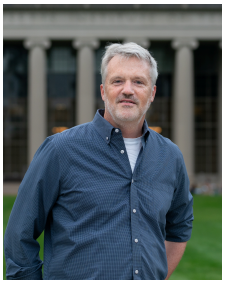
MITMECHE

An Introduction to Robot Autonomy

ROBOTICS RESEARCH CENTER (RRC)

with
MOOS-IvP
and
Aquaticus

Lecture 2: MOOS Overview



Michael Benjamin, PhD
MIT Dept of Mechanical Eng.
mikerb@mit.edu

MOOS-IvP Supported by ONR Code 311 since 2000

Prof. Michael "Misha" Novitzky
United States Military Academy
michael.novitzky@westpoint.edu

Aquaticus Supported by ONR, DARPA, Battelle and the Army Research Lab

1

MITMECHE

MOOS Overview

ROBOTICS RESEARCH CENTER (RRC)

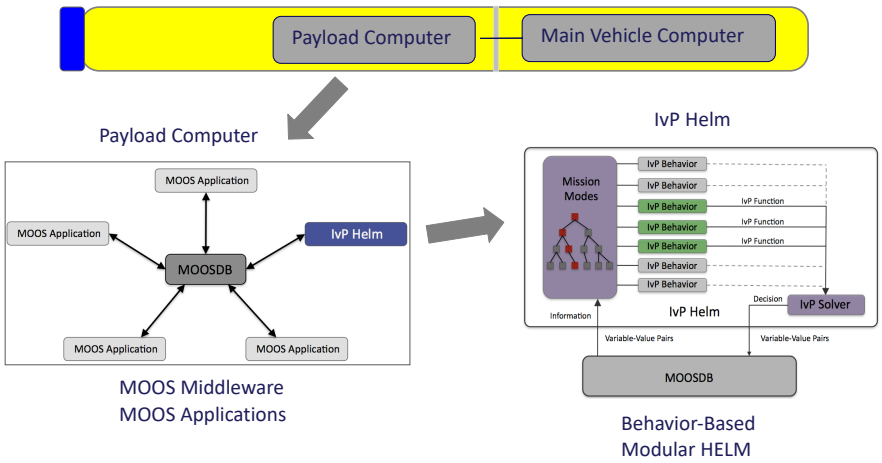
- Nested Software – MOOS, MOOS-IvP, Aquaticus
- MOOS App Structure
- MOOS Messaging
- Mission and Autonomy configuration
- Scoping and Poking the MOOSDB
- Data Logging and Post-mission Tools
  - MOOS developed by Paul Newman, as an MIT post-doc and now Oxford Professor
  - Initial development 2000-2003 on MIT Bluefin Odyssey II UUV

2

MITMECHE

Payload UUV Autonomy
(3 Architecture Principles)

ROBOTICS RESEARCH CENTER (RRC)



Michael Benjamin, mikerb@mit.edu

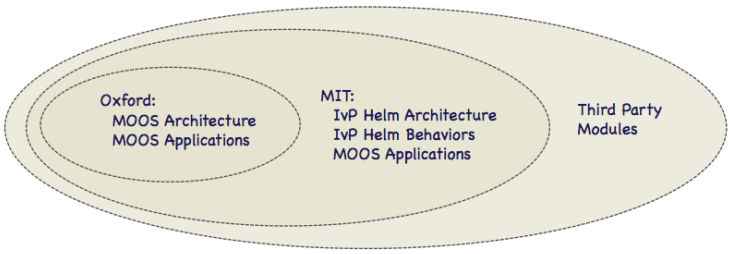
3

MITMECHE

Nested Repositories

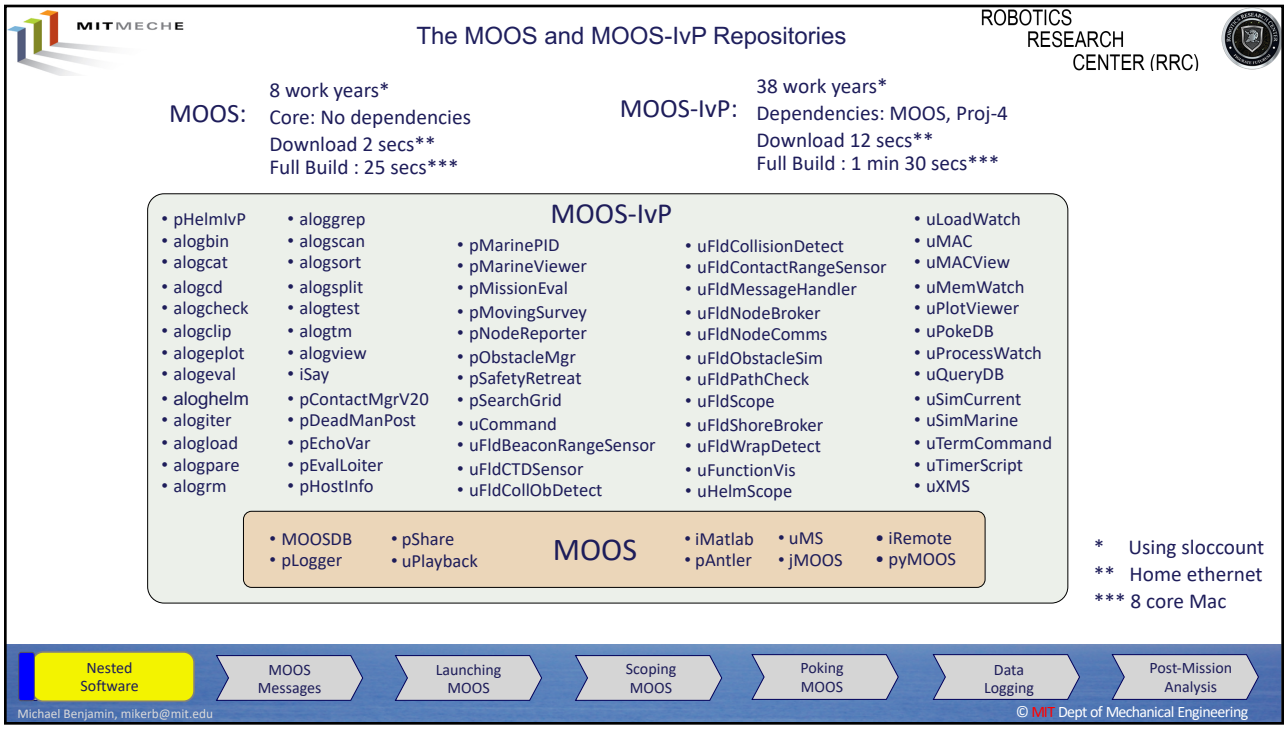
ROBOTICS RESEARCH CENTER (RRC)

