ØMQ

Fabio Galuppo, M.Sc.

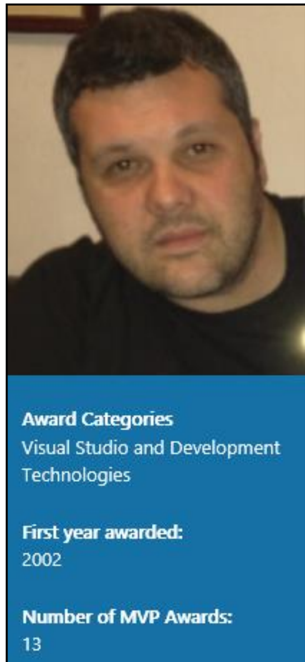
http://fabiogaluppo.com e http://simplycpp.com/

fabiogaluppo@acm.org

@FabioGaluppo

Microsoft MVP Visual Studio and Development Technologies

https://mvp.microsoft.com/en-us/PublicProfile/9529



Award Categories
Visual Studio and Development Technologies

First year awarded:
2002

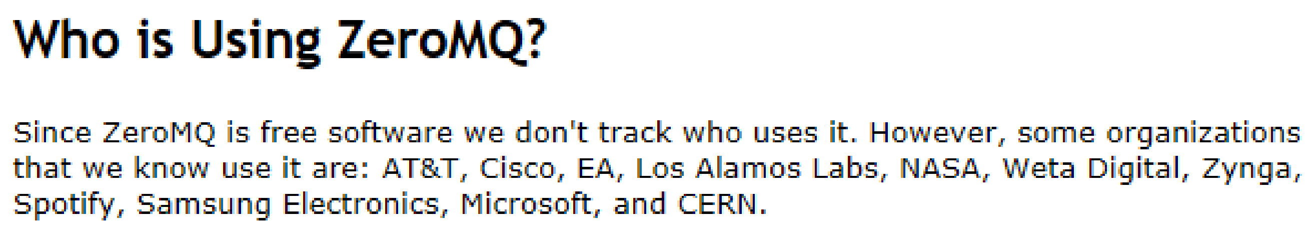
Number of MVP Awards:
13

O que é ZeroMQ?

- *Intelligent socket library for messaging*
  - Variedade nos padrões de comunicação
    - Request-Reply, Publisher-Subscriber, Push-Pull, Dealer-Router, …
  - Suporta: *inproc*, *IPC*, *TCP*, *TIPC*, *multicast*
- Modelo de concorrência baseado em atores (*Erlang-style)*
- *Open Source*
- Multiplas plataformas
- Diversas linguagens (mais de 30)
  - C, C++, Java, C#, Python, …
- *Deploy* simples (uma *library*)
- Alta performance
  - http://zeromq.org/results:multicore-tests
    - ~6 milhões de mensagens por segundo

http://zeromq.org

Destaque



Who is Using ZeroMQ?

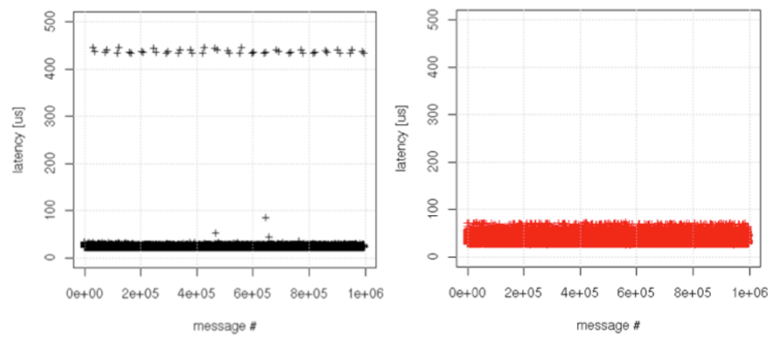
Since ZeroMQ is free software we don't track who uses it. However, some organizations that we know use it are: AT&T, Cisco, EA, Los Alamos Labs, NASA, Weta Digital, Zynga, Spotify, Samsung Electronics, Microsoft, and CERN.

