Market Technologies Pty Ltd

Fix8 and Fix8Pro© Product Overview

---

1 Contents

A. Introduction

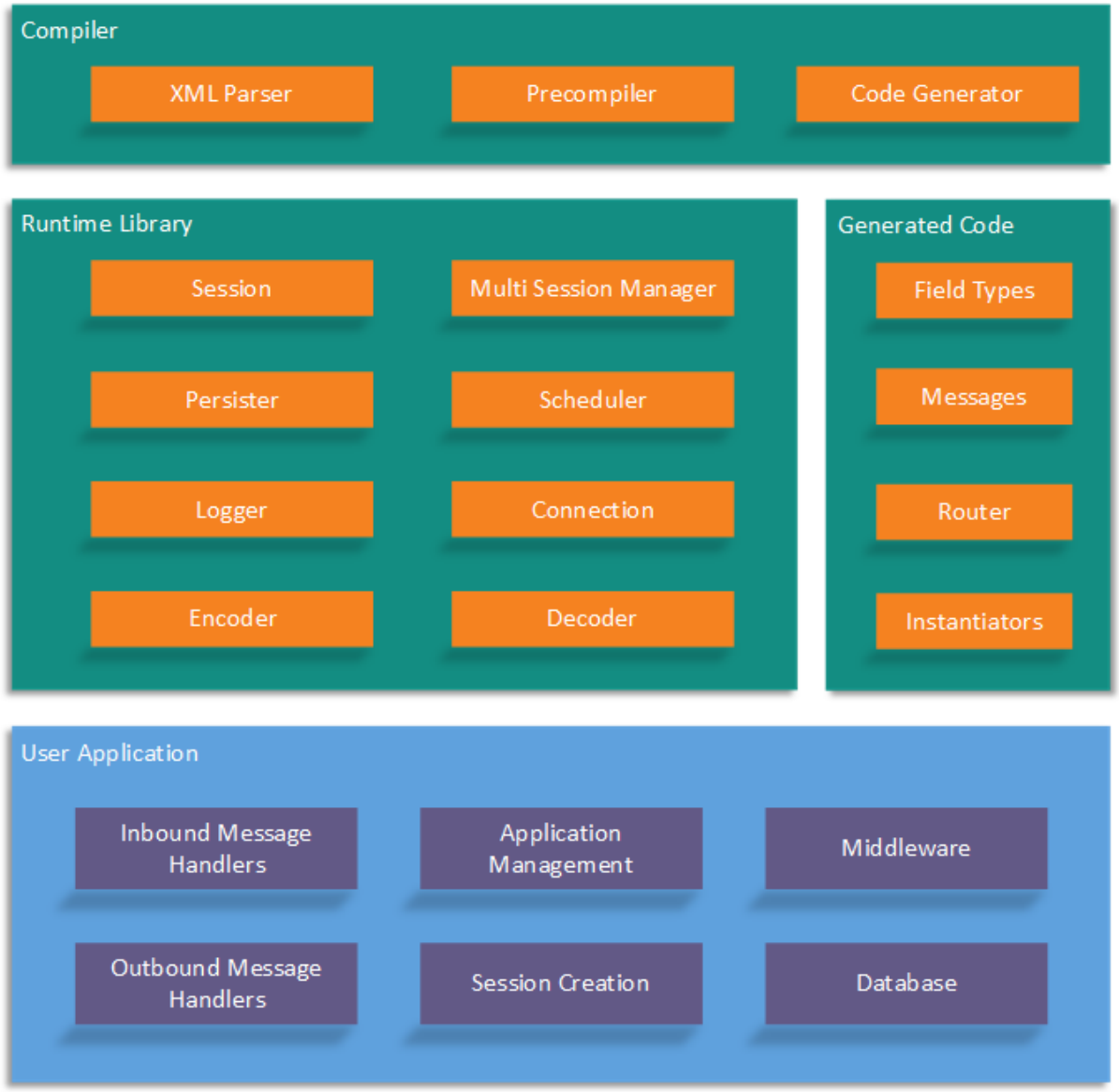
Fix8 is a C++ application framework featuring complete schema driven customisation, high performance and fast application development. The system is comprised of a compiler for generating C++ message and field encoders, decoders and instantiation tables; a runtime library to support the generated code and framework; and a set of complete client/server test applications.

Fix8 comes in two versions:

- **Fix8 Open Source** - the community supported version. With complete source code, sample applications and online documentation.

- **Fix8Pro** - commercially supported version. Provides additional features and a subscription based SLA.

