

## The uXMS Scope List

uXMS is a simple scoping utility launched from the command line

```
$ uXMS mission.moos --all
```

By default, the screen will refresh whenever one variable value changes

VarName	(S)ource	(T)ime	(C)ommunity	VarValue
DB_CLIENTS	MOOSDB_alpha	106.2	alpha	"uXMS,DBWebServer,"
DB_TIME	MOOSDB_alpha	107.2	alpha	1325701208.08963
DB_UPTIME	MOOSDB_alpha	107.2	alpha	107.20791
FRUIT	pFruit	143.21	alpha	"apples"
ANGLE	uMeasure	107.2	alpha	135
SPEED	uMeasure	107.2	alpha	2.8
NAME	pIdentity	3.91	gamma	"alpha"
WIDTH	uMeasure	1873.24	alpha	86
HOURS	uMeasure	1873.25	alpha	23
-- displaying all variables --				

Name of the app that last published the variable

Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

49

MITMECHE MIT

## The uXMS Scope List

uXMS is a simple scoping utility launched from the command line

```
$ uXMS mission.moos --all
```

By default, the screen will refresh whenever one variable value changes


VarName	(S)ource	(T)ime	(C)ommunity	VarValue
DB_CLIENTS	MOOSDB_alpha	106.2	alpha	"uXMS,DBWebServer,"
DB_TIME	MOOSDB_alpha	107.2	alpha	1325701208.08963
DB_UPTIME	MOOSDB_alpha	107.2	alpha	107.20791
FRUIT	pFruit	143.21	alpha	"apples"
ANGLE	uMeasure	107.2	alpha	135
SPEED	uMeasure	107.2	alpha	2.8
NAME	pIdentity	3.91	gamma	"alpha"
WIDTH	uMeasure	1873.24	alpha	86
HOURS	uMeasure	1873.25	alpha	23
-- displaying all variables --				

Time of the last publication


Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

50



## The uXMS Scope List



uXMS is a simple scoping utility launched from the command line

```
$ uXMS mission.moos --all
```

By default, the screen will refresh whenever one variable value changes

VarName	(S)ource	(T)ime	(C)ommunity	VarValue
DB_CLIENTS	MOOSDB_alpha	106.2	alpha	"uXMS,DBWebServer,"
DB_TIME	MOOSDB_alpha	107.2	alpha	1325701208.08963
DB_UPTIME	MOOSDB_alpha	107.2	alpha	107.20791
FRUIT	pFruit	143.21	alpha	"apples"
ANGLE	uMeasure	107.2	alpha	135
SPEED	uMeasure	107.2	alpha	2.8
NAME	pIdentity	3.91	gamma	"alpha"
WIDTH	uMeasure	1873.24	alpha	86
HOURS	uMeasure	1873.25	alpha	23
-- displaying all variables --				

The community of the app that made the last publication

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

51



## The uXMS Scope List



uXMS is a simple scoping utility launched from the command line

```
$ uXMS mission.moos --all
```

By default, the screen will refresh whenever one variable value changes

VarName	(S)ource	(T)ime	(C)ommunity	VarValue
DB_CLIENTS	MOOSDB_alpha	106.2	alpha	"uXMS,DBWebServer,"
DB_TIME	MOOSDB_alpha	107.2	alpha	1325701208.08963
DB_UPTIME	MOOSDB_alpha	107.2	alpha	107.20791
FRUIT	pFruit	143.21	alpha	"apples"
ANGLE	uMeasure	107.2	alpha	135
SPEED	uMeasure	107.2	alpha	2.8
NAME	pIdentity	3.91	gamma	"alpha"
WIDTH	uMeasure	1873.24	alpha	86
HOURS	uMeasure	1873.25	alpha	23
-- displaying all variables --				

The value of the last publication

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

52

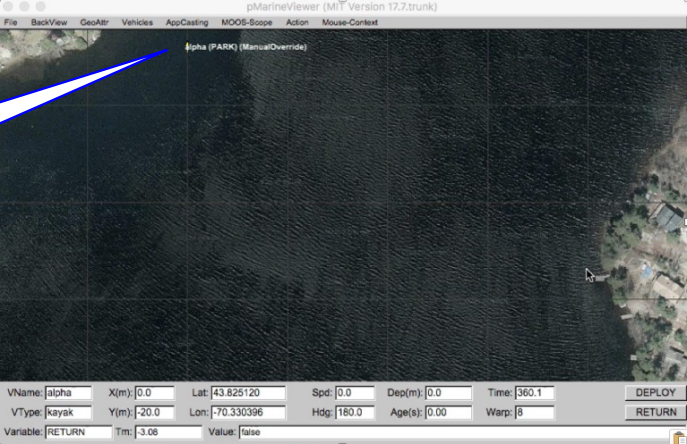


## Scoping on the Alpha Example Mission with uXMS



Launch the mission

```
$ cd moos-ivp/ivp/missions/s1_alpha
$ pAntler alpha.moos
```



At the start of the mission the vehicle sits motionless at the start position at point (0,-20) in local coordinates.

