Informatica™

Ultra Messaging (Version 6.17)

Java Examples

Contents

Chapter 1

Introduction

This document lists and gives some background information on the Java-language example UM programs.

For policies and procedures related to Ultra Messaging Technical Support, see UM Support.

See **UM Glossary** for Ultra Messaging terminology, abbreviations, and acronyms.

1.1 Java Examples Introduction

These programs were written to help in troubleshooting, testing, and demonstrating UM coding techniques. See also C Example Source Code and C# Example Source Code.

Since the tools are written to be useful as well as instructive, they are more complex than purely-instructive examples would be, with many options to add or subtract functionality. See UMExamples for purely-instructive examples of a variety of UM use cases.

The example Java programs listed here are provided in both source form and in binary executable form (in the main UMS jar file).

1.2 Configuring Java Examples

The example programs universally provide the "-c filename" command-line option. Using that option, the example application calls the **LBM.setConfiguration(String fileName)** API. However, note that this API is not recommended for use with XML-format LBM configuration files, largely because you are not able to specify an application name.

To use an XML configuration file with a UM example application, set the environment variables:

- LBM_XML_CONFIG_APPNAME - Desired name of application.
- LBM_XML_CONFIG_FILENAME - Path name of XML configuration file.

In this way, UM will correctly set the example application's name and will properly load the XML configuration file.

1.3 Running Java Examples

The minimal classpath depends on the features being used. Informatica recommends the following jar files be used with running the examples:

- java-getopt-*.jar : `java-getopt`, the GNU Java port of glibc getopt and getlong for parsing command-line options.
- UMS_6.17.jar : main jar file containing UM APIs and also compiled class files for example applications.
- UMSSDM_6.17.jar : Self Describing Messages API (rarely used by customers, but used by some example applications).
- UMSMON_PROTO2_6.17.jar : Monitoring protocol buffer (v2) definitions (rarely used by customers).
- UMSMON_PROTO3_6.17.jar : Monitoring protocol buffer (v3) definitions (rarely used by customers).

Note that the java-getopt jar file can be found in either the MCS/lib or the SRS/lib directories.

For example:

```
L=$HOME/UMQ_6.17/
J=$L/java

export LBM_LICENSE_INFO="Product=LBM,UME,UMQ,UMDRO:Organization=my
    organization:Expiration-Date=xxxx:License-Key=xxxx xxxx xxxx xxxx"
export LD_LIBRARY_PATH=$L/Linux-glibc-2.17-x86_64

java -cp $L/MCS/lib/java-getopt-1.0.13.jar:$J/UMS_6.17.jar:$J/UMSSDM_6.17.jar:$J/
    UMSMON_PROTO2_6.17.jar:$J/UMSMON_PROTO3_6.17.jar lbmsrc -c my_config.cfg
    my_topic
```

1.4 Unhandled Java Events

Each of the example programs is written to demonstrate a subset of UM's total available functionality. For example, some programs are written to demonstrate **Streaming** functionality (e.g. lbmsrc), while other programs are written to demonstrate **Persistence** functionality (e.g. umesrc), while still other programs are written to demonstrate **Queuing** functionality (e.g. umqsrc).

UM is generally designed to be event-driven, with events being delivered to the programs through standard callbacks, like source callbacks and receiver callbacks. There are many events which are common across all streaming, persistence, and queuing. Other events are specific to persistence, and still other events are specific to queuing.

This can lead to example programs reporting "unknown" or "unhandled" events. For example, if the "lbmsrc" streaming program is run with a configuration file that enables persistence, UM will deliver events that are specific to persistence to the "lbmsrc" program. But "lbmsrc" is designed for streaming, and does not include code cases for persistence or queuing events. Maybe you should change your configuration to disable persistence, or you should be using the "umesrc" example program.

Similarly, the "umqsrc" program expects queuing functionality, and can report unhandled events if persistence is configured. Or "umesrc" can report unhandled events if queuing is configured.

If you see an unhandled event, it is generally reported as a number. You can see which event this corresponds to by looking up the number in the C API document:

- **C Receiver Events** for subscribing programs and
- **C Source Events** for publishing programs.

Note that Java uses the same numbering system.

Once you understand the nature of the unhandled event, you can decide how to change your configuration or choose a different program.

