The uMAC Utilities

June 2018

Michael Benjamin, mikerb@mit.edu
Department of Mechanical Engineering
MIT, Cambridge MA 02139

1 Overview

In this section the utilities for viewing appcasts and/or realmcasts are discussed. They include:

- The `uMACView` utility: A GUI tool containing navigation tools for viewing appcasts or realmcasts across multiple vehicles or nodes. A snapshot is shown in Figure 1.
- The `uMAC` utility: A utility implement in the terminal window. Simpler in the interface, but notable in that it allows a user to remotely launch the tool on deployed vehicle.
- `uMACView` integrated with `pMarineViewer`: An interface nearly identical with `uMACView` fully integrated within `pMarineViewer`, convenient for those already using `pMarineViewer`.

Note: Starting with the first release after 2019's Release 19.8, `uMACView` and `pMarineViewer` were augmented to include realmcasting in addition to appcasting. The term *infocasting* is the general term referring to either appcasting or realmcasting. To use realmcasting, an additional app, `pRealm` needs to be running in each MOOS community, typically on the shoreside and in each vehicle community. More information on `pRealm` can be found in [1]

The `uMAC` utility currently only support appcasting output.

2 The uMACView Utility

The `uMACView` utility is an appcast or realmcast viewer with a graphical user interface using the FLTK library. It contains three panes as shown in Figure 1. The bottom pane renders incoming appcasts or realmcasts. The top right pane lists possible applications for selecting appcast or realmcast viewing for the present node. The upper left pane shows all presently known nodes from which appcasts or realmcasts have been received. In realmcasting, the upper left pane may also offer a number of variable clusters to select from.