VName: alpha	X(m): 0.0	Lat: 43.825120	Spd: 0.0	Dep(m): 0.0	Time: 360.1	DEPLOY
VType: kayak	Y(m): -20.0	Lon: -70.330396	Hdg: 180.0	Age(s): 0.00	Warp: β	RETURN
Variable: RETURN	Tm: 3.08	Value: false				

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

53



## Scoping on the Alpha Example Mission with uXMS

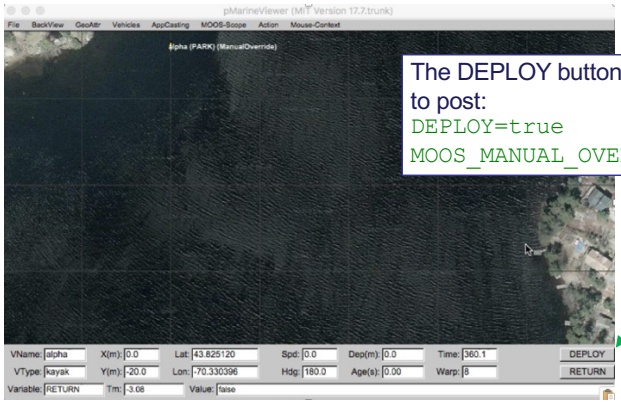


Launch the mission

```
$ cd moos-ivp/ivp/missions/s1_alpha
$ pAntler alpha.moos
```

In a separate terminal window, launch uXMS with the following variables:

```
$ uXMS alpha.moos \
  NAV_X NAV_Y \
  NAV_SPEED \
  NAV_HEADING \
  DEPLOY \
  IVPHELM_STATE \
  MOOS_MANUAL_OVERRIDE
```



The DEPLOY button is configured to post:  
 DEPLOY=true  
 MOOS\_MANUAL\_OVERRIDE=false

Launch the mission. Hit the DEPLOY button.

VName: alpha	X(m): 0.0	Lat: 43.825120	Spd: 0.0	Dep(m): 0.0	Time: 360.1	DEPLOY
VType: kayak	Y(m): -20.0	Lon: -70.330396	Hdg: 180.0	Age(s): 0.00	Warp: β	RETURN
Variable: RETURN	Tm: 3.08	Value: false				

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

54



## Scoping on the Alpha Example Mission with uXMS



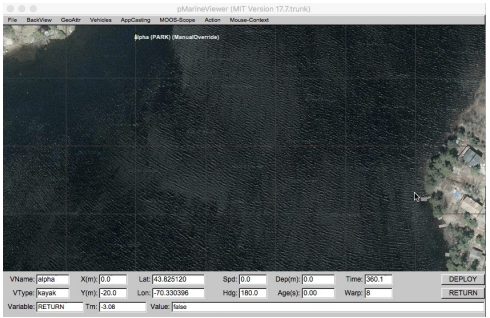
Launch the mission

```
$ cd moos-ivp/ivp/missions/s1_alpha
$ pAntler alpha.moos
```

Launch the scope

```
$ uXMS alpha.moos. NAV_X NAV_Y NAV_SPEED NAV_HEADING
DEPLOY IVPHELM_STATE MOOS_MANUAL_OVERRIDE
```


```
=====
uXMS_353 alpha                                0/0(2065)
=====
VarName      (S)ource      (T)ime      (C)  VarValue (SCOPING:EVENTS)
-----
DEPLOY       pMarineViewer  1808.98     "true"
IVPHELM_STATE pHelmIvP      1972.28     "DRIVE"
MOOS_MANUAL_OVERRIDE pMarineViewer 1808.98     "false"
NAV_HEADING  uSimMarine    1972.20     83.331698
NAV_SPEED    uSimMarine    1972.20     3.58
NAV_X        uSimMarine    1972.20     70.907074
NAV_Y        uSimMarine    1972.20     -161.709742
=====
```




Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

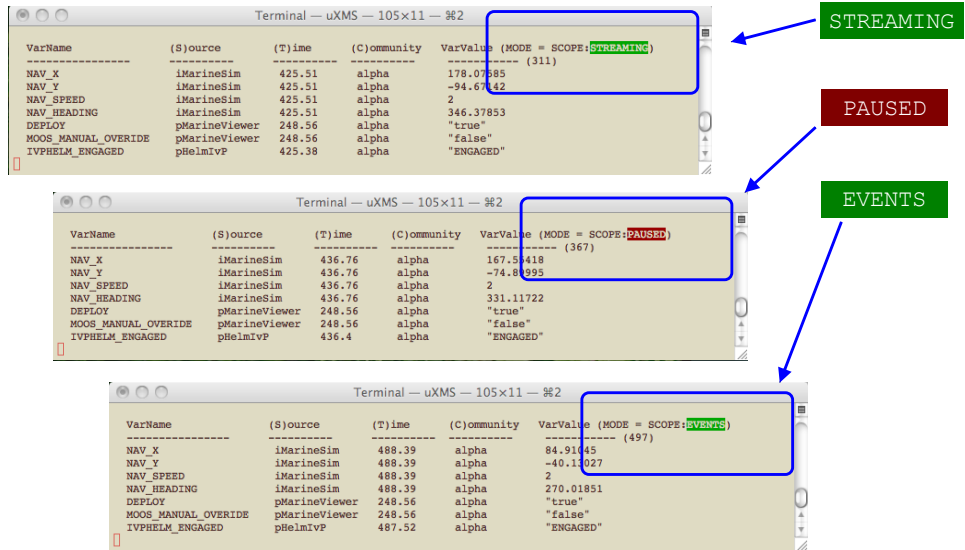
55



## The uXMS Utility Refresh Mode Indicator



The uXMS refresh mode is indicated in the top right-hand corner of each report:




Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging


Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

56

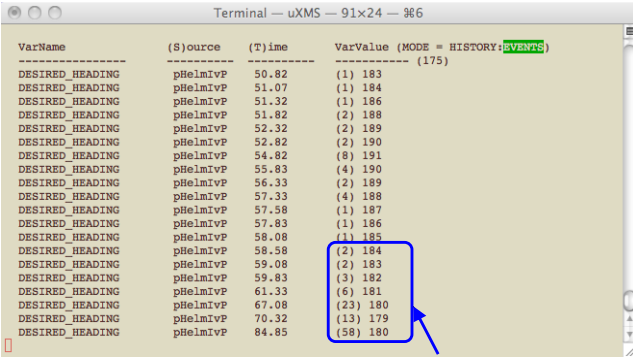






## The uXMS Utility: The "History" Content Mode



```
$ uXMS mission.moos --history=DESIRED_HEADING
```








Successive duplicate entries are condensed into a single line with the number of duplicates indicated in parentheses.


Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

57



## Setting the Scope List by App Name



uXMS can be launched to scope only on variables from a given App:

```
$ uXMS mission.moos --src=uMeasure
```

VarName	(S)ource	(T)ime	(C)ommunity	VarValue	(7)
ANGLE	uMeasure	107.2	alpha	135	
SPEED	uMeasure	107.2	alpha	2.8	
WIDTH	uMeasure	1873.24	alpha	86	
HOURS	uMeasure	1873.25	alpha	23	

Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

58



MITMECHE MIT

## Scoping LOCALLY

- Typically, a scope is run on the same machine as the rest of the MOOS Community.

MOOS App MOOS App MOOS App  
uXMS MOOSDB MOOS App  
MOOS App MOOS App MOOS App

Three Architectures MOOS Overview MOOS Messages Launching Missions **Scoping MOOS** Poking MOOS Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

59

MITMECHE MIT

## Scoping REMOTELY

- A scope may also connect to a remote machine
- Need to specify IP Address, Port Number:

```
$ uXMS mission.moos --serverhost=18.231.8.45 --serverport=9200
```

Remote Machine

Local Machine Network Remote Machine

uXMS MOOSDB MOOS App MOOS App MOOS App MOOS App MOOS App MOOS App

Three Architectures MOOS Overview MOOS Messages Launching Missions **Scoping MOOS** Poking MOOS Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