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

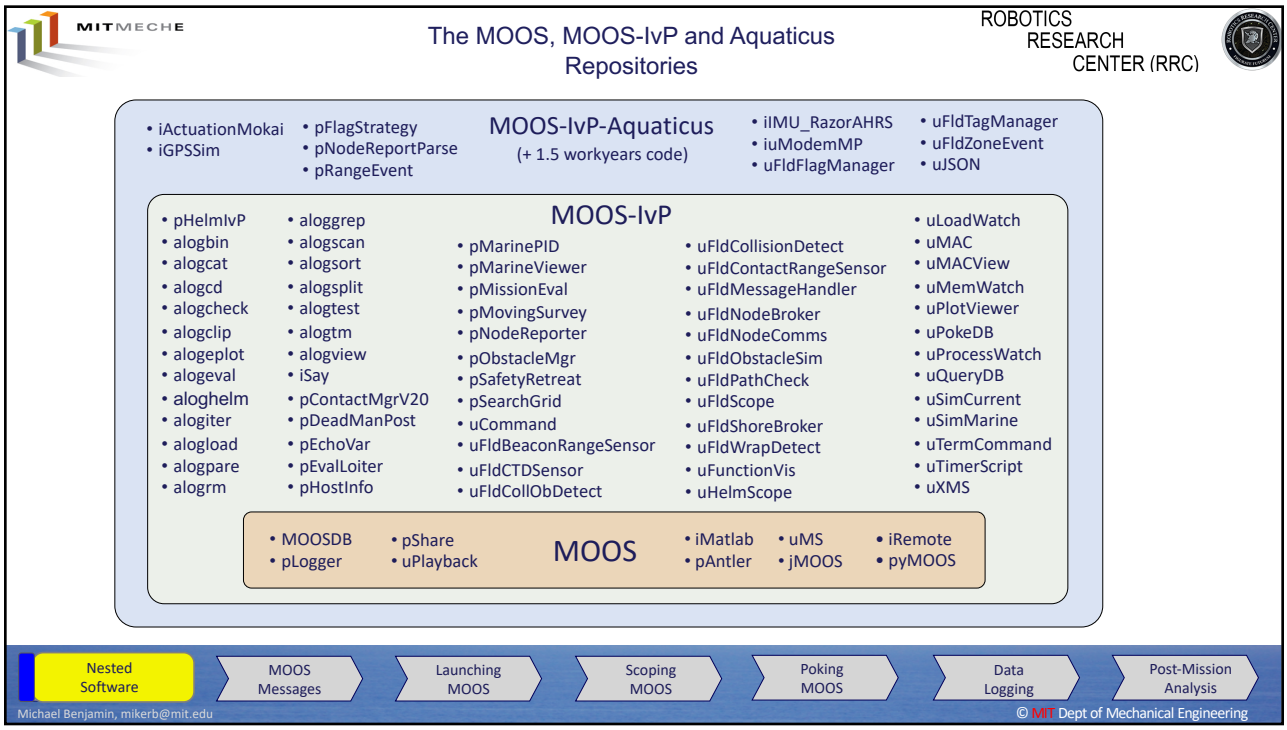


Michael Benjamin, mikerb@mit.edu

4


MITMECHE
The MOOS and MOOS-IvP Repositories
ROBOTICS RESEARCH CENTER (RRC)
MOOS:
8 work years*
Core: No dependencies
Download 2 secs**
Full Build : 25 secs***
MOOS-IvP:
38 work years*
Dependencies: MOOS, Proj-4
Download 12 secs**
Full Build : 1 min 30 secs***
MOOS-IvP
• pHelmIvP
• alogbin
• alogcat
• alogcd
• alogcheck
• alogclip
• alogeplot
• alogeval
• aloghelm
• alogiter
• alogload
• alogpare
• alogrm
• aloggrep
• alogscan
• alogsort
• alogsplit
• alogtest
• alogtm
• alogview
• iSay
• pContactMgrV20
• pDeadManPost
• pEchoVar
• pEvalLoiter
• pHostInfo
• pMarinePID
• pMarineViewer
• pMissionEval
• pMovingSurvey
• pNodeReporter
• pObstacleMgr
• pSafetyRetreat
• pSearchGrid
• uCommand
• uFldBeaconRangeSensor
• uFldCTDSensor
• uFldCollObDetect
• uFldCollisionDetect
• uFldContactRangeSensor
• uFldMessageHandler
• uFldNodeBroker
• uFldNodeComms
• uFldObstacleSim
• uFldPathCheck
• uFldScope
• uFldShoreBroker
• uFldWrapDetect
• uFunctionVis
• uHelmScope
• uLoadWatch
• uMAC
• uMACView
• uMemWatch
• uPlotViewer
• uPokeDB
• uProcessWatch
• uQueryDB
• uSimCurrent
• uSimMarine
• uTermCommand
• uTimerScript
• uXMS
• MOOSDB
• pLogger
• pShare
• uPlayback
MOOS
• iMatlab
• pAntler
• uMS
• jMOOS
• iRemote
• pyMOOS
* Using sloccount
** Home ethernet
*** 8 core Mac
Nested Software
MOOS Messages
Launching MOOS
Scoping MOOS
Poking MOOS
Data Logging
Post-Mission Analysis
Michael Benjamin, mikerb@mit.edu
© MIT Dept of Mechanical Engineering