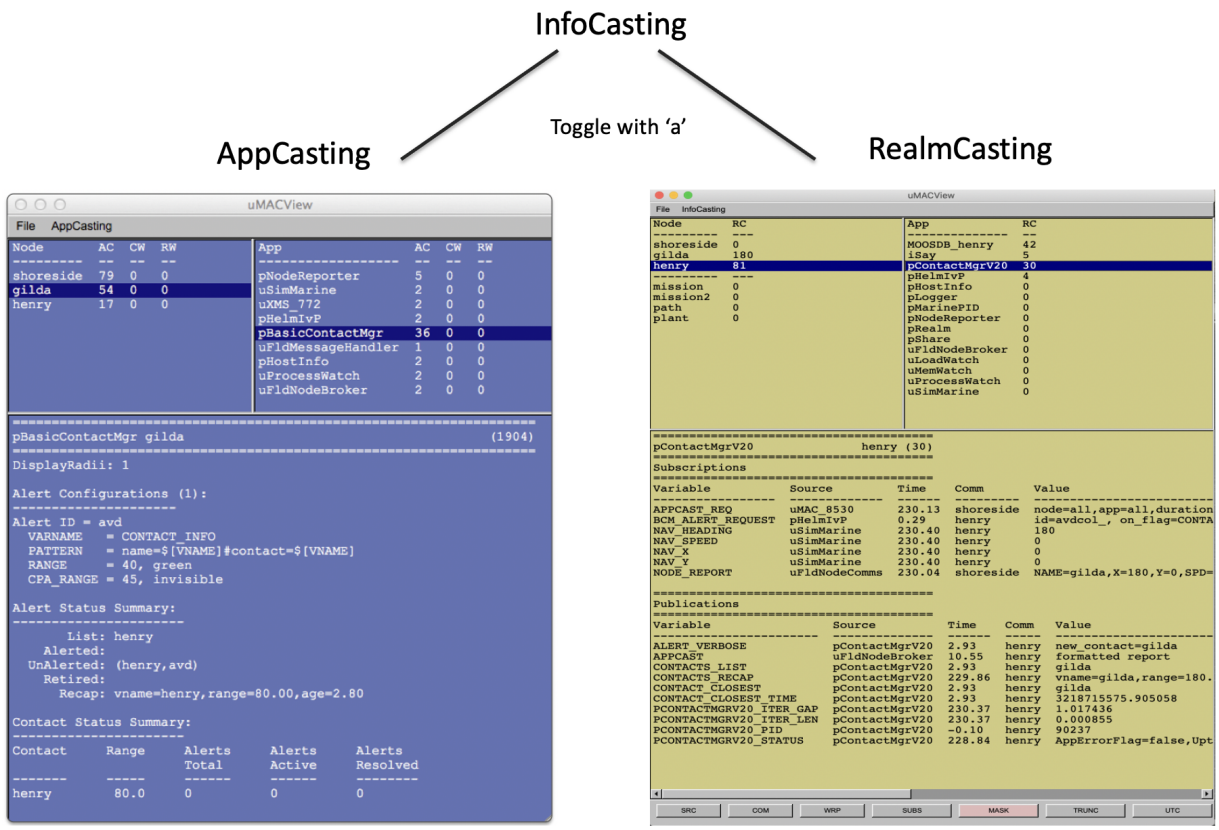

Figure 1: The uMACView utility receives appcasts or realmcasts from perhaps several different nodes and applications on each node. The interface allows the user to select a node and application for viewing the selected application's latest appcast or realmcast. The with appcasting, the viewer will raise alerts from non-selected nodes and applications when or if a run warning occurs. To toggle between appcasting and realmcasting, the `'a'` key is used.

The content mode is either *appcasting* or *realmcasting*. To toggle between the two modes, the `'a'` key is used. Alternatively the startup mode can be set with the `content_mode` configuration parameter, which is set to either `appcast` or `realmcast`, with the former being the default.

2.1 AppCasting

The content of the *appcast pane* is solely determined by the appcast content itself. From the viewer's perspective it is simply pushing a list of strings to a browser pane. Even the formatting of the

header lines showing the application name, node name and application iteration, are formatted by a method defined over the AppCast class.

The content of the `application pane` lists the applications known thus far to the viewer from incoming appcasts for a particular node. The order is shown by the order in which they were received. The three columns, `"AC"`, `"CW"`, and `"RW"`, show the number of appcasts received from the application, and the number of run warnings and configuration warnings in the latest appcast.

The content of the *node pane* in the upper left lists the nodes known thus far to the viewer from incoming appcasts. The order is shown by the order in which they were received. The three columns, `"AC"`, `"CW"`, and `"RW"`, show the number of appcasts received from a given node for *all* applications from that node, as well as the sum of all run warnings and configuration warnings for all applications received from the given node.

2.2 RealmCasting

The content of the *realmcast pane* is fed from an app called `pRealm` running on the node selected in the node pane, and related to the application selected in the upper right app pane. A typical realmcast report is shown in Figure 2:

```
=======================================
pContactMgrV20              gilda (140)
=======================================
Subscriptions
=======================================
Variable           Source         Time    Comm       Value
-----------------  -------------  ------  ---------  ----------------------------------------------------------------------------------------
APPCAST_REQ        uMAC_220       241.52  shoreside  node=all,app=all,duration=3.0,key=uMAC_220:app,thresh=run_warning
BCM_ALERT_REQUEST  pHelmIvP       0.28    gilda      id=avdcol_, on_flag=CONTACT_INFO=name=$[VNAME] # contact=$[VNAME],alert_range=80, cpa_rang
                                                     e=85,match_type=mokai
NAV_HEADING        uSimMarine     242.21  gilda      180
NAV_SPEED          uSimMarine     242.21  gilda      0
NAV_X              uSimMarine     242.21  gilda      180
NAV_Y              uSimMarine     242.21  gilda      0
NODE_REPORT        uFldNodeComms  241.73  shoreside  NAME=henry,X=0,Y=0,SPD=0,HDG=180,TYPE=kayak,MODE=PARK,ALLSTOP=ManualOverride,INDEX=479,TIM
                                                     E=3218502913.32,LENGTH=4

=======================================
Publications
=======================================
Variable                  Source          Time    Comm   Value
------------------------  --------------  ------  -----  ----------------------------------------------------------------------------------
ALERT_VERBOSE             pContactMgrV20  2.91    gilda  new_contact=henry
APPCAST                   uProcessWatch   7.68    gilda  formatted report
CONTACTS_LIST             pContactMgrV20  2.91    gilda  henry
CONTACTS_RECAP            pContactMgrV20  242.05  gilda  vname=henry,range=180.00,age=0.69
CONTACT_CLOSEST           pContactMgrV20  2.91    gilda  henry
CONTACT_CLOSEST_TIME      pContactMgrV20  2.91    gilda  3218502674.874138
PCONTACTMGRV20_ITER_GAP   pContactMgrV20  242.04  gilda  1.004112
PCONTACTMGRV20_ITER_LEN   pContactMgrV20  242.05  gilda  0.000828
PCONTACTMGRV20_PID        pContactMgrV20  -0.12   gilda  42130
PCONTACTMGRV20_STATUS     pContactMgrV20  241.04  gilda  AppErrorFlag=false,Uptime=241.812,cpuload=0.2981,memory_kb=3224,memory_max_kb=3224,
```

Figure 2: **Example RealmCast Report:** A realmcast message is expanded into a multi-line report, consisting of a report on subscribed variables and published variables for a given application. In this case the report is for the `pContactMgrV20` application on vehicle `gilda`.