60

MITMECHE MIT

## Scoping with RealmCasting (Introduced in 2021)

Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging


Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

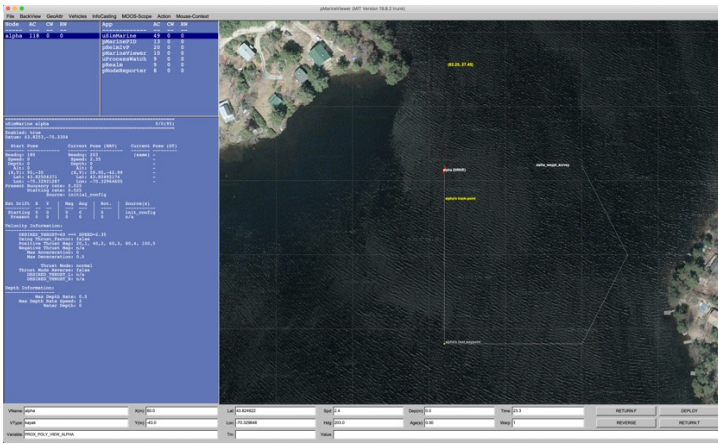
61

MITMECHE MIT

## Scoping with RealmCasting

Default window configuration upon launch







Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering


62

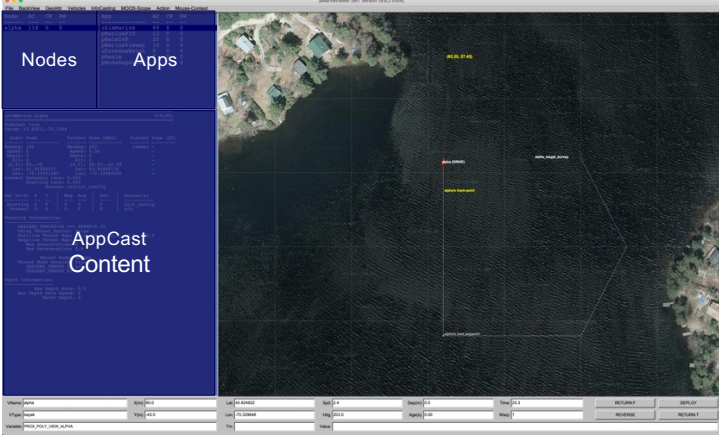


## Scoping with RealmCasting



Default window configuration upon launch





Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering


63

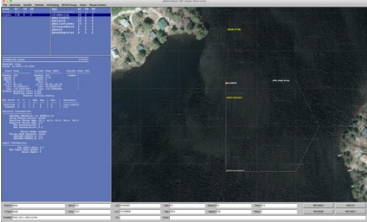


## Scoping with RealmCasting




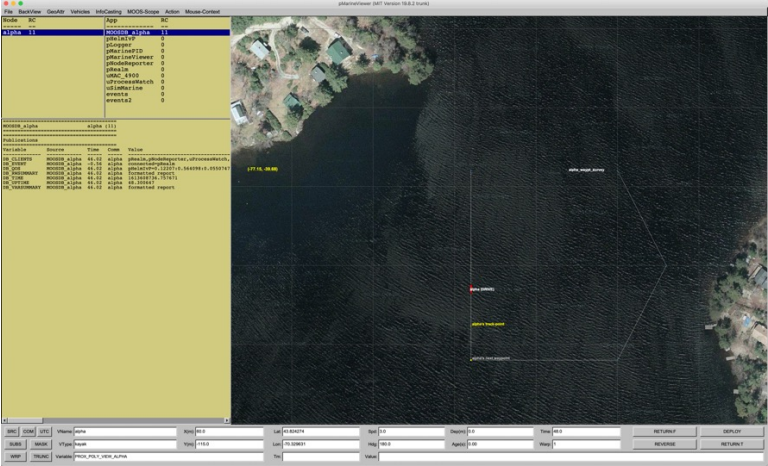
Default window configuration upon launch





Hit 'a' to Toggle





Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023
MIT Dept of Mechanical Engineering