5


MITMECHE
The MOOS, MOOS-IvP and Aquaticus Repositories
ROBOTICS RESEARCH CENTER (RRC)
MOOS-IvP-Aquaticus
(+ 1.5 workyears code)
• iActuationMokai
• iGPSSim
• pFlagStrategy
• pNodeReportParse
• pRangeEvent
• iIMU_RazorAHRS
• iuModemMP
• uFldFlagManager
• uFldTagManager
• uFldZoneEvent
• uJSON
MOOS-IvP
• pHelmIvP
• alogbin
• alogcat
• alogcd
• alogcheck
• alogclip
• alogeplot
• alogeval
• aloghelm
• alogiter
• alogload
• alogpare
• alogrm
• aloggrep
• alogscan
• alogsort
• alogsplit
• alogtest
• alogtm
• alogview
• iSay
• pContactMgrV20
• pDeadManPost
• pEchoVar
• pEvalLoiter
• pHostInfo
• pMarinePID
• pMarineViewer
• pMissionEval
• pMovingSurvey
• pNodeReporter
• pObstacleMgr
• pSafetyRetreat
• pSearchGrid
• uCommand
• uFldBeaconRangeSensor
• uFldCTDSensor
• uFldCollObDetect
• uFldCollisionDetect
• uFldContactRangeSensor
• uFldMessageHandler
• uFldNodeBroker
• uFldNodeComms
• uFldObstacleSim
• uFldPathCheck
• uFldScope
• uFldShoreBroker
• uFldWrapDetect
• uFunctionVis
• uHelmScope
• uLoadWatch
• uMAC
• uMACView
• uMemWatch
• uPlotViewer
• uPokeDB
• uProcessWatch
• uQueryDB
• uSimCurrent
• uSimMarine
• uTermCommand
• uTimerScript
• uXMS
• MOOSDB
• pLogger
• pShare
• uPlayback
MOOS
• iMatlab
• pAntler
• uMS
• jMOOS
• iRemote
• pyMOOS
Nested Software
MOOS Messages
Launching MOOS
Scoping MOOS
Poking MOOS
Data Logging
Post-Mission Analysis
Michael Benjamin, mikerb@mit.edu
© MIT Dept of Mechanical Engineering

6

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

The MOOS, MOOS-IvP and Aquaticus Repositories

MOOS-IvP-Aquaticus

- pFlagStrategy
- pNodeReportParse
- pRangeEvent
- uFldFlagManager
- uFldTagManager
- uFldZoneEvent

MOOS-IvP

- pHelmIvP
- alogclip
- alogrm
- aloggrep
- alogscan
- alogview
- iSay
- pContactMgrV20
- pDeadManPost
- pHostInfo
- pMarinePID
- pMarineViewer
- pMissionEval
- pNodeReporter
- uFldMessageHandler
- uFldNodeBroker
- uFldNodeComms
- uFldShoreBroker
- uLoadWatch
- uPokeDB
- uProcessWatch
- uQueryDB
- uSimMarine
- uTimerScript
- uXMS

MOOS

- MOOSDB
- pLogger
- pShare

Michael Benjamin, mikerb@mit.edu

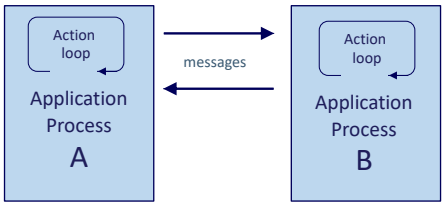
7

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

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



Michael Benjamin, mikerb@mit.edu

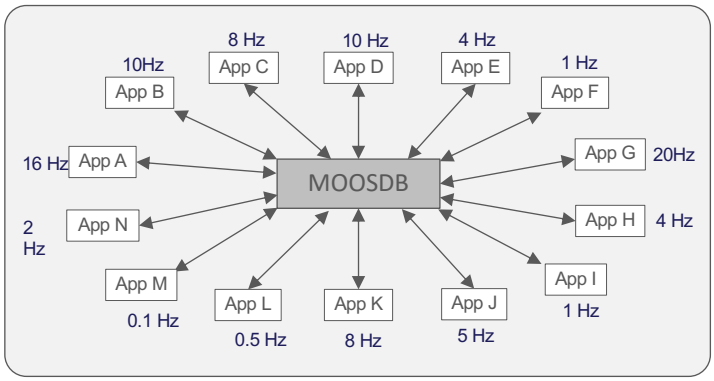
8

MITMECHE

A MOOS Community

ROBOTICS RESEARCH CENTER (RRC)

- The application frequency is set upon, at mission planning time
- If the CPU cannot meet the requested frequency, it will do the best it can
- The user can implement "back-off" within the app itself if desired.