- Cisco: The Avalanche Project: When High Frequency Trading Meets Traffic Classification
- CERN: MIDDLEWARE TRENDS AND MARKET LEADERS 2011

http://accelconf.web.cern.ch/AccelConf/icalepcs2011/papers/frbhmult05.pdf

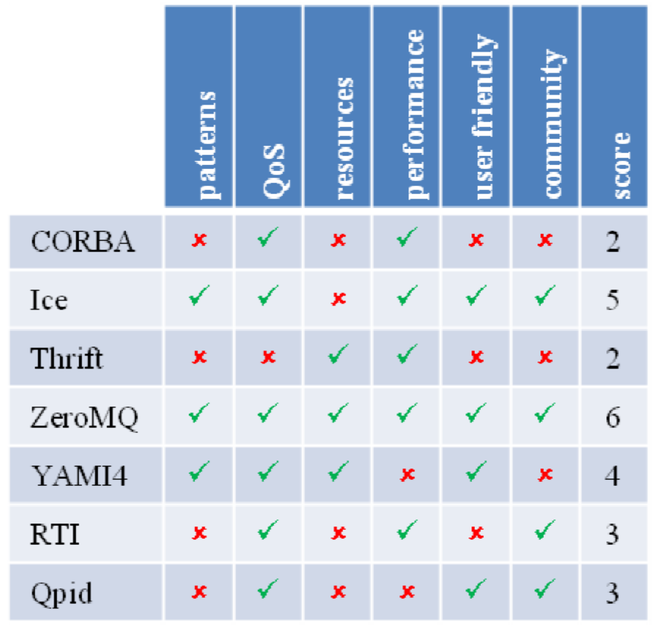
CHOOSING A ROBUST MESSAGING SYSTEM

The foundation of our architecture relies on choosing the right messaging library. This will be a key factor to determine how well our data processing pipeline will perform. I will save you from all the details of a descriptive comparison between all the technology, but ZeroMQ is one of the best messaging middleware available, and it is very well known in the finance world. It also provides an amazing paradigm to build a distributed system with different message passing patterns. During my analysis, some important metrics caught my attention :



Source: http://zeromq.org/results:rt-tests-v031

https://umbrella.cisco.com/blog/2015/11/05/the-avalanche-project-when-high-frequency-trading-meets-traffic-classification/