1.5 Java Examples

1.5.1 Example displayString.java

Source code: `displayString.java`:

```
Usage: This is not a directly-usable java file.
 It contains the 'displayString' methods from the LBM library
 statistics classes, and is provided for reference purposes.
```

1.5.2 Example lbmdump.java

Source code: `lbmdump.java`:

```
Utility functions for UM example programs.
```

1.5.3 Example lbmExampleUtil.java

Source code: `lbmExampleUtil.java`:

```
Utility functions for UM example programs.
```

1.5.4 Example lbmhfxrcv.java

Source code: lbmhfxrcv.java:

```
Purpose: Receive messages on a single topic.

Usage: lbmhfxrcv [options] topic
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -D = Assume received messages are SDM formatted
  -E = exit after source ends
  -e = use LBM embedded mode
  -h = help
  -r msgs = delete receiver after msgs messages
  -S = exit after source ends, print throughput summary
  -s num_secs = print statistics every num_secs along with bandwidth
  -q = use an LBM event queue
  -v = be verbose about each message
  -V = verify message contents
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-rcv NUM = monitor receiver every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be `lbm', `udp', or `lbmsnmp', default is
                                  `lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be `csv' or `pb', default is `csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.5 Example lbmimsg.java

Source code: lbmimsg.java:

```
Purpose: Send immediate messages on a single topic or send topic-less messages.

Usage: lbmimsg [options] topic
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -d delay = delay sending for delay seconds after source creation
  -h = help
  -l len = send messages of len bytes
  -L linger = linger for linger seconds before closing context
  -M msgs = send msgs number of messages
  -o = send immediate topic-less messages
  -R [UM]DATA/RETR = Set transport type to LBT-R[UM], set data rate limit to
                     DATA bits per second, and set retransmit rate limit to
                     RETR bits per second.  For both limits, the optional
                     k, m, and g suffixes may be used.  For example,
                     '-R 1m/500k' is the same as '-R 1000000/500000'
  -T target = target for unicast immediate messages
```

1.5.6 Example lbmlatping.java

Source code: lbmlatping.java:

```
Usage: lbmlatping [options]
Available options:
  -a procnum = set cpu affinity to procnum (disabled).
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -h = help
  -l len = use len length messages
  -P usec = pause after each send usec microseconds (busy wait only)
```

1.5.7 Example lbmlatpong.java

Source code: lbmlatpong.java:

```
Usage: lbmlatpong [options]
Available options:
  -a procnum = set cpu affinity to procnum (disabled).
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -h = help
```

1.5.8 Example lbmmon.java

Source code: lbmmon.java:

```
Purpose: Example LBM statistics monitoring application.

Usage: lbmmon [options]
Available options:
  -h, --help                 help
  -t, --transport TRANS      use transport module TRANS
                             TRANS may be `lbm', `udp', or `lbmsnmp', default is
                                 `lbm'
      --transport-opts OPTS  use OPTS as transport module options
  -f, --format FMT           use format module FMT
                             FMT may be `csv' or `pb', default is `csv'
      --format-opts OPTS     use OPTS as format module options

Transport and format options are passed as name=value pairs, separated by a
    semicolon.

LBM transport options:
  config=FILE            use LBM configuration file FILE
  topic=TOPIC            receive statistics on topic TOPIC
                         default is /29west/statistics
  wctopic=PATTERN        receive statistics on wildcard topic PATTERN
```

```
UDP transport options:
  port=NUM                 receive on UDP port NUM
  interface=IP             receive multicast on interface IP
  mcgroup=GRP              receive on multicast group GRP

LBMSNMP transport options:
  config=FILE              use LBM configuration file FILE
  topic=TOPIC              receive statistics on topic TOPIC
                           default is /29west/statistics
  wctopic=PATTERN          receive statistics on wildcard topic PATTERN

CSV format options:
  separator=CHAR           separate CSV fields with character CHAR
                           defaults to ','
  passthrough=VAL          VAL may be 'off', 'on' or 'convert'
                           defaults to 'off'
PB format options:
  passthrough=VAL          VAL may be 'off', 'on' or 'convert'
                           defaults to 'off'
```

1.5.9 Example lbmmrcv.java

Source code: lbmmrcv.java:

```
Purpose: Receive messages on multiple topics.