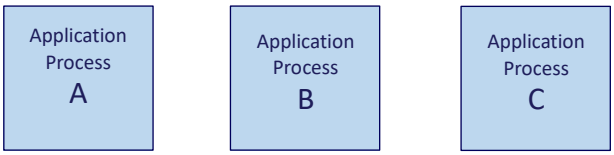
Michael Benjamin, mikerb@mit.edu

9

MITMECHE

The Beauty of Separate Processes

ROBOTICS RESEARCH CENTER (RRC)



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

Michael Benjamin, mikerb@mit.edu

10

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

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

Michael Benjamin, mikerb@mit.edu

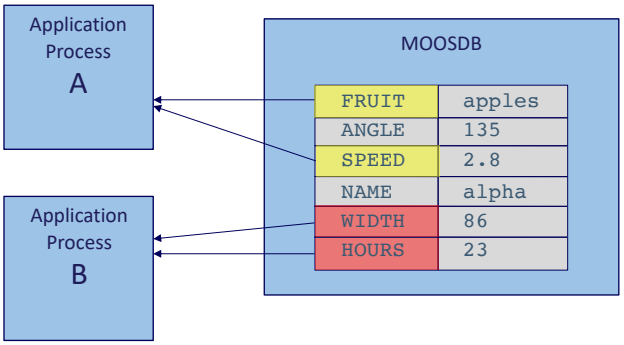
11

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

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)

Michael Benjamin, mikerb@mit.edu

12

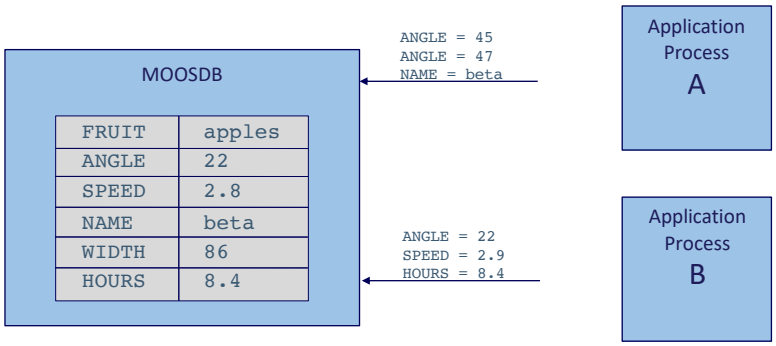
MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

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.



Michael Benjamin, mikerb@mit.edu

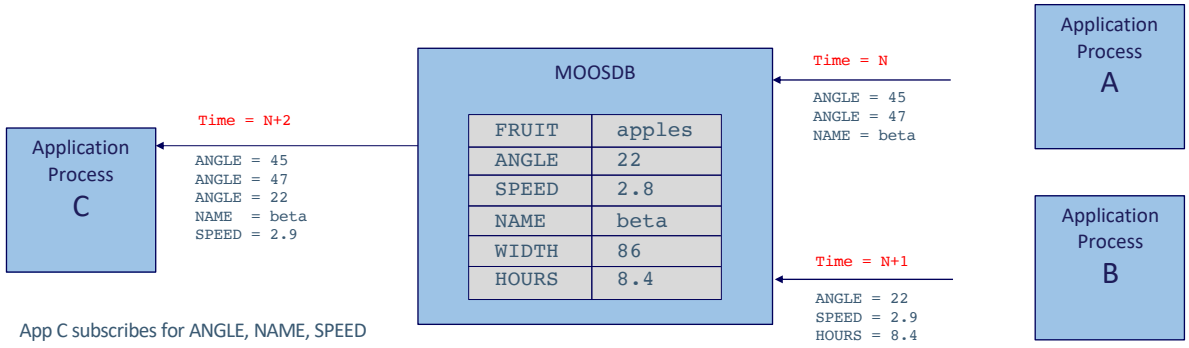
13

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

MOOS Apps Publish to the MOOSDB

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



App C subscribes for ANGLE, NAME, SPEED

Michael Benjamin, mikerb@mit.edu

14

MOOS Messages

Michael Benjamin, mikerb@mit.edu

15

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

Michael Benjamin, mikerb@mit.edu

16

MITMECHE

MOOS Message Examples

ROBOTICS RESEARCH CENTER (RRC)

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

Michael Benjamin, mikerb@mit.edu

17

MITMECHE

MOOS Messages

ROBOTICS RESEARCH CENTER (RRC)

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 (STRING or DOUBLE or BINARY)
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

Michael Benjamin, mikerb@mit.edu

18

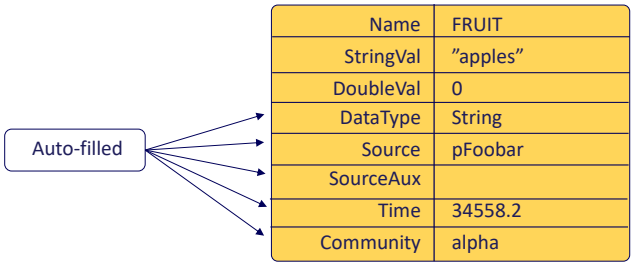
Posting MOOS Messages

ROBOTICS RESEARCH CENTER (RRC)

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:



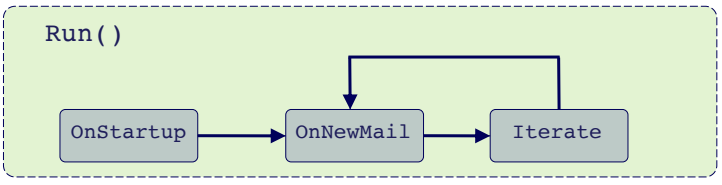
Michael Benjamin, mikerb@mit.edu

19

Reading MOOS Messages

ROBOTICS RESEARCH CENTER (RRC)

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



The Flow of Control for all MOOS Apps

Michael Benjamin, mikerb@mit.edu

20

Launching MOOS

21

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"
  Name look up is                    off
  Asynchronous support is            on
  Connect to this server on port    9000
----------------------------------------------------
network performance data published on localhost:9020
listen with "nc -u -lk 9020"
```

Default Community Name

Default Port Number

Default IP Address

22

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

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"
```


Michael Benjamin, mikerb@mit.edu

23

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

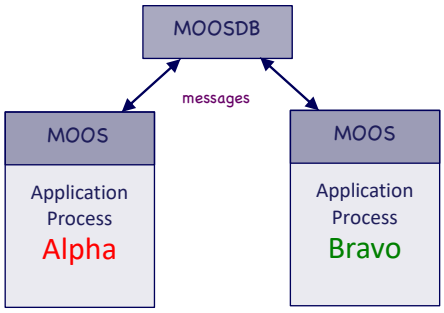
Launching MOOS and Mission Configuration