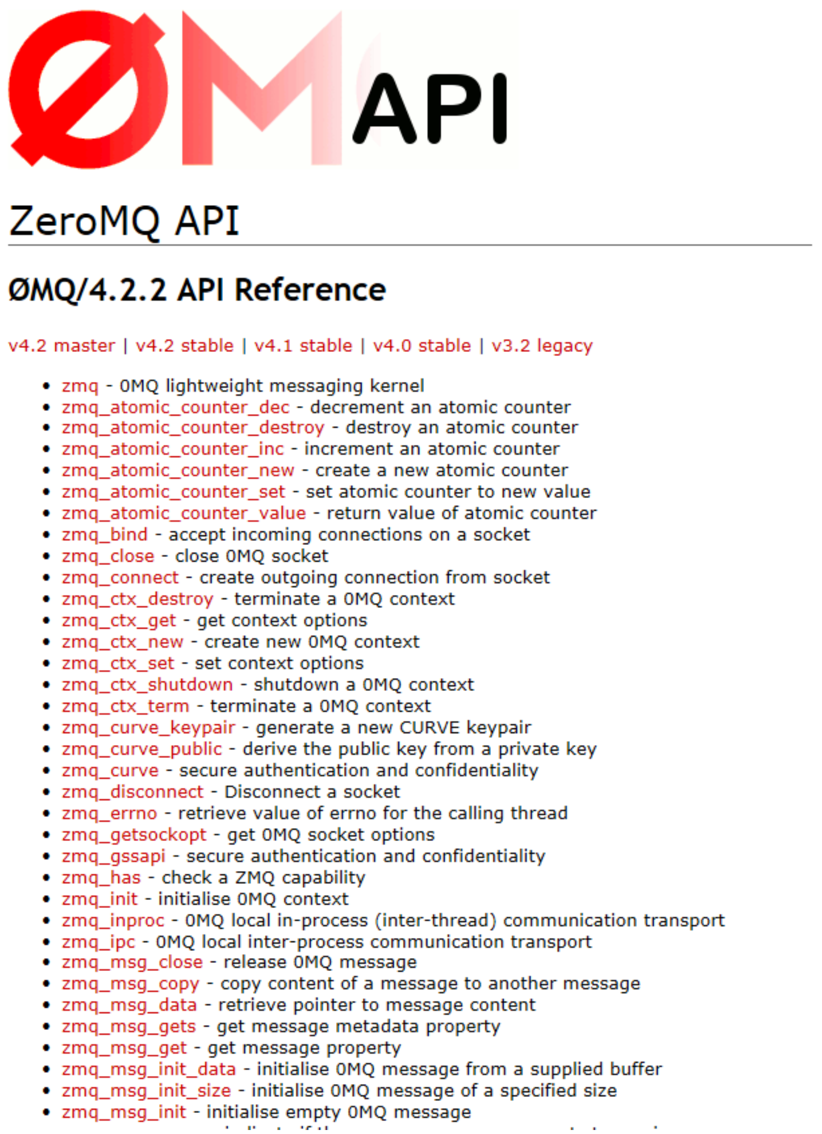


	patterns	QoS	resources	performance	user friendly	community	score
CORBA	✗	✓	✗	✓	✗	✗	2
Ice	✓	✓	✗	✓	✓	✓	5
Thrift	✗	✗	✓	✓	✗	✗	2
ZeroMQ	✓	✓	✓	✓	✓	✓	6
YAMI4	✓	✓	✓	✗	✓	✗	4
RTI	✗	✓	✗	✓	✗	✓	3
Qpid	✗	✓	✗	✗	✓	✓	3

Figure 3: Summary of evaluated middleware products.

ZeroMQ: APIs essenciais

- http://api.zeromq.org/
  - zmq_ctx_new - create new 0MQ context
  - zmq_ctx_term - terminate 0MQ context
  - zmq_socket - create 0MQ socket
  - zmq_close - close 0MQ socket
  - zmq_bind - accept incoming connections on a socket
  - zmq_connect - create outgoing connection from socket
  - zmq_send - send a message part on a socket
    - zmq_send, zmq_sendmsg, zmq_send_const
  - zmq_recv - receive a message part from a socket
    - zmq_recv, zmq_recvmsg
  - zmq_setsockopt - set 0MQ socket options
  - zmq_getsockopt - get 0MQ socket options



ØM API

ZeroMQ API

ØMQ/4.2.2 API Reference

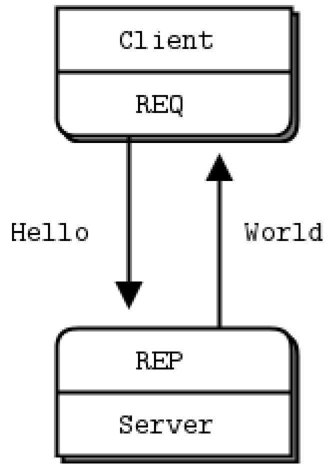
v4.2 master | v4.2 stable | v4.1 stable | v4.0 stable | v3.2 legacy

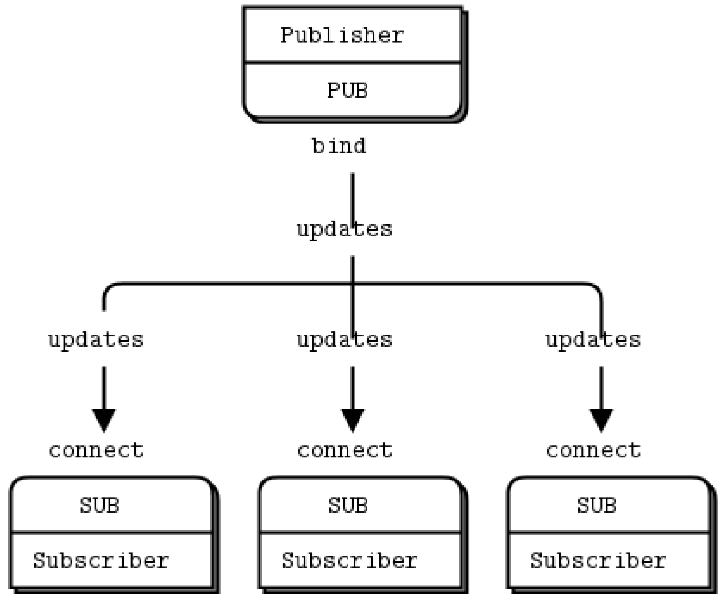
- zmq - 0MQ lightweight messaging kernel
- zmq_atomic_counter_dec - decrement an atomic counter
- zmq_atomic_counter_destroy - destroy an atomic counter
- zmq_atomic_counter_inc - increment an atomic counter
- zmq_atomic_counter_new - create a new atomic counter
- zmq_atomic_counter_set - set atomic counter to new value