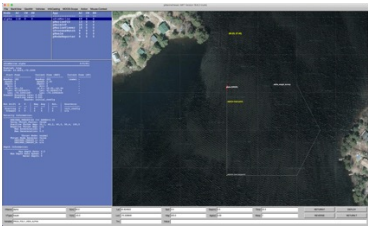
64




## Scoping with RealmCasting

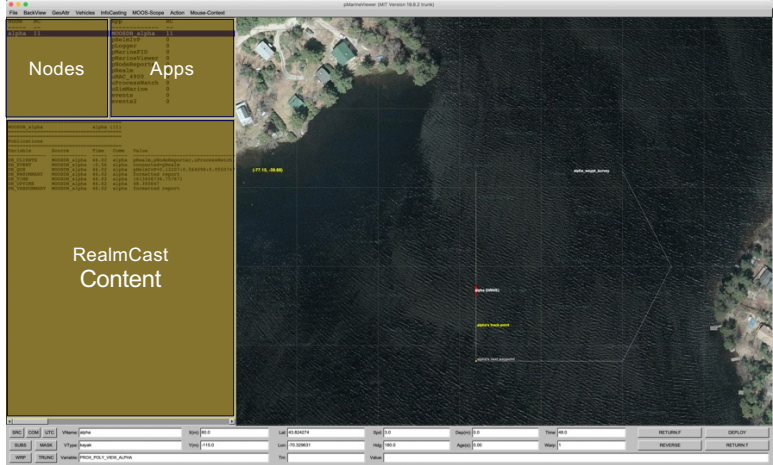


Default window configuration upon launch



Hit 'a' to Toggle





Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

65

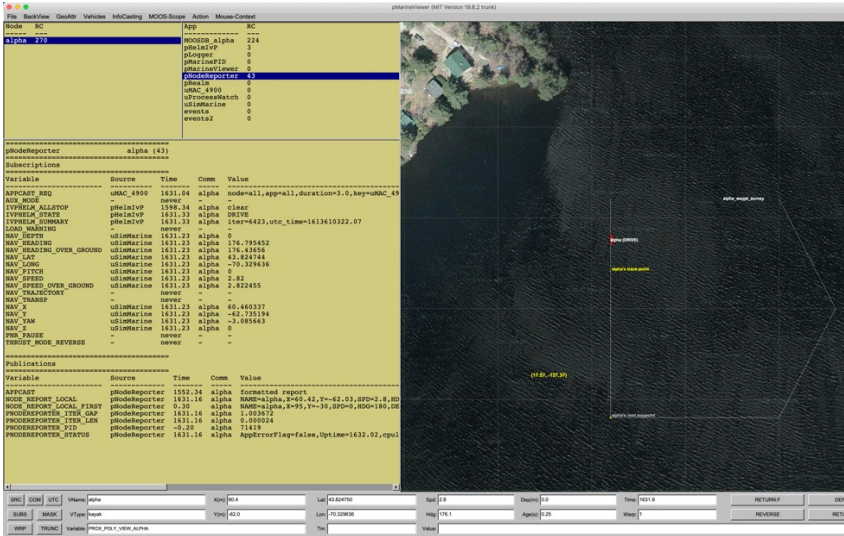


## Scoping with RealmCasting



Subscriptions

Publications



Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS

Poking MOOS

Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

66



MITMECHE Scoping with RealmCasting MIT

**pNodeReporter**

```

pNodeReporter
-----
subscriptions
-----
Variable      Source      Time      Com  Value
-----
APFCAPT_REQ   USMC_4900  1631.04   alpha node=all,app=all,duration=3.0,key=USMC_49
MDC_MODE      never      -         -
IPVPRM_ALLSTOP  pSimInVp  1596.34   alpha clear
IPVPRM_STATUS  pSimInVp  1631.23   alpha 0819E
IPVPRM_SUMMARY  pSimInVp  1631.33   alpha iter=6423,utc_time=1613610322.07
LOCN_MARINE    never      -         -
NAV_DEPTH      uSimMarine 1631.23   alpha 0
NAV_READING     uSimMarine 1631.23   alpha 176.795452
NAV_READING_OVER_GROUND  uSimMarine 1631.23   alpha 176.49564
NAV_LAT         uSimMarine 1631.23   alpha 43.824744
NAV_LONG        uSimMarine 1631.23   alpha -70.129436
NAV_FTTC        uSimMarine 1631.23   alpha 0
NAV_FREQ        uSimMarine 1631.23   alpha 2.82
NAV_SPEED_OVER_GROUND  uSimMarine 1631.23   alpha 2.822455
NAV_TRACKING    never      -         -
NAV_X           uSimMarine 1631.23   alpha 60.460337
NAV_Y           uSimMarine 1631.23   alpha -62.735194
NAV_Z           uSimMarine 1631.23   alpha -42.735194
NAV_YAW         uSimMarine 1631.23   alpha -3.085463
USMC_C          uSimMarine 1631.23   alpha 0
PWR_PAUSE      never      -         -
PWRUP_MODE_REVERSE  never      -         -
-----
Publications
-----
Variable      Source      Time      Com  Value
-----
APFCAPT       pNodeReporter 1631.16   alpha formatted report
NODE_REPORT_LOCAL  pNodeReporter 1631.16   alpha NAME=alpha,x=60.42,y=-62.63,SPD=2.8,HD
PRODREPORTER_ITER_IDN  pNodeReporter 1631.16   alpha NAME=alpha,x=95.1,y=30,SPD=0,HD=10,th
PRODREPORTER_ITER_CAP  pNodeReporter 1631.16   alpha 1.001672
PRODREPORTER_ITER_IDN  pNodeReporter 1631.16   alpha 0.000204
PRODREPORTER_PID  pNodeReporter 1631.16   alpha 1819
PRODREPORTER_STATUS  pNodeReporter 1631.16   alpha AppErrorFlag=false,UptTime=1632.02,cpu1
  
```

**NAV\_X** uSimMarine 1631.23 alpha 60.460337  
**NAV\_Y** uSimMarine 1631.23 alpha -62.735194

**NODE\_REPORT\_LOCAL**

Three Architectures MOOS Overview MOOS Messages Launching Missions Scoping MOOS **Poking MOOS** Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

67


MITMECHE Poking MOOS MIT

Poking MOOS


Three Architectures MOOS Overview MOOS Messages Launching Missions Scoping MOOS **Poking MOOS** Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

68



## Poking MOOS



**Poking** the MOOSDB :

- A write to the MOOSDB
- Implies that it is outside a typical application write to the MOOSDB


**MOOSDB**

FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	alpha
WIDTH	86
HOURS	23


Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

69



## Changing a Variable Value with a MOOS Poke



- A poke may simply alter the variable value

**MOOSDB**

FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	alpha
WIDTH	86
HOURS	23

before

**Poke**

NAME = "bravo"

**MOOSDB**


FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	bravo
WIDTH	86
HOURS	23

Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging


Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

70





## Publishing a New Variable with a MOOS Poke



• A poke may write to a new MOOS variable

MOOSDB	
FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	alpha
WIDTH	86
HOURS	23

Poke

BAND = "Beatles"


MOOSDB	
FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	alpha
WIDTH	86
HOURS	23
BAND	beatles

before


Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

71



## A Poke May Not Change an Existing Variable Type



- Once a variable is of type string – it is always a string
- Once a variable is of type double – it is always a double
- Subsequent pokes are ignored

MOOSDB	
FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	alpha
WIDTH	86
HOURS	23

Poke

WIDTH = "thin"


MOOSDB	
FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	bravo
WIDTH	86
HOURS	23

before


Three Architectures
MOOS Overview
MOOS Messages
Launching Missions
Scoping MOOS
Poking MOOS
Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

72



## Poking with uXMS



• **uPokeDB** is a command line tool for poking the MOOSDB

```
$ uPokeDB mission.moos BAND="abba" ANGLE=45
```

**MOOSDB**

FRUIT	apples
ANGLE	135
SPEED	2.8
NAME	alpha
WIDTH	86
HOURS	23

before

Poke

BAND = "abba"  
 ANGLE = 45

**MOOSDB**

FRUIT	apples
ANGLE	45
SPEED	2.8
NAME	alpha
WIDTH	86
HOURS	23
BAND	abba

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

73



## MOOS Conventions



MOOS Variables are

- Typically uppercase
- seldom use numbers
- Never have white space
- Only special character is the underscore '\_'

**Nice Variables:**

- NAV\_HEADING
- TOTAL\_POINTS
- DESIRED\_SPEED
- CLIENTS

**Ugly Variables:**

- TIME OF DAY
- basic\_value
- #ofdays
- SLIP-JOINT

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS

Poking MOOS

Data Logging

MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

74

MITMECHE MIT

# Data Logging

Three Architectures MOOS Overview MOOS Messages Launching Missions Scoping MOOS Poking MOOS Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

75

MITMECHE MIT

# Development Cycle

Code Writing  
Mission Configuration

Launch

Observe and fix


Observe and fix

In many cases, observation of the mission provides enough insight into how to fix either the code or mission configuration.