The content of the report can be adjusted by the client, e.g., `uMACViewer` user, to help visualize the important information. There are seven options:

- Source: The Source column of the report may be suppressed.
- Community: The Community column of the report may be suppressed.
- UTC Time: The time format may be shown in absolute UTC time, or relative to the start of the local MOOSDB.
- Subscriptions: The subscriptions portion of the report may be suppressed.

- Mask: In the subscriptions portion, virgin variables may be suppressed.
- Wrap: Long string content may be wrapped over several lines, e.g., as in Figure 2.
- Truncate: Long string content may be truncated.

Realmcast content may be adjusted through the seven buttons on the lower part of uMACViewer. These buttons are only present when realmcasting is enabled.

The start-up value of these settings may also be set to the user's liking in the uMACViewer configuration block:

```
realmcast_show_source        = true/false   // Default is true
realmcast_show_community     = true/false   // Default is true
realmcast_show_subscriptions = true/false   // Default is true
realmcast_show_masked        = true/false   // Default is true
realmcast_wrap_content       = true/false   // Default is false
realmcast_trunc_content      = true/false   // Default is false
realmcast_time_format_utc    = true/false   // Default is false
```

2.3 Publications and Subscriptions

2.3.1 Variables Published by uMACViewer

- APPCAST: Contains an appcast report identical to the terminal output. Appcasts are posted only after an appcast request is received from an appcast viewing utility.
- APPCAST_REQ_<COMMUNITY>: As an appcast viewer, pMarineViewer also generates outgoing appcast requests to MOOS communities it is aware of, including its own MOOS community. These postings are typically bridged to the other named MOOS community with the variable renamed simply to APPCAST_REQ when it arrives in the other community.

2.3.2 Variables Subscribed for by uMACViewer

- APPCAST: As an appcast enabled *viewer*, pMarineViewer also subscribes for appcasts from other other applications and communities to provide the content for its own viewing capability.
- REALMCAST: As an realmcast enabled *viewer*, pMarineViewer also subscribes for realmcasts from MOOS communities running pRealm, and renders them in the infocast panes described in Section **??**.
- WATCHCAST: As an realmcast enabled *viewer*, uMACViewer also subscribes for watchcasts from MOOS communities running pRealm, and renders them in the infocast panes.

2.4 Configuration File Parameters

A few configuration parameters are exposed to facilitate visual preference settings that would otherwise need to be done each time the application is launched via pull-down menus. These parameters are listed below, but may be recalled anytime by typing on the command line: "uMACView -e".

Note: uMACView was released in 2012. Starting with the first release after 2019's Release 19.8, this app was augmented to include realmcasting in addition to appcasting. The term *infocasting* is the general term referring to either appcasting or realmcasting. As such, some of the earlier configuration parameters, e.g., appcast_font_size, have been replace with a term like infocast_font_size. The older configuration parameters are deprecated but still supported foreseeable releases.

Listing 2.1: Configuration Parameters for uMACView.