- zmq_atomic_counter_value - return value of atomic counter
- zmq_bind - accept incoming connections on a socket
- zmq_close - close 0MQ socket
- zmq_connect - create outgoing connection from socket
- zmq_ctx_destroy - terminate a 0MQ context
- zmq_ctx_get - get context options
- zmq_ctx_new - create new 0MQ context
- zmq_ctx_set - set context options
- zmq_ctx_shutdown - shutdown a 0MQ context
- zmq_ctx_term - terminate a 0MQ context
- zmq_curve_keypair - generate a new CURVE keypair
- zmq_curve_public - derive the public key from a private key
- zmq_curve - secure authentication and confidentiality
- zmq_disconnect - Disconnect a socket
- zmq_errno - retrieve value of errno for the calling thread
- zmq_getsockopt - get 0MQ socket options
- zmq_gssapi - secure authentication and confidentiality
- zmq_has - check a ZMQ capability
- zmq_init - initialise 0MQ context
- zmq_inproc - 0MQ local in-process (inter-thread) communication transport
- zmq_ipc - 0MQ local inter-process communication transport
- zmq_msg_close - release 0MQ message
- zmq_msg_copy - copy content of a message to another message
- zmq_msg_data - retrieve pointer to message content
- zmq_msg_gets - get message metadata property
- zmq_msg_get - get message property
- zmq_msg_init_data - initialise 0MQ message from a supplied buffer
- zmq_msg_init_size - initialise 0MQ message of a specified size
- zmq_msg_init - initialise empty 0MQ message