Three Architectures MOOS Overview MOOS Messages Launching Missions Scoping MOOS Poking MOOS Data Logging

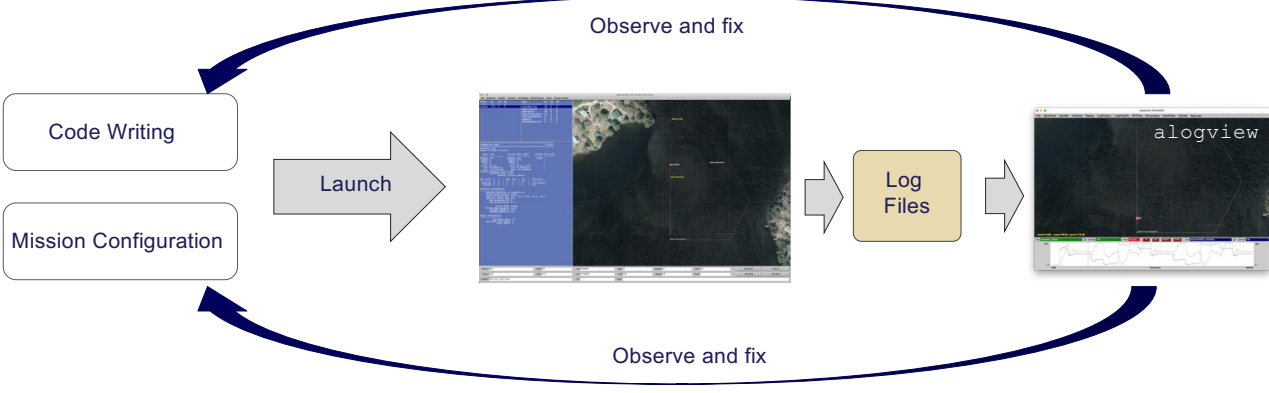
Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

76



# Development Cycle





Log files and log tools are used when the problem too complex to solve from live observation.

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

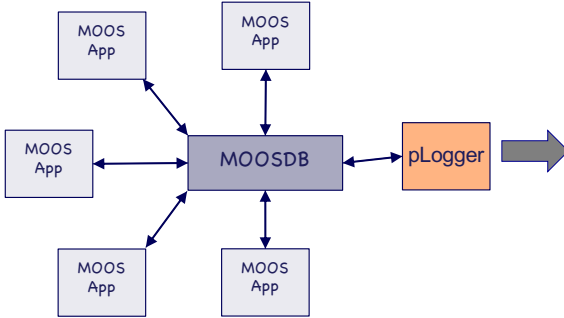
77



# Data Logging



pLogger is a MOOS application that logs all or select publications to a file.



```

=====
%% LOG FILE:      ./LOG_JAMES/JAMES.alog
%% FILE OPENED ON Tue Jan 10 22:51:43 2012
%% LOGSTART      15915046845.4
=====
0.834           DB_UPTIME           MOOSDB_james 8.16382
0.834           DB_CLIENTS         MOOSDB_james
0.994           NAV_Y              uSimMarine -25.00000
0.994           NAV_X              uSimMarine 105.00000
0.994           NAV_SPEED_SOG      uSimMarine 0.00000
0.994           NAV_SPEED          uSimMarine 0.00000
0.994           NAV_LONG           uSimMarine -70.32909
0.994           NAV_LAT            uSimMarine 43.82509
                    
```

file.alog

The logger creates at least four files for each mission:

- `file.alog` – asynchronous log (a new entry any time a post is made)
- `file.slog` – synchronous log (a sampling of variable values at fixed intervals)
- `file._bhv` – a log of critical messages
- `file._moos` – a copy of the mission file used to launch the mission.

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

78



## Log File Format



The alog file format is meant to be human readable.

```
file.alog
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% LOG FILE:      ./LOG_JAMES/JAMES.alog
%% FILE OPENED ON Tue Jan 10 22:51:43 2012
%% LOGSTART      15915046845.4 ← UTC Start Time
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
0.834            DB_UPTIME            MOOSDB_james      8.16382
0.994            NAV_Y                uSimMarine       -25.00000
0.994            NAV_X                uSimMarine       105.00000
0.994            NAV_SPEED_SOG        uSimMarine       0.00000
0.994            NAV_SPEED            uSimMarine       0.00000
0.994            NAV_LONG             uSimMarine       -70.32909
0.994            NAV_LAT              uSimMarine       43.82509
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Time Since  
UTC Start Time

MOOS  
Variable

MOOS App  
Source

Data  
Value

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

79



## Configuring the pLogger App



pLogger, like other MOOS Apps, has a configuration block in `mission.moos`.

```
ProcessConfig = pLogger
{
  AppTick      = 10
  CommstTick   = 10

  File         = RED_LOG
  PATH         = ./
  AsyncLog     = true
  FileTimeStamp = true

  // Log it all!!!!
  LogAuxSrc    = true
  WildCardLogging = true
}
```

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

80



## Wildcard Logging with Finer Control



(Exclusion by Pattern Matching)

- Wildcard logging allows you to capture everything
- Variables or variable patterns may be omitted

```

ProcessConfig = pLogger
{
  AppTick      = 10
  CommsTick    = 10

  File         = BLUE_LOG
  PATH         = ./
  AsyncLog     = true
  FileTimeStamp = true

  WildcardLogging = true
  WildcardOmitPattern = *_STATUS
}

```

Will log all MOOS variables  
*except* those ending with:  
\_STATUS

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

81



## Wildcard Logging – Playing it Safe



- What if a variable was excluded by mistake?
- Use the WildcardExclusionLog to log everything otherwise excluded

```

ProcessConfig = pLogger
{
  AppTick      = 10
  CommsTick    = 10

  File         = GREEN_LOG
  PATH         = ./
  AsyncLog     = true
  SyncLog      = true @ 0.2
  FileTimeStamp = true