appcast_color_scheme:	Possible settings, `default`, `beige`, `indigo` or `white`. The default is `indigo`.
content_mode:	Sets the on startup content, which can be changed by the user at any time. Either `appcast` or `realmcast`. The default is `appcast`.
infocast_font_size:	Possible settings, `xsmall`, `small`, `medium`,`large`,`xlarge`. The default is `medium`.
infocast_height:	Possible settings, [`30,35,40,..., 85,90`]. The default is `70`.
nodes_font_size:	The font size used in the *nodes* pane of the set of infocasting panes. Possible settings, `xsmall`, `small`, `medium`, `large`, or `xlarge`. The default is `large`.
procs_font_size:	Possible settings, `xsmall`, `small`, `medium`, `large` or `xlarge`. The default is `large`.
realmcast_color_scheme:	Either `indigo`, `beige`, `hillside`, `white`, or `default`. The default is `hillside`.
realmcast_show_source:	If `true`, the Source column is shown on realmcast output. Setting to `false` conserves screen space, possibly enhancing readability. The default is `true`.
realmcast_show_community:	If `true`, the Community column is shown on realmcast output. Setting to `false` conserves screen space, possibly enhancing readability. The default is `true`.
realmcast_show_subscriptions:	If `true`, the Subscriptions block is shown on realmcast output. Setting to `false` conserves screen space, possibly enhancing readability. The default is `true`.
realmcast_show_masked:	If `true`, certain variables are excluded in realmcast output, e.g., virgin variables. Setting to `false` conserves screen space, possibly enhancing readability. The default is `true`.
realmcast_wrap_content:	If `true`, the variable Value column in realmcast output will wrap onto several lines. Setting to `true` possibly enhances readability for very long output. The default is `false`.
realmcast_trunc_content:	If `true`, the variable Value column in realmcast output will be truncated. Setting to `true` possibly enhances readability for very long output. The default is `false`.
realmcast_time_format_utc:	If `true` the Time column in realmcast output will be shown in UTC time instead of local time since app startup. The default is `false`.

refresh_mode: Possible settings, paused, events, streaming. The default is events.

The infocast_height refers to the relative height of the infocast pane to the window. The default of 70 means the pane will be 70% of the overall window height.

The refresh_mode refers to the policy of refreshing appcasts via appcast requests sent to the nodes and their applications. This is discussed below in Section 2.6

2.5 Command Line Arguments and Options

A few command line arguments are available. They are similar to most other MOOS-IvP applications. These arguments are listed below, but may be recalled anytime by typing on the command line:

```
$ MACView --help or -h
```

- --alias=<ProcessName>: Launch with the given process name rather than uMACView.
- --example, -e: Display example MOOS configuration block.
- --help, -h: Display command line usage.
- --interface, -i: Display MOOS publications and subscriptions.
- --version, -v: Display release version information.

2.6 Refresh Modes

The uMACView utility, operates in one of three *refresh modes*. This mode is always shown on the upper right in the title bar in reverse color. The three modes are the *streaming*, *events*, and *paused* modes.

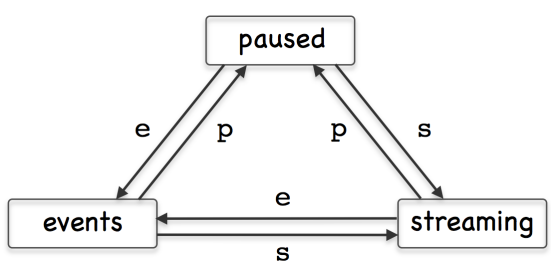

Figure 3: The uMAC utility is in one of three *refresh modes*, determining the manner in which appcast requests are conveyed to known applications and nodes. Reserved keyboard keys are used to transition between modes.

The Paused Refresh Mode In the *paused* refresh mode, no appcasts are solicited by the uMAC utility. The information being rendered should remain constant, virtually paused. Since appcast requests have a duration associated with them, prior appcast requests received by an application may need some time before expiring. Therefore the uMAC user may still see a trickle of updates after entering the paused mode. Furthermore, if there is another uMAC utility open, generating appcast requests, updates might still be seen after pausing.

The Events Refresh Mode In the *events* refresh mode, appcast requests of two types are sent to all known nodes and applications. The first type of request is sent only to the selected node and application. This type requests a continual update of appcasts, unconditionally. This application is, after all, the selected application for viewing. The second type of request is sent to all other nodes and applications requesting an appcast *only if a new run warning is generated*. The *events* refresh mode is the default mode upon launch.

The Streaming Refresh Mode In the *streaming* refresh mode, appcast requests are sent to all nodes and all applications, all the time. Furthermore, the request type is unconditional, meaning the application is requested to post a new appcast regardless of whether anything has changed in the application status. This mode is normally used for brief debugging, perhaps to check whether a node or application fails to respond. It creates a lot of appcast messaging traffic. It is not recommended to operate in this mode normally.