- 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
}
```



Michael Benjamin, mikerb@mit.edu

24

MITMECHE

MOOS Mission Configuration

ROBOTICS RESEARCH CENTER (RRC)

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.

Michael Benjamin, mikerb@mit.edu

25

MITMECHE

MOOS Mission Configuration

ROBOTICS RESEARCH CENTER (RRC)

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.

Nested Software | MOOS Messages | Launching MOOS | Scoping MOOS | Poking MOOS | Data Logging | Post-Mission Analysis

Michael Benjamin, mikerb@mit.edu

© MIT Dept of Mechanical Engineering

26

Scoping MOOS

27

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

28

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

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

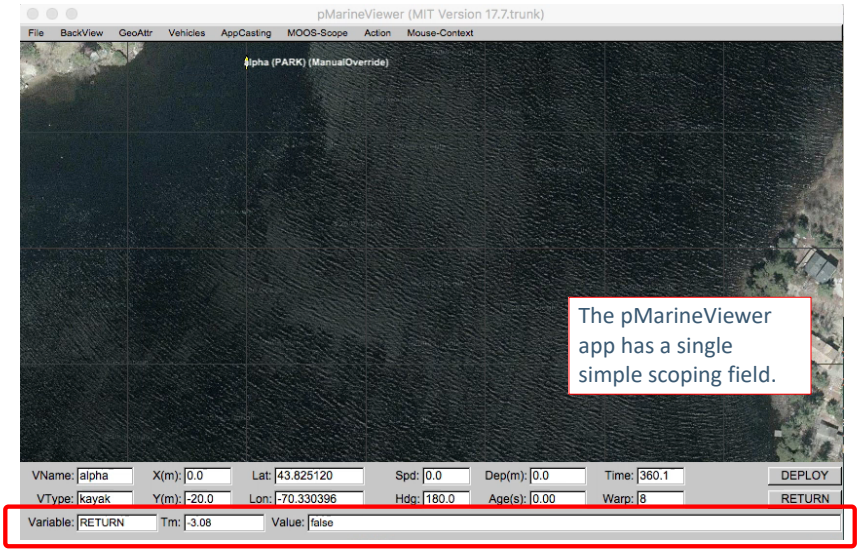
Michael Benjamin, mikerb@mit.edu

29

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

A Simple Single Scope in pMarineViewer

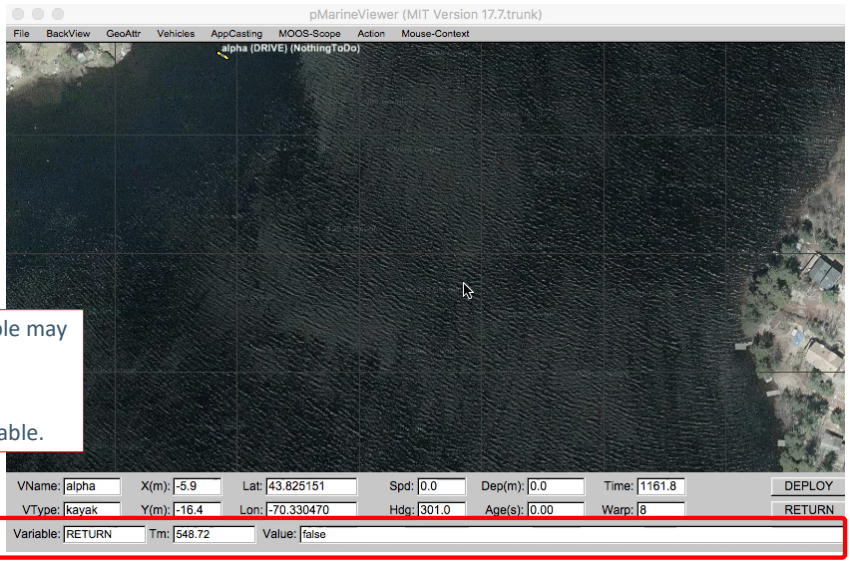


Michael Benjamin, mikerb@mit.edu

30

Changing the Scope Variable in pMarineViewer

ROBOTICS RESEARCH CENTER (RRC)



The scope variable may be changed:

Hit SHIFT-'A'

Enter a new variable.

Michael Benjamin, mikerb@mit.edu

31

The uXMS Scope List

ROBOTICS RESEARCH CENTER (RRC)

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
```


Michael Benjamin, mikerb@mit.edu

32

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

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
  ----------------   ----------      ----------  ----------    ----------- (7)
  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

Name of the app that *last* published the variable

Time of the *last* publication

The community of the app that made the *last* publication

The value of the *last* publication

Michael Benjamin, mikerb@mit.edu

33

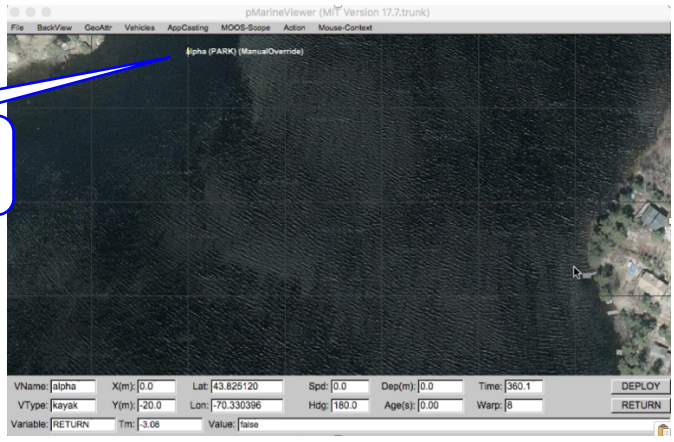
MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

Scoping on the Alpha Example Mission with uXMS

Launch the mission

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



The vehicle starts motionless at the start position at point (0,-20) in local coordinates.