Usage: lbmrcv [options]
Available options:
  -B # = Set receive socket buffer size to # (in MB)
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -C ctxs = use ctxs number of context objects
  -E = Exit on receiving EOS
  -e = use LBM embedded mode
  -h = help
  -i num = initial topic number num
  -o offset = use offset to calculate Registration ID
              (as source registration ID + offset)
             offset of 0 forces creation of regid by store
  -r root = use topic names with root of \
  -R rcvs = create rcvs receivers
  -s = print statistics along with bandwidth
  -v = be verbose about each message
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-rcv NUM = monitor all receivers every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be 'lbm', 'udp', or 'lbmsnmp', default is
                                'lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be 'csv' or 'pb', default is 'csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.10 Example lbmmsrc.java

Source code: lbmmsrc.java:

```
Purpose: Send messages on multiple topics.

Usage: lbmmsrc [options]
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -d delay = delay sending for delay seconds after source creation
  -e = use LBM embedded mode
  -h = help
  -i num = initial topic number
  -l len = send messages of len bytes
  -L linger = linger for linger seconds before closing context
  -M msgs = send maximum of msgs number of messages
  -r root = use topic names with root of \
  -s = print source statistics before exiting
  -P msec = pause msec milliseconds after each send
  -R [UM]DATA/RETR = Set transport type to LBT-R[UM], set data rate limit to
                     DATA bits per second, and set retransmit rate limit to
                     RETR bits per second.  For both limits, the optional
                     k, m, and g suffixes may be used.  For example,
                     '-R 1m/500k' is the same as '-R 1000000/500000'
  -S srcs = use srcs sources
  -T thrds = use thrds threads
  -v = be verbose
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-src NUM = monitor each source every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be `lbm', `udp', or `lbmsnmp', default is
                                  `lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be `csv' or `pb', default is `csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.11 Example lbmpong1.java

Source code: lbmpong1.java:

```
Purpose: One-way message trip processor.

Usage: lbmpong1 [options] id
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -E = exit after source ends
  -e = use LBM embedded mode
  -h = help
  -i ID = producer/consumer IDentifier
  -M msgs = stop after receiving msgs messages
  -q = use an LBM event queue
  -t secs = run for secs seconds
  -v = be verbose about each message (for RTT only)
```

1.5.12 Example lbmpong.java

Source code: lbmpong.java:

```
Purpose: Message round trip processor.

Usage: lbmpong [options] id
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -C = collect RTT data
  -d = delay start (wait milliseconds to complete registration)
  -E = exit after source ends
  -e = use LBM embedded mode
  -h = help
  -i msgs = send and ignore msgs messages to warm up
  -I = Use MIM
  -j = join live stream (do not recover from stores)
  -l len = use len length messages
  -M msgs = stop after receiving msgs messages
  -o offset = use offset to calculate Registration ID
              (as source registration ID + offset)
              offset of 0 forces creation of regid by store
  -P msec = pause after each send msec milliseconds
  -q = use an LBM event queue
  -r [UM]DATA/RETR = Set transport type to LBT-R[UM], set data rate limit to
                     DATA bits per second, and set retransmit rate limit to
                     RETR bits per second.  For both limits, the optional
                     k, m, and g suffixes may be used.  For example,
                     '-R 1m/500k' is the same as '-R 1000000/500000'
  -t secs = run for secs seconds
  -T topic = topic name prefix (appended with '/' and id) [lbmpong]
  -v = be verbose about each message (for RTT only)
  id = either \
```

1.5.13 Example lbmrcv.java

Source code: lbmrcv.java:

```
Purpose: Receive messages on a single topic.

Usage: lbmrcv [options] topic
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -D = Assume received messages are SDM formatted
  -E = exit after source ends
  -e = use LBM embedded mode
  -f = use hot-failover
  -h = help
  -r msgs = delete receiver after msgs messages
  -N NUM = subscribe to channel NUM
  -S = exit after source ends, print throughput summary
  -s num_secs = print statistics every num_secs along with bandwidth
  -q = use an LBM event queue
  -v = be verbose about each message
  -V = verify message contents
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-rcv NUM = monitor receiver every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be `lbm', `udp', or `lbmsnmp', default is
                                  `lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be `csv' or `pb', default is `csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.14 Example lbmrcvxsp.java

Source code: `lbmrcvxsp.java`:

```
Purpose: Receive messages on a single topic, mapping transports to various XSPs.

Usage: lbmrcv [options] topic
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -d = don't delete XSPs until shutdown
  -D = use the default XSP for all transports
  -E = exit after source ends
  -h = help
  -P = preallocate the XSPs - use with -R
  -Q = use sequential mode for XSPS
  -r msgs = delete receiver after msgs messages
  -R NUM = use a simple round-robin method for assigning transports to NUM XSPs.
           (this is the DEFAULT for this application, with a NUM of 3
  -s num_secs = print statistics every num_secs along with bandwidth
  -S = exit after source ends, print throughput summary
  -v = be verbose about each message
  -V = verify message contents
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-rcv NUM = monitor receiver every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be `lbm', `udp', or `lbmsnmp', default is
                                  `lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be `csv' or `pb', default is `csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.15 Example lbmreq.java

Source code: `lbmreq.java`:

```
Purpose: Send request messages from a single source with settable interval between
    messages.