3 The uMAC Utility

The `uMAC` utility is an appcast viewer implemented in the terminal console. It acts the same as `uMACView` in terms of soliciting appcasts by publishing `APPCAST_REQ` messages and receiving `APPCAST` mail. The advantage over the GUI tool is the ability to remotely log into a vehicle and launch `uMAC` in a terminal window. This is espcially useful if the vehicle is otherwise not behaving in its communication with a shoreside MOOS community. Like `uMACView`, multiple versions of the utility may be running at the same time without interference with one another.

3.1 Content Modes

The viewable information in the `uMAC` tool is rendered in one of four *content modes*. Besides a *help* mode, the other three modes correlate to one of the three panes of the `uMACView` window in Figure 1. The primary content mode is the *appcast content mode*, where the output is an appcast of a particular application as shown in Figure 4.

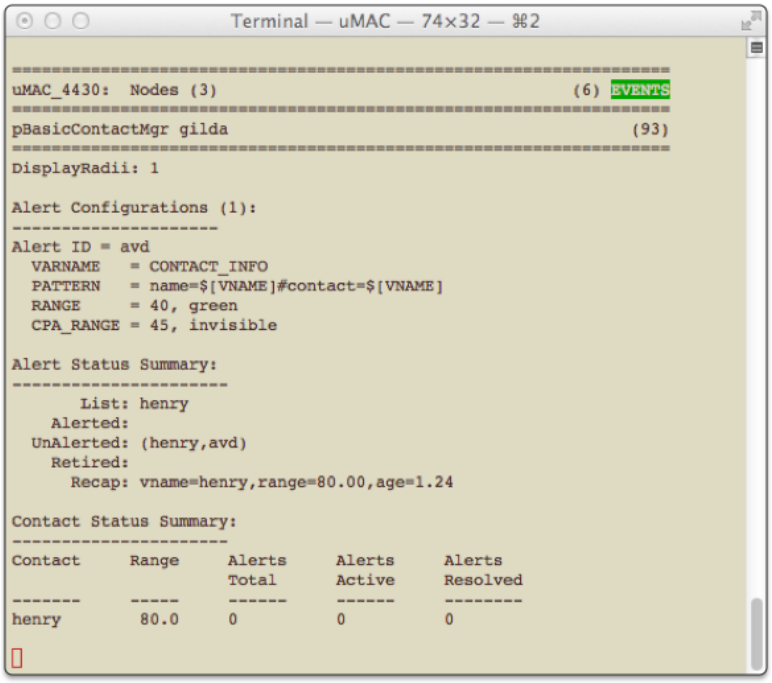

Figure 4: The `uMAC` utility monitors appcasts from a terminal window. The user may switch between appcasting sources by navigating with a keyboard menu. The primary advantage or `uMAC` is the ability to run it on a remotely deployed vehicle with a network connection.

The content of this window should look very similar to the bottom pane of the `uMACView` window in Figure 1. Two other content modes are supported, the *nodes content mode*, and the *procs content mode*. The former allows the user to select between different nodes, i.e., vehicles. The latter allows selection between different processes (MOOS applications) for the selected node. These modes correlate to the top two panes in the `uMACView` tool shown in Figure 1. A fourth, help, content mode is also supported. Transitioning between modes usually is done by selecting one of the choices presented, or popping back up a mode to allow a higher level choice. This done with three reserved keyboard keys, `p, n, h`, implementing the transitions shown in Figure 5.

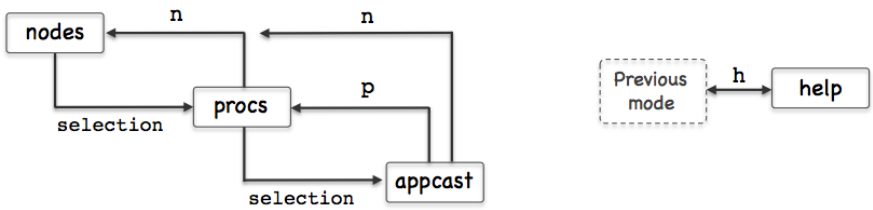

Figure 5: The `uMAC` utility monitors appcasts from a terminal window. The

An example rendering of `uMAC` in the *nodes content mode* is shown on the left in Figure 6. Each discovered node is assigned a character ID in the left-most column for selecting the node. The ID's are assigned as the nodes are discovered from incoming `APPCAST` messages.