Michael Benjamin, mikerb@mit.edu

34

MITMECHE

Scoping on the Alpha Example Mission with uXMS

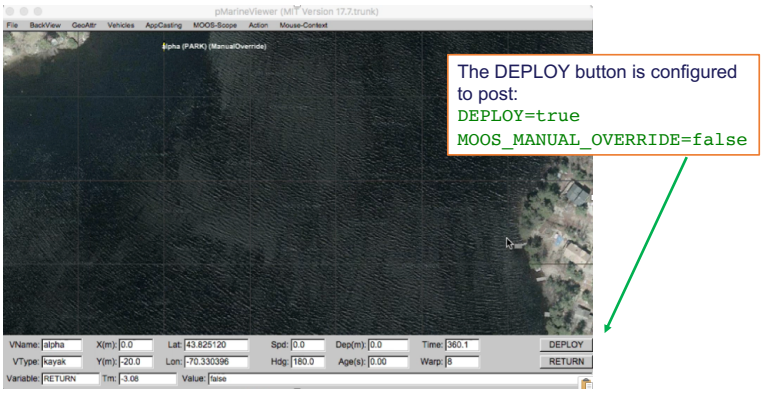
ROBOTICS RESEARCH CENTER (RRC)

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
```



Launch the mission. Hit the DEPLOY button.

Michael Benjamin, mikerb@mit.edu

35

MITMECHE

Scoping on the Alpha Example Mission with uXMS

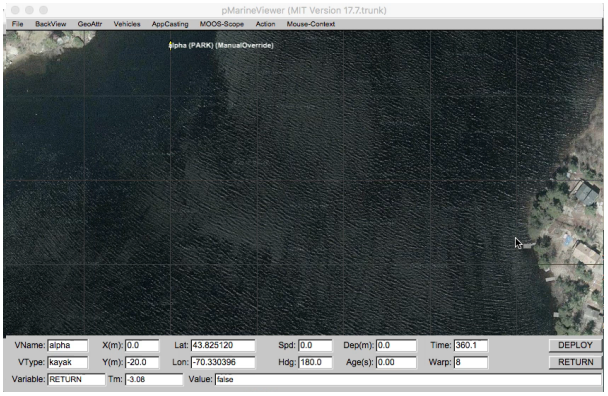
ROBOTICS RESEARCH CENTER (RRC)

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
```



Launch the mission. Hit the DEPLOY button.

Nested Software | MOOS Messages | Launching MOOS | Scoping MOOS | Poking MOOS | Data Logging | Post-Mission Analysis

Michael Benjamin, mikerb@mit.edu

© MIT Dept of Mechanical Engineering

36

MITMECHE

Scoping on the Alpha Example Mission with uXMS

ROBOTICS RESEARCH CENTER (RRC)

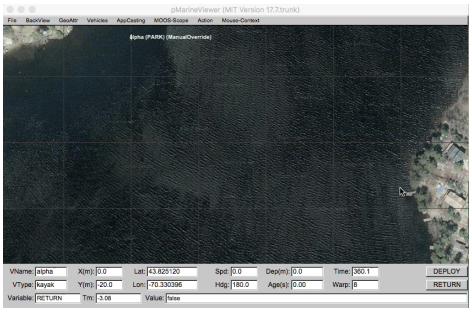
Launch the mission

```
$ 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
```



Nested Software | MOOS Messages | Launching MOOS | Scoping MOOS | Poking MOOS | Data Logging | Post-Mission Analysis

Michael Benjamin, mikerb@mit.edu

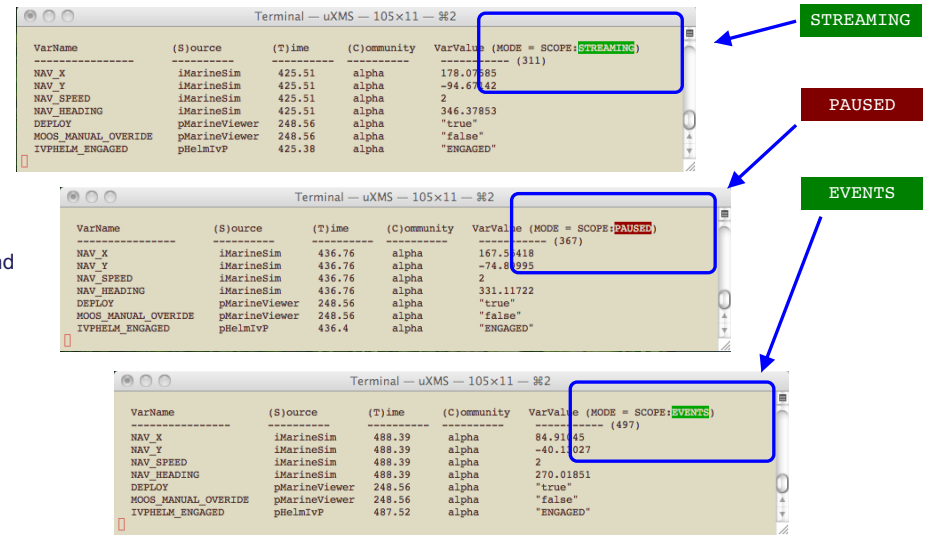
37

MITMECHE

The uXMS Utility Refresh Mode Indicator

ROBOTICS RESEARCH CENTER (RRC)

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



Nested Software | MOOS Messages | Launching MOOS | Scoping MOOS | Poking MOOS | Data Logging | Post-Mission Analysis