  WildcardLogging = true
  WildcardOmitPattern = *_STATUS
  WildcardExclusionLog = true
}

```

Will log all MOOS variables  
ending with:  
\_STATUS  
in logfile.xlog

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering


82





## The Alog Toolbox

Tools for Modifying and Analyzing Alog Files



Command-Line log file tools:

- `aloggrep`: Prune an alog file by specifying a set of variables to keep.
- `alogscan`, `alohelm`: Examine the contents of an alog file in a short summary.
- `alogrm`: Prune an alog file by removing a given set of MOOS variables.
- `alogclip`: Prune an alog file by specifying a min/max timestamp

Each tool is a light-weight single-purpose command-line executable.  
Each tool accepts the `--help` command line option for further usage info.

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

83



## The aloggrep Tool



- The `aloggrep` tool is passed an alog file and list of variables to *keep*
- Output is to the terminal window

```
$ aloggrep file.alog NAV_X NAV_Y
```

- If provided the name of a new alog file, the new file is created
- The new file is a syntactically complete alog file (retaining header info)

```
$ aloggrep file.alog NAV_X NAV_Y newfile.alog
```

Hint

We often use this tool to help us create a focused set of data for debugging.

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


MIT Dept of Mechanical Engineering

Michael Benjamin, Spring 2023

84



# The alogrm Tool



- The `alogrm` tool is passed an alog file and list of variables to *remove*
- Output is to the terminal window

```
$ alogrm file.alog DB_STATUS
```

- If provided the name of a new alog file, the new file is created
- The new file is a syntactically complete alog file (retaining header info)

```
$ alogrm file.alog DB_STATUS newfile.alog
```

**Hint**

We often use this tool to reduce unnecessary variables to reduce alog file size

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


Michael Benjamin, Spring 2023

MIT Dept of Mechanical Engineering

85



# The alogclip Tool



- The `alogclip` tool is passed an alog file and start and end time
- *All entries in this time window will be kept.*
- Output is to the terminal window

```
$ alogclip file.alog 200 1200
```

- If provided the name of a new alog file, the new file is created
- The new file is a syntactically complete alog file (retaining header info)

```
$ alogclip file.alog 200 1200 newfile.alog
```

**Hint**

We often use this tool to reduce alog file size

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS


Poking MOOS

Data Logging


Michael Benjamin, Spring 2023

MIT Dept of Mechanical Engineering

86



## Example alogscan Output



```

Terminal - tssh - 111x30
Variable Name      Lines  Chars  Start  Stop  Sources
-----
DB_CLIENTS        181   16315  -0.57  363.44 MOOSDB_alpha
DB_TIME           358   6086   0.46   363.00 MOOSDB_alpha
DB_UPTIME         358   3119   0.46   363.00 MOOSDB_alpha
VIEW_POINT        859  123241 36.14  363.07 pHelmIvP:waypt_survey
VIEW_SBGLIST      2     130   36.14  36.14 pHelmIvP:waypt_survey,pHelmIvP:hsline
DEPLOY            1     5     12.77  12.77 pHelmIvP
DESIRED_HEADING   1295  11324  12.77  363.07 pHelmIvP
DESIRED_SPEED     1295  9065   12.77  363.07 pHelmIvP
HELM_IPF_COUNT    1293  9051   36.40  363.07 pHelmIvP
LOGGER_CMD        1     27    12.77  12.77 pHelmIvP
LOGGER_DIRECTORY  36    1152   0.72   355.24 pLogger
DESIRED_RUDDER    6241  47868  9.86   363.36 pMarinePID
DESIRED_THRUST    6248  49976  9.86   363.36 pMarinePID
MOOS_DEBUG        15    319    9.78   36.12 pMarinePID,pHelmIvP
NODE_REPORT_LOCAL 702   105140 6.71   362.92 pNodeReporter
PID_OK            1     4     18.83  18.83 uProcessWatch
PROC_WATCH_FULL_SUMMARY 1     64    18.83  18.83 uProcessWatch
PROC_WATCH_SUMMARY 68    680    18.83  357.93 uProcessWatch
UPW_EVENT         1     38    18.83  18.83 uProcessWatch
NAV_DEPTH         1419  9933   4.66   363.31 uSimMarine
NAV_HEADING       1419  12454  4.66   363.31 uSimMarine
NAV_SPEED         1419  9933   4.66   363.31 uSimMarine
NAV_X             1419  11671  4.66   363.31 uSimMarine
NAV_Y             1419  13056  4.66   363.31 uSimMarine
-----
Total variables: 24
Start/Stop Time: -0.57 / 363.44
ptsur:s1_alpha/MOOSLog_12_1_2012__06_57_53(42kool)%
    