The title line at the top of the report shows, on the left, the name of the uMAC application as it is known to the MOOSDB, and the number of nodes discovered. When uMAC is launched it gives itself a random suffix, such as uMAC_5991 in this example, to allow multiple uMAC sessions connect with the same MOOSDB. Recall the MOOSDB requires unique names across applications. The righthand side of the title line shows the number of iterations of the uMAC in parentheses, and the *refresh mode* shown in reverse color.

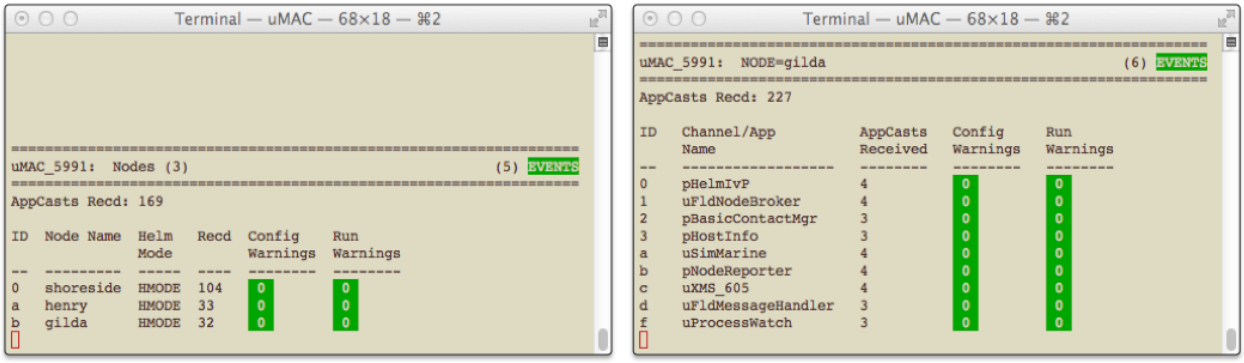

Figure 6: The uMAC utility monitors appcasts from a terminal window. The user may switch between appcasting sources by navigating with a keyboard menu. The primary advantage or uMAC is the ability to run it on a remotely deployed vehicle with a network connection.

An example from the *procs content mode* is shown on the right in Figure 6. Each discovered process (MOOS application) is assigned an ID in the left-most column for selecting the process. The ID's are assigned as the apps are discovered from incoming APPCAST messages. The title line at the top is nearly the same format as in the *nodes* content mode, except that the selected node is shown rather than the total nodes. The appcast counter just below the title line indicates the total number of appcasts received over all nodes, not just the selected node.

3.2 Refresh Modes

The uMAC utility, like the uMacView utility, operates in one of three *refresh modes*. This mode is always shown on the upper right in the title bar in reverse color. The three modes are the *streaming*, *events*, and *paused* modes. The desciption of these modes, given in Section 2.6, is also applicable to the operation for the uMAC utility.

3.3 A Tip Regarding Process Monitoring and uMAC Sessions

The uProcessWatch application is a utility for monitoring the presence of applications connected to the MOOSDB. It is appcast enabled, and will post a run warning when it detects the disappearance of a prior noted process. If a uMAC is launched and then exited, the uProcessWatch utility may interpret this as a problem and post a run warning. The effect may be that real run warnings are then later ignored. Tip: Add the following configuration line to uProcessWatch to ignore the exit of a uMAC session: `nowatcth = uMAC*`.

3.4 Publications and Subscriptions

The sole subscription for uMACView is the appcast message in the MOOS variable APPCAST. The sole publication is the appcast request, in the variable APPCAST_REQ.

3.5 Configuration File Parameters

There are no configuration file parameters specific to uMAC.

3.5.1 Command Line Arguments and Options

A few command line arguments are available. They are similar to most other MOOS-IvP applications. These arguments are listed below, but may be recalled anytime by typing on the command line:

```
$ uMAC --help or -h
```

- `--alias=<ProcessName>`: Launch with the given process name rather than uMACView.
- `--example, -e`: Display example MOOS configuration block.
- `--help, -h`: Display command line usage.
- `--interface, -i`: Display MOOS publications and subscriptions.
- `--version, -v`: Display release version information.

4 The uMACView Utility Integrated with pMarineViewer

The final uMAC tool is essentially uMACView embedded in the pMarineViewer application as shown in Figure 7. This is mostly just a convenience for users already using pMarineViewer. The appcasting mode may be toggled with the 'a' key to return to the traditional viewing layout. The AppCasting pull-down menu offers the same set of selections as uMACView with the exception of the hot keys used to switch between appcasting refresh modes since those keys were already used for other things in pMarineViewer.