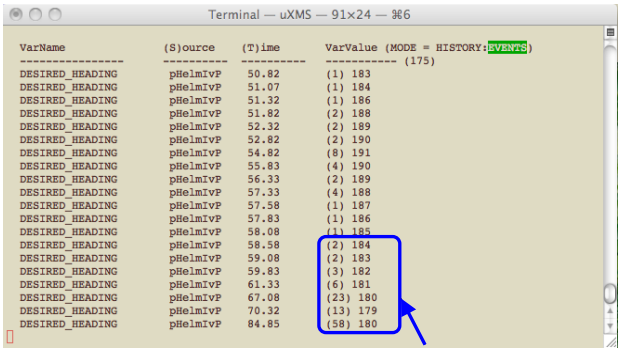
Michael Benjamin, mikerb@mit.edu

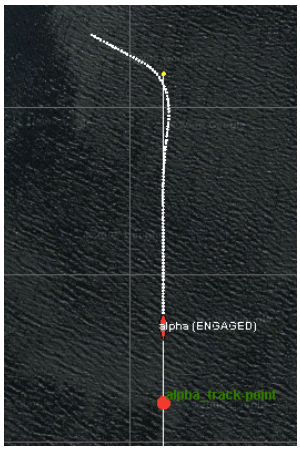
38

The uXMS Utility: The "History" Content Mode

ROBOTICS RESEARCH CENTER (RRC)

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





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

Nested Software | MOOS Messages | Launching MOOS | Scoping MOOS | Poking MOOS | Data Logging | Post-Mission Analysis

Michael Benjamin, mikerb@mit.edu

39

Setting the Scope List by App Name

ROBOTICS RESEARCH CENTER (RRC)

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
```

Nested Software | MOOS Messages | Launching MOOS | Scoping MOOS | Poking MOOS | Data Logging | Post-Mission Analysis

Michael Benjamin, mikerb@mit.edu

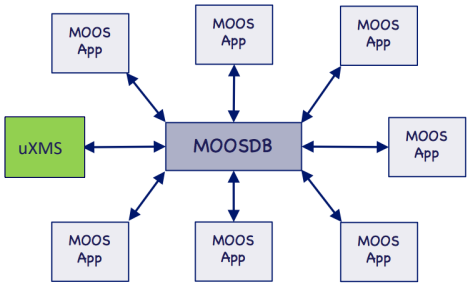
40

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

Scoping LOCALLY

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



Michael Benjamin, mikerb@mit.edu

41

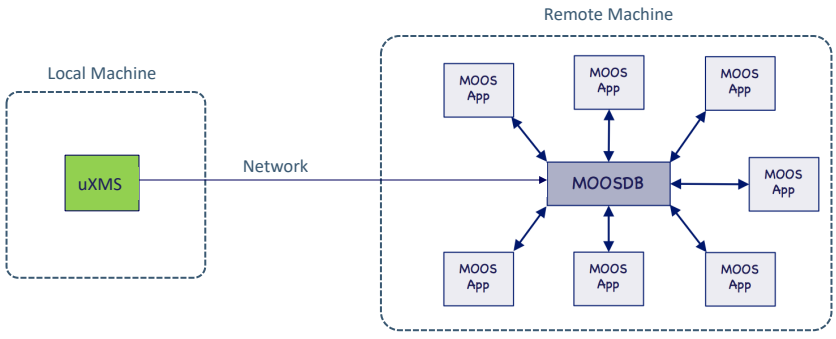
MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

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
```