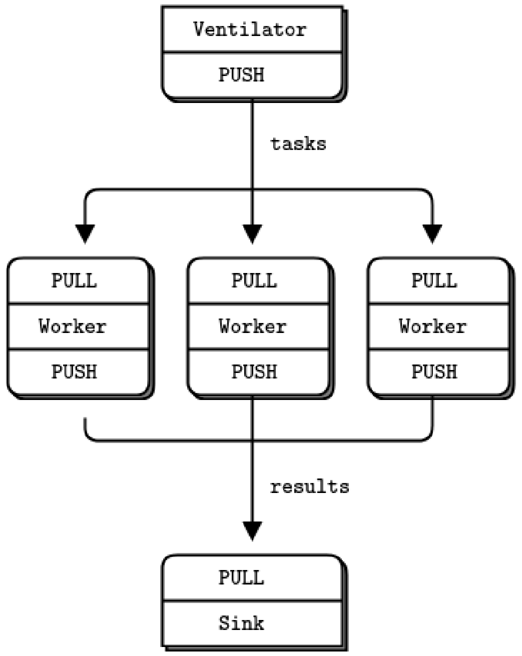
http://api.zeromq.org

ZeroMQ: *Patterns*

- *Request Reply*
- *Publisher Subscriber*
- *Parallel Pipeline*





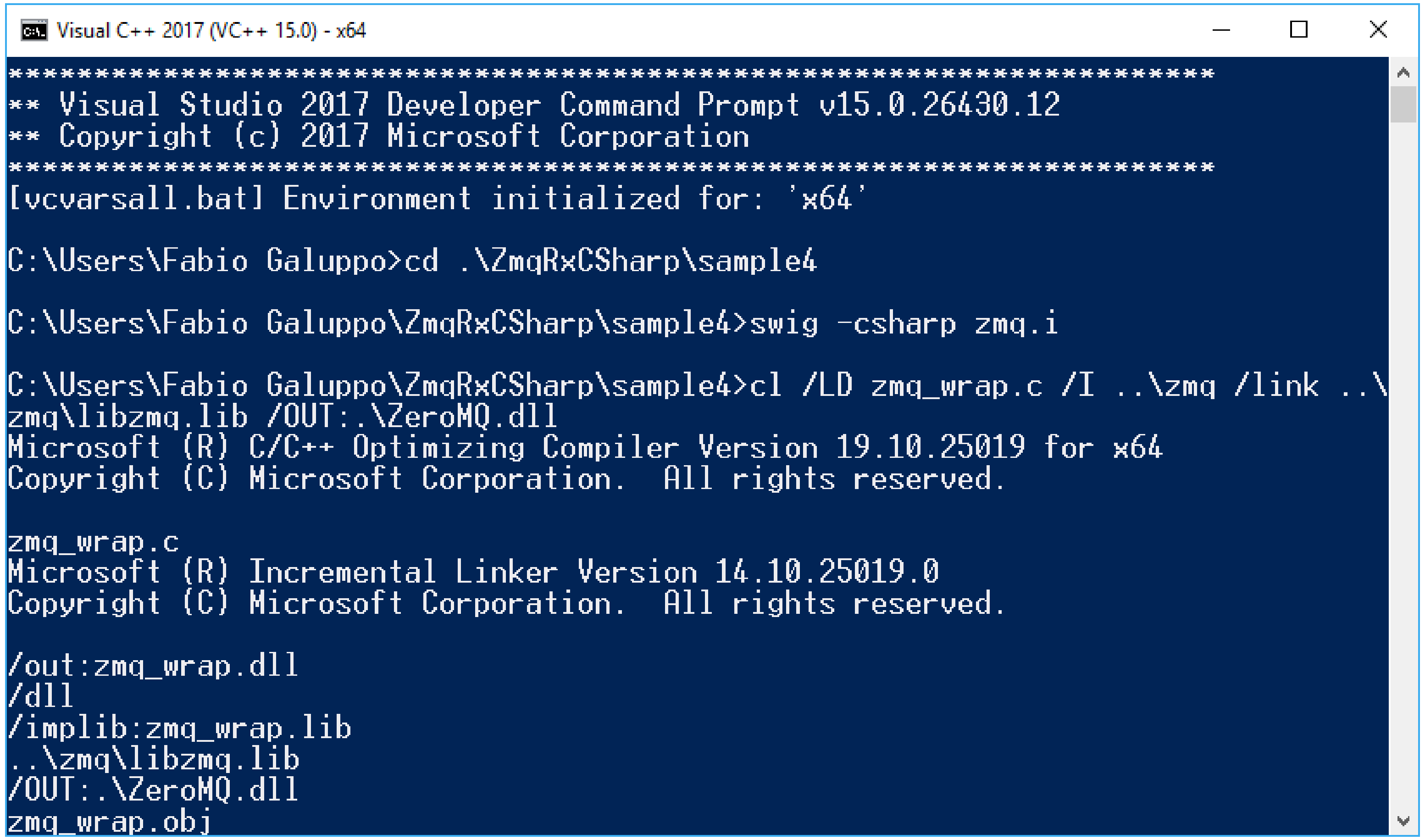


…

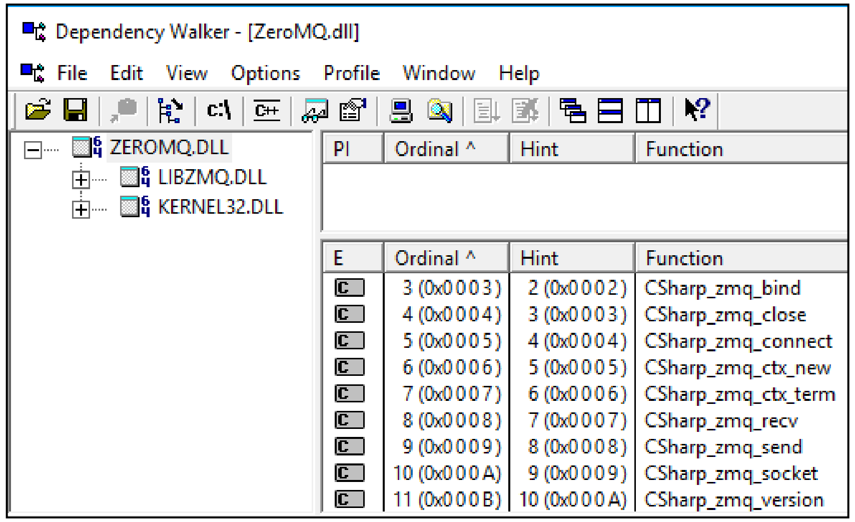
http://zguide.zeromq.org/page:all#Messaging-Patterns

ZeroMQ: SWIG

- http://www.swig.org


Visual C++ 2017 (VC++ 15.0) - x64

```
**********************************************************************
** Visual Studio 2017 Developer Command Prompt v15.0.26430.12
** Copyright (c) 2017 Microsoft Corporation
**********************************************************************
[vcvarsall.bat] Environment initialized for: 'x64'

C:\Users\Fabio Galuppo>cd .\ZmqRxCSharp\sample4

C:\Users\Fabio Galuppo\ZmqRxCSharp\sample4>swig -csharp zmq.i

C:\Users\Fabio Galuppo\ZmqRxCSharp\sample4>cl /LD zmq_wrap.c /I ..\zmq /link ..\
zmq\libzmq.lib /OUT:.\ZeroMQ.dll
Microsoft (R) C/C++ Optimizing Compiler Version 19.10.25019 for x64
Copyright (C) Microsoft Corporation.  All rights reserved.

zmq_wrap.c
Microsoft (R) Incremental Linker Version 14.10.25019.0
Copyright (C) Microsoft Corporation.  All rights reserved.

/out:zmq_wrap.dll
/dll
/implib:zmq_wrap.lib
..\zmq\libzmq.lib
/OUT:.\ZeroMQ.dll
zmq_wrap.obj
```

