Ultra Messaging (Version 6.15)

.NET Examples

Contents

Chapter 1

Introduction

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

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

The information in this documentation is subject to change without notice. If you find any problems in this documentation, please report them to us in writing at Informatica LLC 2100 Seaport Blvd. Redwood City, CA 94063.

Informatica products are warranted according to the terms and conditions of the agreements under which they are provided.
INFORMATICA LLC PROVIDES THE INFORMATION IN THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT.

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

1.1 .NET Examples Introduction

These programs were written to help in troubleshooting, testing, and demonstrating UM coding techniques. See also C Example Source Code and Java 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 C# programs listed here are provided in both source form and in binary executable form.

1.2 Configuring .NET 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 Unhandled C# 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 C# 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.

1.4 C# Examples

1.4.1 Example lbmExampleUtil.cs

Source code: lbmExampleUtil.cs:

```
General utility functions for UM example programs.
```

1.4.2 Example lbmhfxrcv.cs

Source code: lbmhfxrcv.cs:

```
Purpose: Receive messages via a HFX.

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 qdelay = monitor event queue delay above qdelay usecs
  -E = exit after source ends
  -h = help
  -I CIDR = create a receiver on the interface specified by CIDR
            Multiple interfaces are allowed.
            Example:  '-I 10.29.1.0/24 -I 10.29.2.0/24
  -q = use an LBM event queue
  -S = exit after source ends, print throughput summary
  -s num_secs = print statistics every num_secs along with bandwidth
  -r msgs = delete receiver after msgs messages
  -v = be verbose about each message
  -V = verify message contents
  -z qsize = monitor event queue size above qsize in length
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'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.4.3 Example lbmimsg.cs

Source code: lbmimsg.cs:

```
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 topic-less immediate messages
  -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 target = target for unicast immediate messages
```

1.4.4 Example lbmlatping.cs

Source code: lbmlatping.cs:

```
Usage: lbmlatping [options]
Available options:
  -a procmask = set cpu affinity mask.
                Available processors bitmask:

  -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)
```

1.4.5 Example lbmlatpong.cs

Source code: lbmlatpong.cs:

```
Usage: lbmlatpong [options]
Available options:
  -a procmask = set cpu affinity mask.
                Available processors bitmask:

  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -h = help
```

1.4.6 Example lbmmon.cs

Source code: lbmmon.cs:

```
Purpose: Example LBM statistics monitoring application.

Usage: lbmmon [options]
Available options:
  -h = 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'
      --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 ','
```

1.4.7 Example lbmmrcv.cs

Source code: lbmmrcv.cs:

```
Purpose: Receive messages on multiple topics.

Usage: lbmmrcv [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
  -h = help
  -i num = initial topic number num
  -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 each 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'
  --monitor-format-opts OPTS = use OPTS as format module options
```

```
  --monitor-appid ID = use ID as application ID string
```

1.4.8 Example lbmmsrc.cs

Source code: lbmmsrc.cs:

```
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
  -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
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'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.4.9 Example lbmpong.cs

Source code: lbmpong.cs:

```
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
  -E = exit after source ends
  -h = help
```

```
  -i msgs = send and ignore msgs messages to warm up
  -I = Use MIM
  -l len = use len length messages
  -M msgs = stop after receiving msgs messages
  -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
  -v = be verbose about each message (for RTT only)
  id = either \
```

1.4.10 Example lbmrcv.cs

Source code: lbmrcv.cs:

```
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 qdelay = monitor event queue delay above qdelay usecs
  -D = Assume received messages are SDM formatted
  -E = exit after source ends
  -f = use hot-failover
  -h = help
  -n nsrcs = stop topic resolution queries after nsrcs sources
  -q = use an LBM event queue
  -S = exit after source ends, print throughput summary
  -s num_secs = print statistics every num_secs along with bandwidth
  -r msgs = delete receiver after msgs messages
  -N NUM = subscribe to channel NUM
  -v = be verbose about each message
  -V = verify message contents
  -z qsize = monitor event queue size above qsize in length
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'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.4.11 Example lbmrcvxsp.cs

Source code: lbmrcvxsp.cs:

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

Usage: lbmrcvxsp [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 = exit after source ends, print throughput summary
  -s num_secs = print statistics every num_secs along with bandwidth
  -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'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.4.12 Example lbmreq.cs

Source code: lbmreq.cs:

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

Usage: lbmreq [options] topic
Available options:
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -d sec = delay sending initial request sec seconds after
            source creation
  -h = help
  -i = send immediate requests
  -q = implement with EventQueues
  -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)
  -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
```