Michael Benjamin, mikerb@mit.edu

42

Poking MOOS

43

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

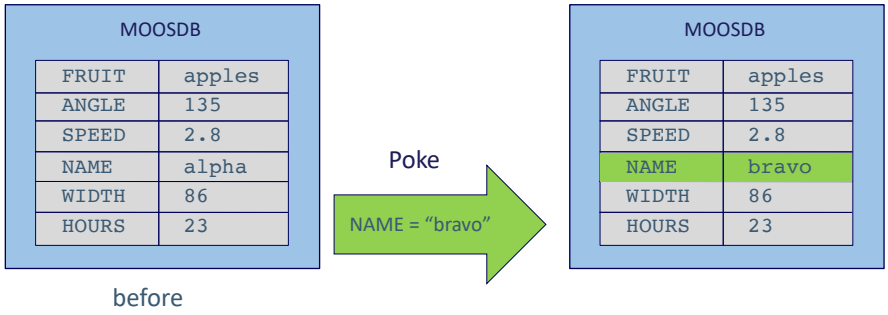
44

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

Changing a Variable Value with a MOOS Poke

- A poke may simply alter the variable value



before

Michael Benjamin, mikerb@mit.edu

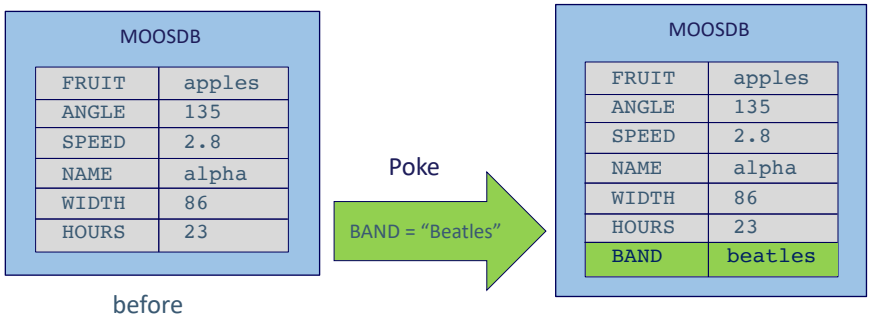
45

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

Publishing a New Variable with a MOOS Poke

- A poke may write to a new MOOS variable



before

Michael Benjamin, mikerb@mit.edu

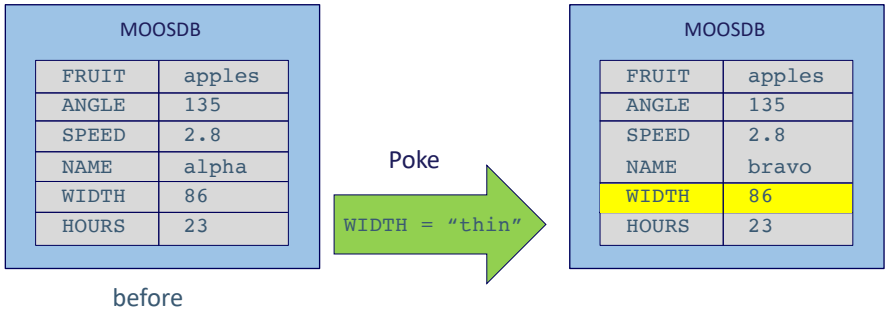
46

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

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



before

Michael Benjamin, mikerb@mit.edu

47

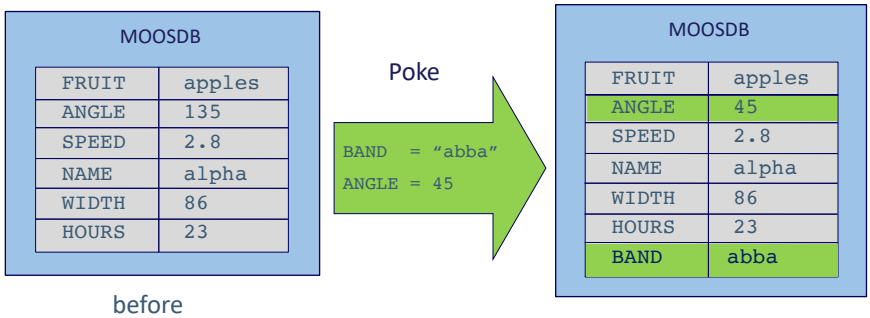
MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

Poking with uXMS

- uXMS is a command line tool for poking the MOOSDB

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



before

Michael Benjamin, mikerb@mit.edu

48

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

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

Michael Benjamin, mikerb@mit.edu

49

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

Data Logging

Michael Benjamin, mikerb@mit.edu

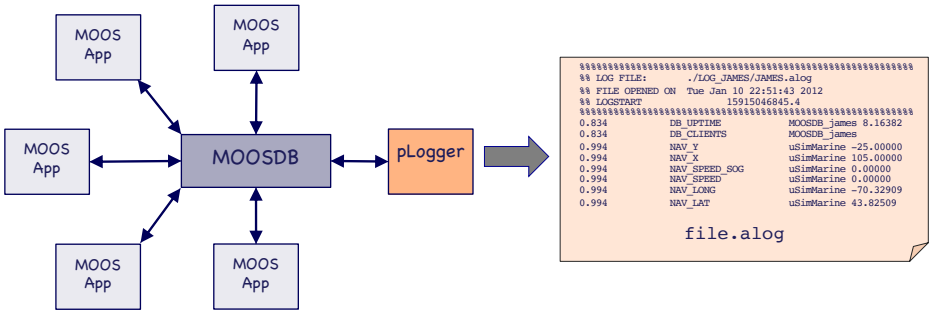
50

MITMECHE

Data Logging

ROBOTICS RESEARCH CENTER (RRC)

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



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.

Michael Benjamin, mikerb@mit.edu

51

MITMECHE

Log File Format

ROBOTICS RESEARCH CENTER (RRC)

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
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
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
```

UTC Start Time

Time Since UTC Start Time

MOOS Variable

MOOS App Source

Data Value

Michael Benjamin, mikerb@mit.edu

52

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

Configuring the pLogger App

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

```
ProcessConfig = pLogger
{
  AppTick       = 10
  CommsTick     = 10

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

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


Michael Benjamin, mikerb@mit.edu

53

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

Wildcard Logging with Finer Control

(Exclusion by Pattern Matching)

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

```
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

Michael Benjamin, mikerb@mit.edu

54

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

Nested Software | MOOS Messages | Launching MOOS | Scoping MOOS | Poking MOOS | **Data Logging** | Post-Mission Analysis

Michael Benjamin, mikerb@mit.edu

55

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`, `aloghelm`: Examine the contents of a 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.

Nested Software | MOOS Messages | Launching MOOS | Scoping MOOS | Poking MOOS | Data Logging | **Post-Mission Data Handling**

Michael Benjamin, mikerb@mit.edu

56

MITMECHE

The aloggrep Tool

ROBOTICS RESEARCH CENTER (RRC)

- 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.

Michael Benjamin, mikerb@mit.edu

57

MITMECHE

The alogrm Tool

ROBOTICS RESEARCH CENTER (RRC)

- 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 unecessary variables to reduce alog file size

Michael Benjamin, mikerb@mit.edu

58

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

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

Michael Benjamin, mikerb@mit.edu

59

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

Example *alogscan* Output

Terminal — tcsh — 111×30

```
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_SEGLIST                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
PLOGGER_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.

Michael Benjamin, mikerb@mit.edu

60

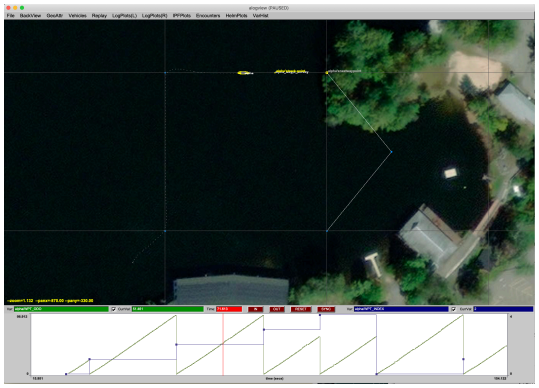
MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

The alogview Tool

- THE most import tool for post-mission analysis.
- Allows playback of the mission, including multiple vehicles.
- Step forward or backward through time
- Scope any variable posted by any app at any time.

```
$ alogview file.alog file2.alog ...
```



Michael Benjamin, mikerb@mit.edu

61

MITMECHE

ROBOTICS RESEARCH CENTER (RRC)

END

Michael Benjamin, mikerb@mit.edu

62