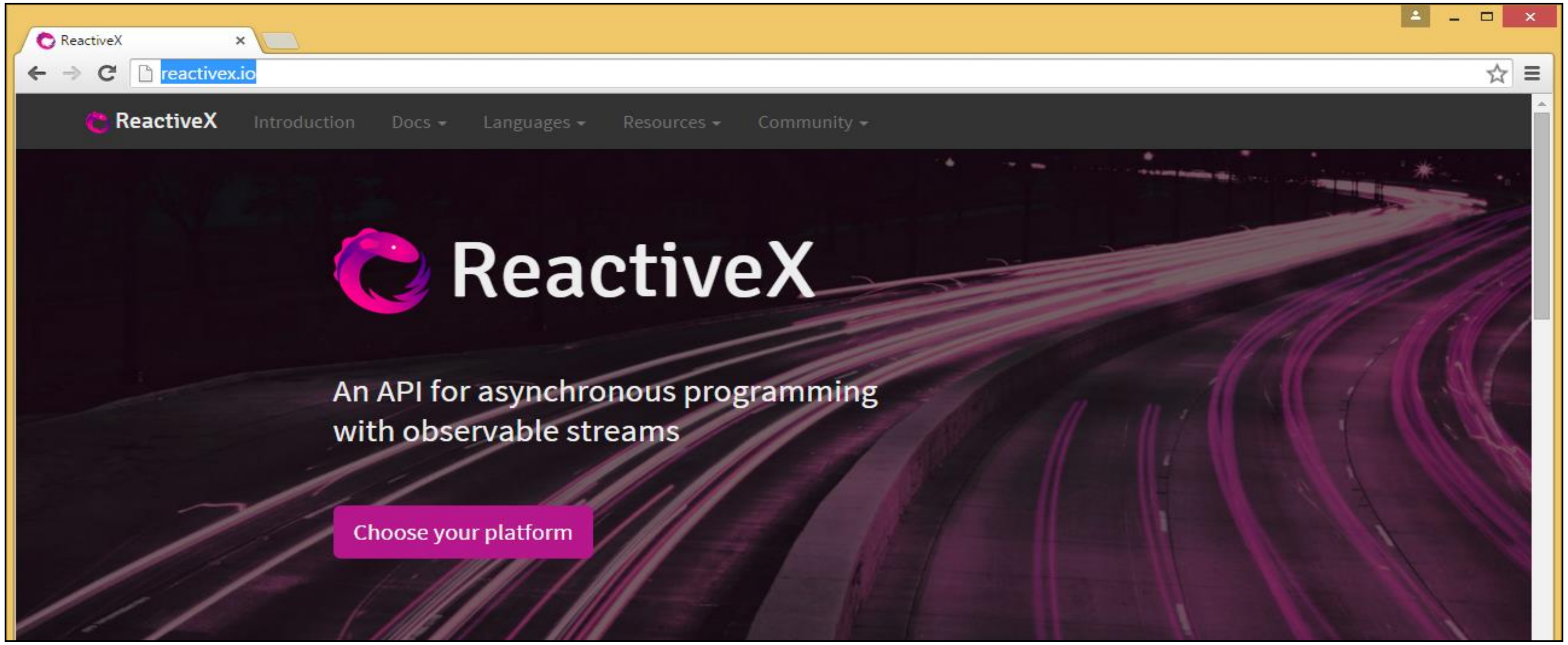

Dependency Walker - [ZeroMQ.dll]
File Edit View Options Profile Window Help
ZEROMQ.DLL
LIBZMQ.DLL
KERNEL32.DLL