Usage: [options] topic
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -d delay = delay sending for delay seconds after source creation
  -e = use LBM embedded mode
  -h = help
  -i = send immediate requests
  -l len = send messages of len bytes
  -L linger = linger for linger seconds before closing context
  -P sec = pause sec seconds after sending request (for responses to arrive)
  -q use event queue
  -r [UM]DATA/RETR = Set transport type to LBT-R[UM], set data rate limit to
                     DATA bits per second, and set retransmit rate limit to
                     RETR bits per second.  For both limits, the optional
                     k, m, and g suffixes may be used.  For example,
                     '-r 1m/500k' is the same as '-r 1000000/500000'
  -R requests = send request number of requests
  -T target = target for unicast immediate requests
  -v = be verbose
  -v -v = be even more verbose
```

1.5.16 Example lbmresp.java

Source code: lbmresp.java:

```
Purpose: Respond to request messages on a single topic.

Usage: lbmresp [options] topic
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -E = end after end-of-stream
  -e = use LBM embedded mode
  -f topic = forward request to responders listening on given topic
  -h = help
  -l len = use len bytes for the length of each response
  -q = use an LBM event queue
  -r responses = send responses messages for each request
  -v = be verbose about each message
  -v -v = be even more verbose about each message
```

1.5.17 Example lbmsrc.java

Source code: lbmsrc.java:

```
Purpose: Send messages on a single topic.

Usage: lbmsrc [options] topic
Available options:
  -c filename = Use LBM configuration file filename.
```

```
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -C filename = read context config parameters from filename
  -d delay = delay sending for delay seconds after source creation
  -D = Use SDM Messages
  -e = use LBM embedded mode
  -f = use hot-failover
  -i init =  hot-failover: start with this initial sequence number
  -h = help
  -l len = send messages of len bytes
  -L linger = linger for linger seconds before closing context
  -M msgs = send msgs number of messages
  -N chn = send messages on channel chn
  -n = use non-blocking I/O
  -P msec = pause after each send msec milliseconds
  -R [UM]DATA/RETR = Set transport type to LBT-R[UM], set data rate limit to
                     DATA bits per second, and set retransmit rate limit to
                     RETR bits per second.  For both limits, the optional
                     k, m, and g suffixes may be used.  For example,
                     '-R 1m/500k' is the same as '-R 1000000/500000'
  -t filename = use filename contents as a recording of message sequence numbers
      (HF only!)
  -s sec = print stats every sec seconds
  -v = be verbose about each message
  -V = construct verifiable messages
  -x bits = Use 32 or 64 bits for hot-failover sequence numbers
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-src NUM = monitor source every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be `lbm', `udp', or `lbmsnmp', default is
                                  `lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be `csv' or `pb', default is `csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.18 Example lbmssrc.java

Source code: `lbmssrc.java`:

```
Purpose: Uses Smart Source to send messages on a single topic.

Usage: lbmssrc [options] topic
Available options:
  -a available-data-space = print the length of available data space
  -b user-supplied-buffer =  send messages using a user-supplied buffer
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -C filename = read context config parameters from filename
  -d delay = delay sending for delay seconds after source creation
  -e = use LBM embedded mode
  -h = help
  -i msg_prop = send message property as integer value with string key
                Example:  '-i 1,prop1'
  -l len = send messages of len bytes
  -L linger = linger for linger seconds before closing context
  -M msgs = send msgs number of messages
  -N chn = send messages on channel chn
  -P msec = pause after each send msec milliseconds
  -s sec = print stats every sec seconds
  -v = be verbose about each message
  -V = construct verifiable messages
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-src NUM = monitor source every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be 'lbm', 'udp', or 'lbmsnmp', default is
                                  'lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be 'csv' or 'pb', default is 'csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.19 Example lbmstrm.java

Source code: lbmstrm.java:

```
Purpose: Send messages on multiple topics.