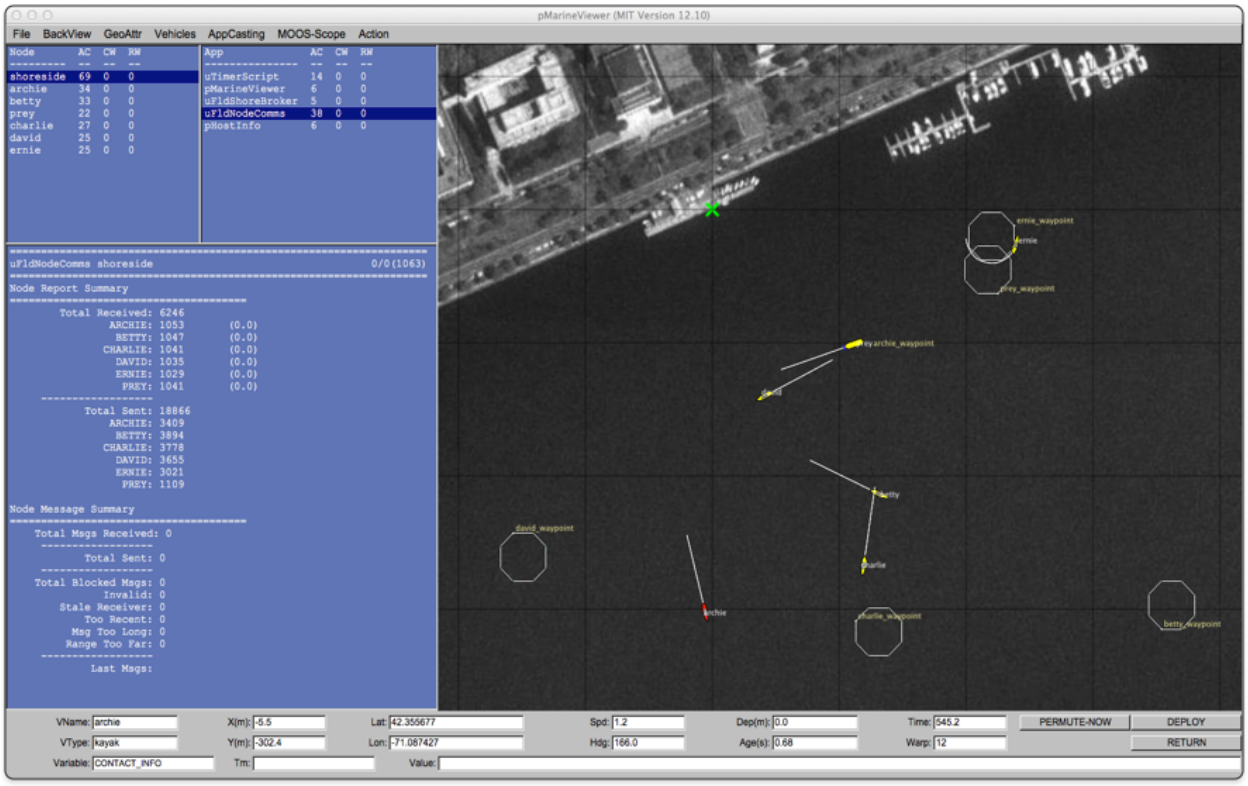

Figure 7: The pMarineViewer utility has an appcasting viewing capability very similar to uMACView embedded in the viewer. The rendering of the appcasting panes may be toggled on/off with the 'a' key.

In addition to toggling on/off the appcasting portion of the window, the width of the set of appcasting panes may be made wider or thinner using the CTRL-ALT-ARROW keys. By default the appcasting panes consume 30% of the width. The default height of the appcasting (bottom) portion of the appcasting panes consume 75% of the height of the set of appcasting panes. These startup extents may be changed with configuration the parameters:

```
appcasting_width  = 25    // legal values [20, 25,... ,65, 70]
appcasting_height = 80    // legal values [30, 35,..., 85, 90]
```

The prevailing value of these parameters can always be discovered by checking which radio button in the pull-down menu is presently selected.

References

[1] Michael R. Benjamin. pRealm: Integrated Scoping of the MOOSDB. http://oceanai.mit.edu/ivpman/apps/pRealm.