PI	Ordinal ^	Hint	Function

E	Ordinal ^	Hint	Function
C	3 (0x0003)	2 (0x0002)	CSharp_zmq_bind
C	4 (0x0004)	3 (0x0003)	CSharp_zmq_close
C	5 (0x0005)	4 (0x0004)	CSharp_zmq_connect
C	6 (0x0006)	5 (0x0005)	CSharp_zmq_ctx_new
C	7 (0x0007)	6 (0x0006)	CSharp_zmq_ctx_term
C	8 (0x0008)	7 (0x0007)	CSharp_zmq_recv
C	9 (0x0009)	8 (0x0008)	CSharp_zmq_send
C	10 (0x000A)	9 (0x0009)	CSharp_zmq_socket
C	11 (0x000B)	10 (0x000A)	CSharp_zmq_version

http://www.swig.org/Doc3.0/SWIGDocumentation.html#CSharp

ReactiveX



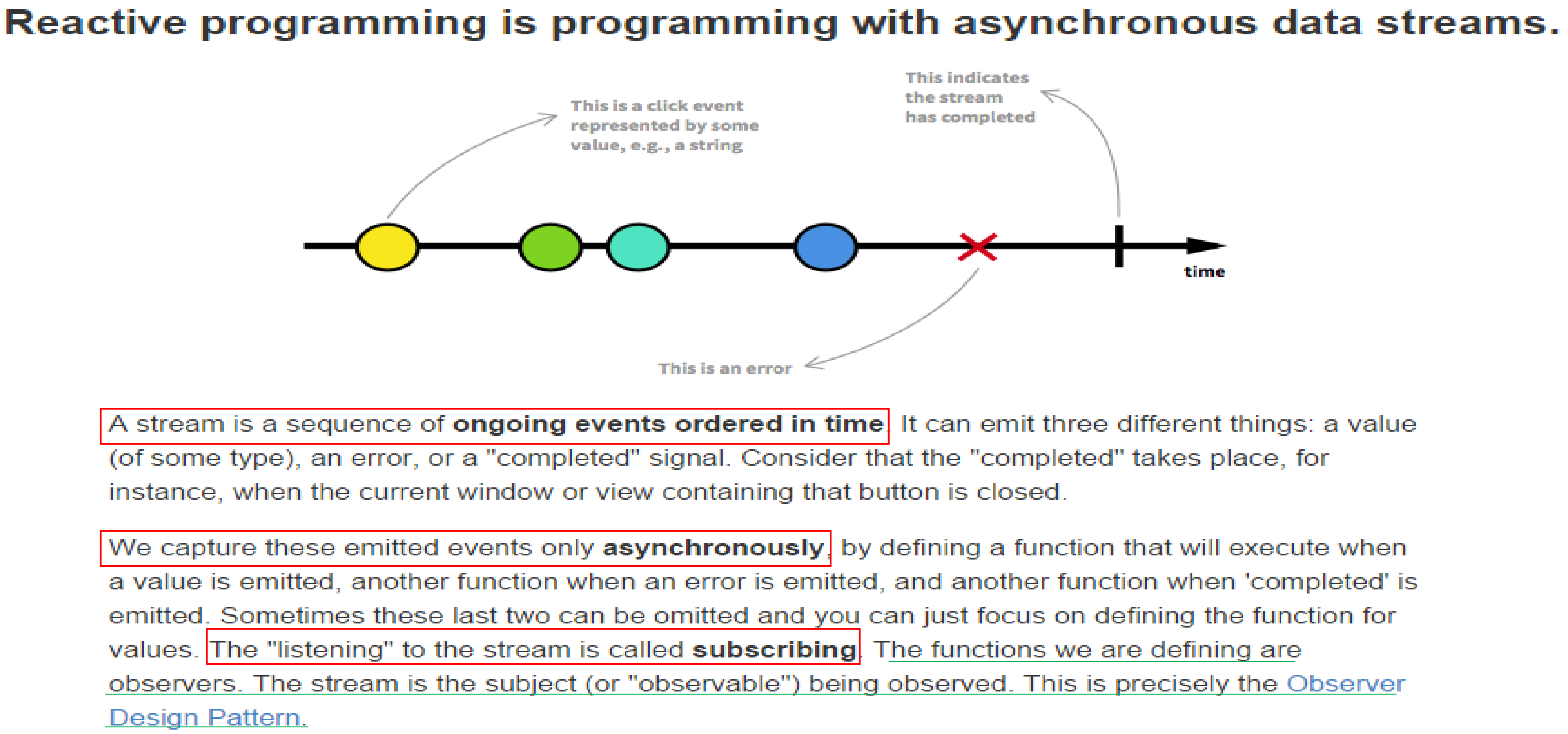
An API for asynchronous programming with observable streams

The Observer pattern done right

ReactiveX is a combination of the best ideas from
the Observer pattern, the Iterator pattern, and functional programming

http://reactivex.io

Programação Reativa



Reactive programming is programming with asynchronous data streams.