1.4.13 Example lbmresp.cs

Source code: lbmresp.cs:

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

Usage: lbmresp [options] topic
Available options:
  -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
  -h = help
  -l len = use len bytes for the length of each response
  -r responses = send responses messages for each request
  -q = implement with EventQueues
  -v = be verbose about each message
  -v -v = be even more verbose about each message
```

1.4.14 Example lbmsrc.cs

Source code: lbmsrc.cs:

```
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'
  -d delay = delay sending for delay seconds after source creation
  -D = Use SDM Messages
  -f = use hot-failover
  -i seq = hot-failover: begin sequencing with this 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 = used 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 sec = print stats every sec seconds
  -t filename = use filename contents as a recording of message sequence numbers
      (HF only!)
  -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'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.4.15 Example lbmStatistics.cs

Source code: lbmStatistics.cs:

```
General utility functions for UM example programs.
```

1.4.16 Example lbmtrreq.cs

Source code: lbmtrreq.cs:

```
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'
  -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
```

1.4.17 Example lbmwrcv.cs

Source code: lbmwrcv.cs:

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

Usage: [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 receiver after 100 messages
  -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'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.4.18 Example MinRcv.cs

Source code: MinRcv.cs:

```
Purpose: minimal subscriber application.

MinRcv.cs - Minimal receiver program.
See Quick Start document.
```

1.4.19 Example MinSrc.cs

Source code: MinSrc.cs:

```
Purpose: minimal publisher application.

MinSrc.cs - Minimal source program.
See Quick Start document.
```

1.4.20 Example umercv.cs

Source code: umercv.cs:

```
Purpose: Receive messages on a single topic via the UM .NET API.

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'
  -d qdelay = monitor event queue delay above qdelay usecs
  -D = Deregister after 1000 messages
  -e num_messages = send an Explicit ACK every num_messages messages
  -E = exit after source ends
  -h = help
  -i offset = use offset to calculate Registration ID
              (as source registration ID + offset)
              offset of 0 forces creation of regid by store
  -n nsrcs = stop topic resolution queries after nsrcs sources
```

```
  -N offset = display recovery sequence number info and set low seqnum to low+offset
  -q = use an LBM event queue
  -r msgs = delete receiver after msgs messages
  -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
  -z qsize = monitor event queue size above qsize in length
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'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.4.21 Example umesrc.cs

Source code: umesrc.cs:

```
Purpose: Send messages on a single topic via the UM .NET API

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'
  -D Send Deregistration after 1000 messages
  -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 = used 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 = verbose
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'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.4.22 Example umqrcv.cs

Source code: umqrcv.cs:

```
Purpose: Receive messages from a queue.

Usage: umqrcv [options] topic
Available options:
  -B broker = use broker specified by address
  -c filename = Use LBM configuration file filename.
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -d qdelay = monitor event queue delay above qdelay usecs
  -D = deregister upon exit
  -E = exit after source ends
  -e num_messages = send an Explicit ACK every num_messages 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 offset = display recovery sequence number info and set low seqnum to low+offset
  -S = exit after source ends, print throughput summary
  -s num_secs = print statistics every num_secs along with bandwidth
  -h = help
  -n nsrcs = stop topic resolution queries after nsrcs sources
  -q = use an LBM event queue
  -z qsize = monitor event queue size above qsize in length
  -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'
  --monitor-format-opts OPTS = use OPTS as format module options
  --monitor-appid ID = use ID as application ID string
```

1.4.23 Example umqsrc.cs

Source code: umqsrc.cs:

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

Usage: umqsrc [options] topic
Available options:
  -A cfg = use ULB Application Sets given by cfg
  -B broker = use broker specified by address
  -c filename = Use LBM configuration file filename.
```

```
                Multiple config files are allowed.
                Example:  '-c file1.cfg -c file2.cfg'
  -d NUM = delay sending for NUM seconds after source creation
  -f NUM = allow NUM unstabilized messages in flight
  -h = help
  -i = display message IDs for sent message
  -l len = send messages of len bytes
  -L linger = linger for linger seconds before closing context
  -M msgs = send msgs number of messages
  -N = display sequence number information source events
  -m NUM = send at NUM messages per second (trumped by -f)
  -n = used 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 X
     -Y = Send using named UMQ index for broker sources
  -v = verbose
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'
  --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
```

1.4.24 Example VerifiableMessage.cs

Source code: VerifiableMessage.cs:

```
Utility functions to produce randomized messages whose contents
can be checked.
```