```

Will report behavior sources on helm output.

Will report multiple sources if applicable.

Three Architectures

MOOS Overview

MOOS Messages

Launching Missions


Scoping MOOS

Poking MOOS


Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

87



## The alogview Replay Utility

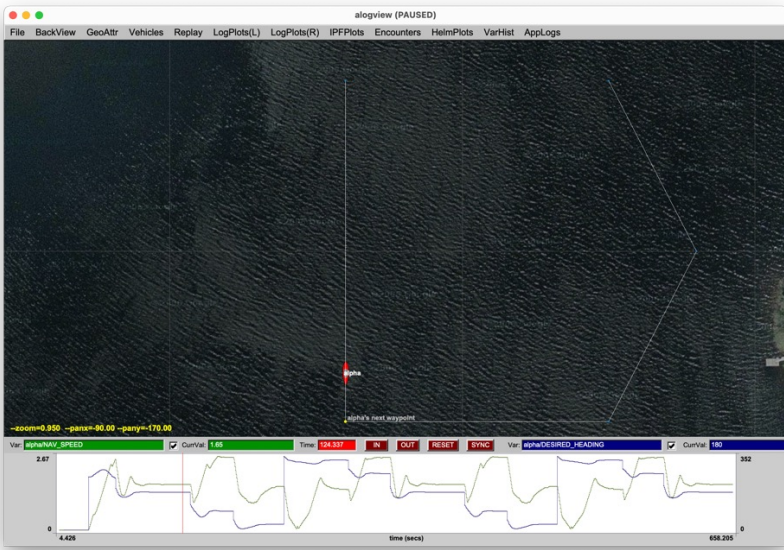


**Notables:**

- Replay the mission, start, stop, or jump around in time.
- Replay any number of vehicles, auto synchronized.
- View logged data plotted vs. time
- View Helm objective functions, helm state, stepping through time.
- View terminal output produced by any application.

```
$ alogview file.alog
```

```
$ alogview file1.alog. \
file2.alog ... fileN.alog
```



Three Architectures

MOOS Overview

MOOS Messages

Launching Missions

Scoping MOOS

Poking MOOS

Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

88

MITMECHE Variable History in Alogview MIT

alogview (PAUSED) alpha (PAUSED)

Time: 88.776

Var: alphaNAV\_SPEED

Time: 88.776

Node	Variable	Source	AutoSource
84.232	DESIRED_SPEED	pbeIn1VP	4
84.579	DESIRED_SPEED	pbeIn1VP	4
84.705	NAV_SPEED	uSisMarInex	4
84.839	DESIRED_SPEED	pbeIn1VP	4
85.099	DESIRED_SPEED	pbeIn1VP	4
85.266	NAV_SPEED	uSisMarInex	4
85.354	DESIRED_SPEED	pbeIn1VP	4
85.634	DESIRED_SPEED	pbeIn1VP	4
85.804	NAV_SPEED	uSisMarInex	4
85.890	DESIRED_SPEED	pbeIn1VP	4
86.166	DESIRED_SPEED	pbeIn1VP	4
86.315	NAV_SPEED	uSisMarInex	3.87
86.621	DESIRED_SPEED	pbeIn1VP	4
86.677	NAV_SPEED	uSisMarInex	3.74
86.843	NAV_SPEED	uSisMarInex	3.74
86.843	NAV_SPEED	uSisMarInex	3.74
86.932	DESIRED_SPEED	pbeIn1VP	4
86.932	DESIRED_SPEED	pbeIn1VP	4
87.186	DESIRED_SPEED	pbeIn1VP	4
87.389	NAV_SPEED	uSisMarInex	3.6
87.445	DESIRED_SPEED	pbeIn1VP	4
87.718	DESIRED_SPEED	pbeIn1VP	4
87.953	NAV_SPEED	uSisMarInex	3.46
87.960	DESIRED_SPEED	pbeIn1VP	4
88.251	DESIRED_SPEED	pbeIn1VP	4
88.505	DESIRED_SPEED	pbeIn1VP	4
88.517	NAV_SPEED	uSisMarInex	3.32
88.785	DESIRED_SPEED	pbeIn1VP	4
89.033	NAV_SPEED	uSisMarInex	3.19
89.037	DESIRED_SPEED	pbeIn1VP	4
89.293	DESIRED_SPEED	pbeIn1VP	4
89.545	DESIRED_SPEED	pbeIn1VP	4
89.557	NAV_SPEED	uSisMarInex	3.06
89.833	DESIRED_SPEED	pbeIn1VP	4
90.083	NAV_SPEED	uSisMarInex	3.04
90.087	DESIRED_SPEED	pbeIn1VP	4
90.344	DESIRED_SPEED	pbeIn1VP	4
90.635	NAV_SPEED	uSisMarInex	3.11
90.641	DESIRED_SPEED	pbeIn1VP	4
90.893	DESIRED_SPEED	pbeIn1VP	4
91.150	DESIRED_SPEED	pbeIn1VP	4
91.159	NAV_SPEED	uSisMarInex	3.17
91.423	DESIRED_SPEED	pbeIn1VP	4
91.680	DESIRED_SPEED	pbeIn1VP	4
91.710	NAV_SPEED	uSisMarInex	3.24
91.953	DESIRED_SPEED	pbeIn1VP	4
92.230	NAV_SPEED	uSisMarInex	3.31
92.250	DESIRED_SPEED	pbeIn1VP	4
92.561	DESIRED_SPEED	pbeIn1VP	4
92.740	NAV_SPEED	uSisMarInex	3.37
92.795	DESIRED_SPEED	pbeIn1VP	4
93.080	DESIRED_SPEED	pbeIn1VP	4
93.268	NAV_SPEED	uSisMarInex	3.44
93.336	DESIRED_SPEED	pbeIn1VP	4

Navigation: Three Architectures, MOOS Overview, MOOS Messages, Launching Missions, Scoping MOOS, Poking MOOS, Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

89

MITMECHE END MIT

Navigation: Three Architectures, MOOS Overview, MOOS Messages, Launching Missions, Scoping MOOS, Poking MOOS, Data Logging

Michael Benjamin, Spring 2023 MIT Dept of Mechanical Engineering

90