This is a click event represented by some value, e.g., a string
This indicates the stream has completed
time
This is an error

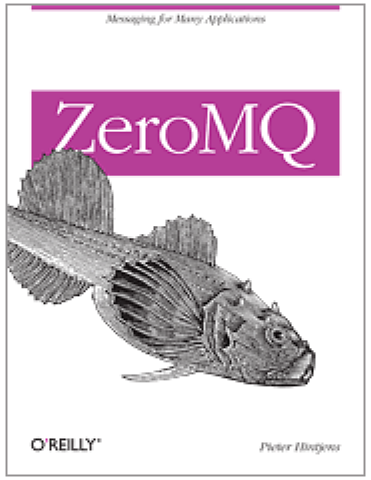
A stream is a sequence of **ongoing events ordered in time**. It can emit three different things: a value (of some type), an error, or a "completed" signal. Consider that the "completed" takes place, for instance, when the current window or view containing that button is closed.

We capture these emitted events only **asynchronously**, by defining a function that will execute when a value is emitted, another function when an error is emitted, and another function when 'completed' is emitted. Sometimes these last two can be omitted and you can just focus on defining the function for values. The "listening" to the stream is called **subscribing**. The functions we are defining are observers. The stream is the subject (or "observable") being observed. This is precisely the Observer Design Pattern.

https://gist.github.com/staltz/868e7e9bc2a7b8c1f754

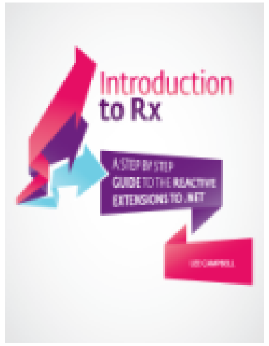
Referências

- http://zeromq.org
- http://reactivex.io
- ZeroMQ
  - http://shop.oreilly.com/product/0636920026136.do
  - http://zguide.zeromq.org/



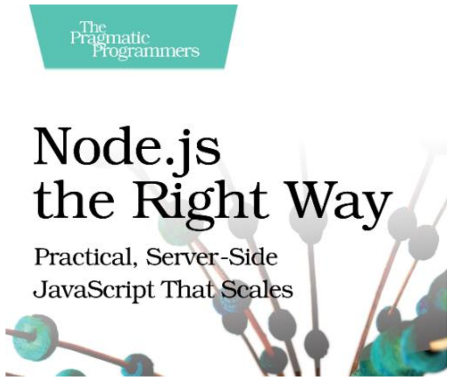
ZeroMQ
Messaging for Many Applications
By Pieter Hintjens
Publisher: O'Reilly Media
Final Release Date: March 2013
Pages: 516
★★★★★ 5.0
Read 6 Reviews | Write a Review

- Introduction to Rx
  - http://www.introtorx.com/
  - ReactiveX para .NET



- Node.js the Right Way
  - https://pragprog.com/book/jwnode/node-js-the-right-way
  - Um capítulo dedicado a ZeroMQ

- Robust Messaging Services
  - Advantages of ØMQ
  - Importing External Modules with npm
  - Message-Publishing and -Subscribing
  - Responding to Requests
  - Routing and Dealing Messages
  - Clustering Node.js Processes
  - Pushing and Pulling Messages
  - Wrapping Up


The Pragmatic Programmers
Node.js the Right Way
Practical, Server-Side JavaScript That Scales

ØMQ +

Fabio Galuppo, M.Sc.

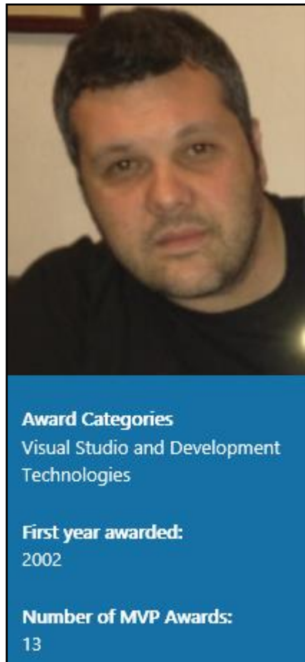
http://fabiogaluppo.com e http://simplycpp.com/

fabiogaluppo@acm.org

@FabioGaluppo

Microsoft MVP Visual Studio and Development Technologies

https://mvp.microsoft.com/en-us/PublicProfile/9529



Award Categories
Visual Studio and Development Technologies

First year awarded:
2002

Number of MVP Awards:
13