Usage: lbmstrm [options]
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -e = use LBM embedded mode
  -h = help
  -i num = initial topic number
  -l len = send messages of len bytes
  -L linger = linger for linger seconds before closing context
  -m NUM = send at NUM messages per second
  -M msgs = send maximum of msgs number of messages
  -r root = use topic names with root of \
  -s = print source statistics before exiting
  -R [UM]DATA/RETR = Set transport type to LBT-R[UM], set data rate limit to
                     DATA bits per second, and set retransmit rate limit to
                     RETR bits per second.  For both limits, the optional
                     k, m, and g suffixes may be used.  For example,
                     '-R 1m/500k' is the same as '-R 1000000/500000'
  -S srcs = use srcs sources
  -t = tight loop (cpu-bound) for even message spacing
  -T thrds = use thrds threads
  -v = be verbose
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-src NUM = monitor each source every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be 'lbm', 'udp', or 'lbmsnmp', default is
                                  'lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be 'csv' or 'pb', default is 'csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.20 Example lbmtrreq.java

Source code: lbmtrreq.java:

```
Purpose: Request topic resolution for quiescent components.

Usage: lbmtrreq [options]
Available options:
  -c filename =      Use LBM configuration file filename.
                     Multiple config files are allowed.
                     Example:  '-c file1.cfg -c file2.cfg'
  -C filename =      read Java Properties file for context config
  -a, --adverts      Request Advertisements
  -q, --queries      Request Queries
  -w, --wildcard     Request Wildcard Queries
  -i, --interval=NUM Interval between requests (milliseconds)
  -d, --duration=NUM Minimum duration of requests (seconds)
  -L, --linger=NUM   Linger for NUM seconds before closing context
```

1.5.21 Example lbmwrcv.java

Source code: lbmwrcv.java:

```
Purpose: Receive messages using a wildcard receiver.

Usage: lbmwrcv [options] topic
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -E = exit after source ends
  -e = use LBM embedded mode
  -h = help
  -q = use an LBM event queue
  -r msgs = delete receiver after msgs messages
  -s = print statistics along with bandwidth
  -N NUM = subscribe to channel NUM
  -v = be verbose about each message
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be `lbm', `udp', or `lbmsnmp', default is
                                  `lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be `csv' or `pb', default is `csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.22 Example MinRcv.java

Source code: MinRcv.java:

```
Purpose: minimal subscriber application.

MinRcv.java - Minimal receiver program.
```

1.5.23 Example MinSrc.java

Source code: MinSrc.java:

```
Purpose: minimal publisher application.

MinSrc.java - Minimal source program.
```

1.5.24 Example umercv.java

Source code: umercv.java:

```
Purpose: Receive messages on a single topic.

Usage: umercv [options] topic
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -E = exit after source ends
  -D = Deregister after receiving 1000 messages
  -e = use LBM embedded mode
  -X num_msgs = send an eXplicit ACK every num_msgs messages
  -i offset = use offset to calculate Registration ID
              (as source registration ID + offset)
              offset of 0 forces creation of regid by store
  -N NUM = subscribe to channel NUM
  -Q seqnum_offset = display recovery sequence number info and set low seqnum to
      low+seqnum_offset
  -S = exit after source ends, print throughput summary
  -s num_secs = print statistics every num_secs along with bandwidth
  -h = help
  -q = use an LBM event queue
  -r msgs = delete receiver after msgs messages
  -v = be verbose about each message
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-rcv NUM = monitor receiver every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be 'lbm', 'udp', or 'lbmsnmp', default is
                                  'lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be 'csv' or 'pb', default is 'csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.25 Example umesrc.java

Source code: umesrc.java:

```
Purpose: Send messages on a single topic.

Usage: umesrc [options] topic
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -C filename = read context config parameters from filename
  -D = Send deregistration after sending 1000 messages
  -e = use LBM embedded mode
  -f NUM = allow NUM unstabilized messages in flight (determines message rate)
  --flight-size = See -f above
  -h = help
  -j = turn on UME late join
  -l len = send messages of len bytes
  -L linger = linger for linger seconds before closing context
  -m NUM = send at NUM messages per second (trumped by -f)
  --message-rate = See -m above
  -M msgs = send msgs number of messages
  -N = display sequence number information source events
  -n = use non-blocking I/O
  -P msec = pause after each send msec milliseconds
  -R [UM]DATA/RETR = Set transport type to LBT-R[UM], set data rate limit to
                     DATA bits per second, and set retransmit rate limit to
                     RETR bits per second.  For both limits, the optional
                     k, m, and g suffixes may be used.  For example,
                     '-R 1m/500k' is the same as '-R 1000000/500000'
  -S ip:port = use UME store at the specified address and port
  -s sec = print stats every sec seconds
  -t storename = use UME store with name storename
  -v = bump verbose level
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-src NUM = monitor source every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be `lbm', `udp', or `lbmsnmp', default is
                                  `lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be `csv' or `pb', default is `csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.26 Example umessrc.java

Source code: umessrc.java:

```
Purpose: Uses smart source to send messages on a single topic.

Usage: umessrc [options] topic
Available options:
  -a available-data-space = print the length of available data space
  -b user-supplied-buffer =  send messages using a user-supplied buffer
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -C filename = read context config parameters from filename
  -D = Send deregistration after sending 1000 messages
  -e = use LBM embedded mode
  -f NUM = allow NUM unstabilized messages in flight (determines message rate)
  --flight-size = See -f above
  -h = help
  -i msg_prop = send message property as integer value with string key
                Example:  '-i 1,prop1'
  -j = turn on UME late join
  -l len = send messages of len bytes
  -L linger = linger for linger seconds before closing context
  -m NUM = send at NUM messages per second (trumped by -f)
  --message-rate = See -m above
  -M msgs = send msgs number of messages
  -N chn = send messages on channel chn
  -P msec = pause after each send msec milliseconds
  -Q = display sequence number information source events
  -S ip:port = use UME store at the specified address and port
  -s sec = print stats every sec seconds
  -t storename = use UME store with name storename
  -v = bump verbose level
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-src NUM = monitor source every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be `lbm', `udp', or `lbmsnmp', default is
                                  `lbm'
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be `csv' or `pb', default is `csv'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.27 Example umqrcv.java

Source code: umqrcv.java:

```
Purpose: Receive messages on a single topic.

Usage: umqrcv [options] topic
Available options:
  -B broker = use broker given by address.
  -c filename = read config file filename
  -D = deregister upon exit
  -E = exit after source ends
  -e = use LBM embedded mode
  -X num_msgs = send an eXplicit ACK every num_msgs messages
  -i offset = use offset to calculate Registration ID
              (as source registration ID + offset)
              offset of 0 forces creation of regid by store
  -I ID = set Receiver Type ID to ID
  -N seqnum_offset = display recovery sequence number info and set low seqnum to
      low+seqnum_offset
  -S = exit after source ends, print throughput summary
  -s num_secs = print statistics every num_secs along with bandwidth
  -h = help
  -q = use an LBM event queue
  -r msgs = delete receiver after msgs messages
  -v = be verbose about each message
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-rcv NUM = monitor receiver every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
```

```
                              TRANS may be ‘lbm’, ‘udp’, or ‘lbmsnmp’, default is
                                  ‘lbm’
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be ‘csv’ or ‘pb’, default is ‘csv’
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.5.28 Example umqsrc.java

Source code: umqsrc.java:

```
Purpose: Send messages on a single topic.

Usage: umqsrc [options] topic
Available options:
  -A cfg = use ULB application set specification cfg
  -B broker = use broker given by address.
  -c filename = read config parameters from filename
  -C filename = read context config parameters from filename
  -e = use LBM sequential mode
  -f NUM = allow NUM unstabilized messages in flight
  -h = help
  -i = display Message IDs for sent messages
  -l len = send messages of len bytes
  -L linger = linger for linger seconds before closing context
  -m NUM = send at NUM messages per second (trumped by -f)
  -M msgs = send msgs number of messages
  -N = display sequence number information source events
  -n = use non-blocking I/O
  -P msec = pause after each send msec milliseconds
  -R rate/pct = send with LBT-RM at rate and retranmission pct%
  -s sec = print stats every sec seconds
  -X = Send using numeric or named UMQ index for ULB sources X
     -Y = Send using named UMQ index for broker sources
  -v = bump verbose level
Monitoring options:\n
  --monitor-ctx NUM = monitor context every NUM seconds
  --monitor-src NUM = monitor source every NUM seconds
  --monitor-transport TRANS = use monitor transport module TRANS
                              TRANS may be ‘lbm’, ‘udp’, or ‘lbmsnmp’, default is
                                  ‘lbm’
  --monitor-transport-opts OPTS = use OPTS as transport module options
  --monitor-format FMT = use monitor format module FMT
                         FMT may be ‘csv’ or ‘pb’, default is ‘csv’
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
  --flight-size = See -f above
  --message-rate = See -m above
  --appsets = see -A above
```