B. System Architecture

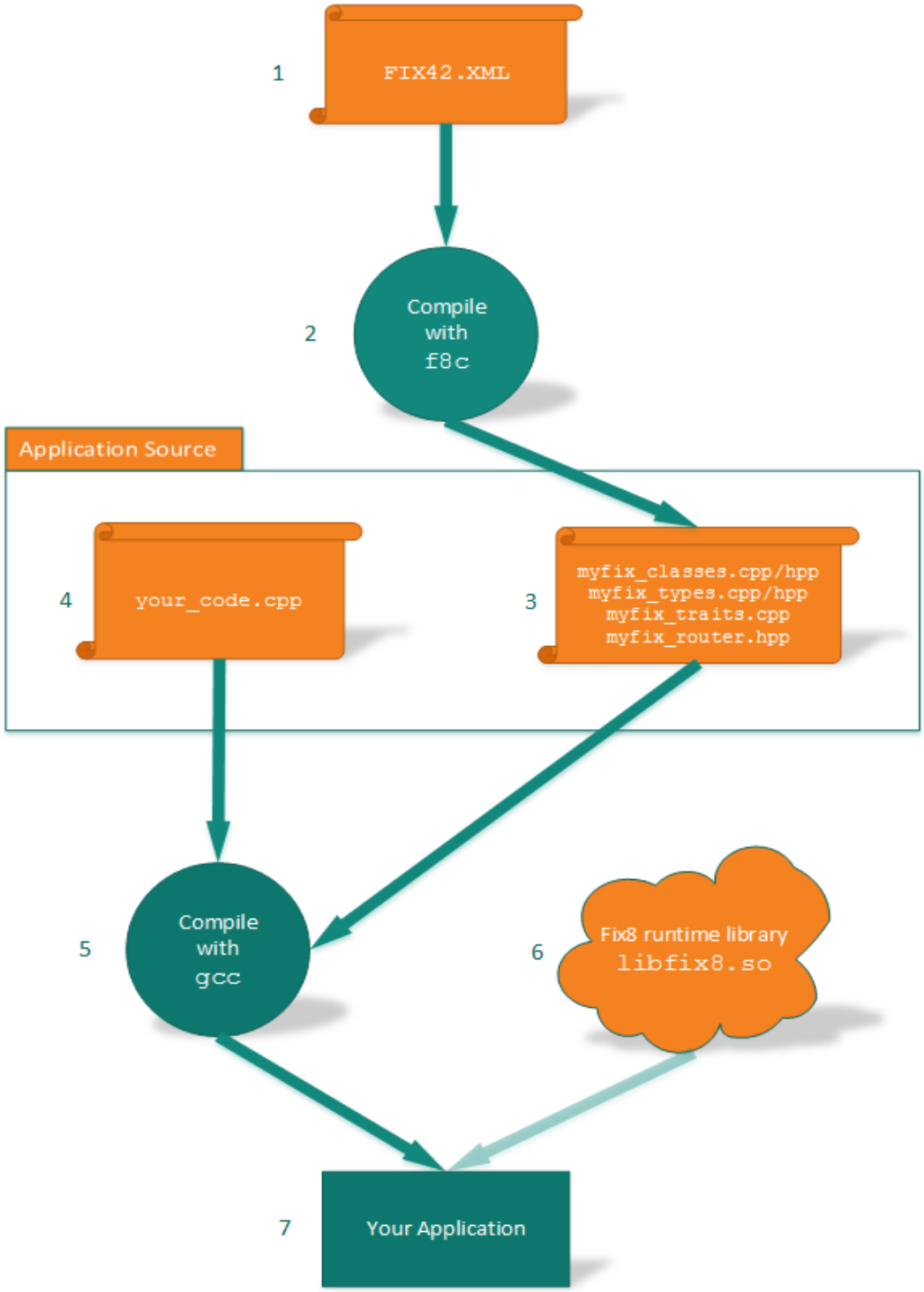


i. System architecture overview

Fix8 (green and orange) is a development library which provides a highly evolved FIX application framework. Your application (blue and purple) is built on this framework.

C. How it works

The following diagram summarises the steps to develop an application using Fix8.



ii. How it works

1. **Prepare your XML.** Use one of the supplied standard XML FIX schemas or create or modify your own custom schema. You have complete control over every message and field that will be spoken by your interface.

2. **Compile the XML.** Run the Fix8 compiler f8c over your XML to generate a set of `cpp` and `hpp` source files. You can control the `c++` namespace, class prefix and so forth to neatly frame the code in your application.

3. **Use the generated code.** Integrate the generated code into your application. The Fix8 compiler can even generate the session and dispatcher class stubs for you to fill in. Fix8 can support multiple FIX variants within one application if required.

4. **Implement your application.** Your business code goes here. Fix8 provides a robust application framework on which you can build your interface. Fix8 provides all the necessary services to establish a session, send and receive messages, create logs and so forth. Fix8 supports client or server modes.

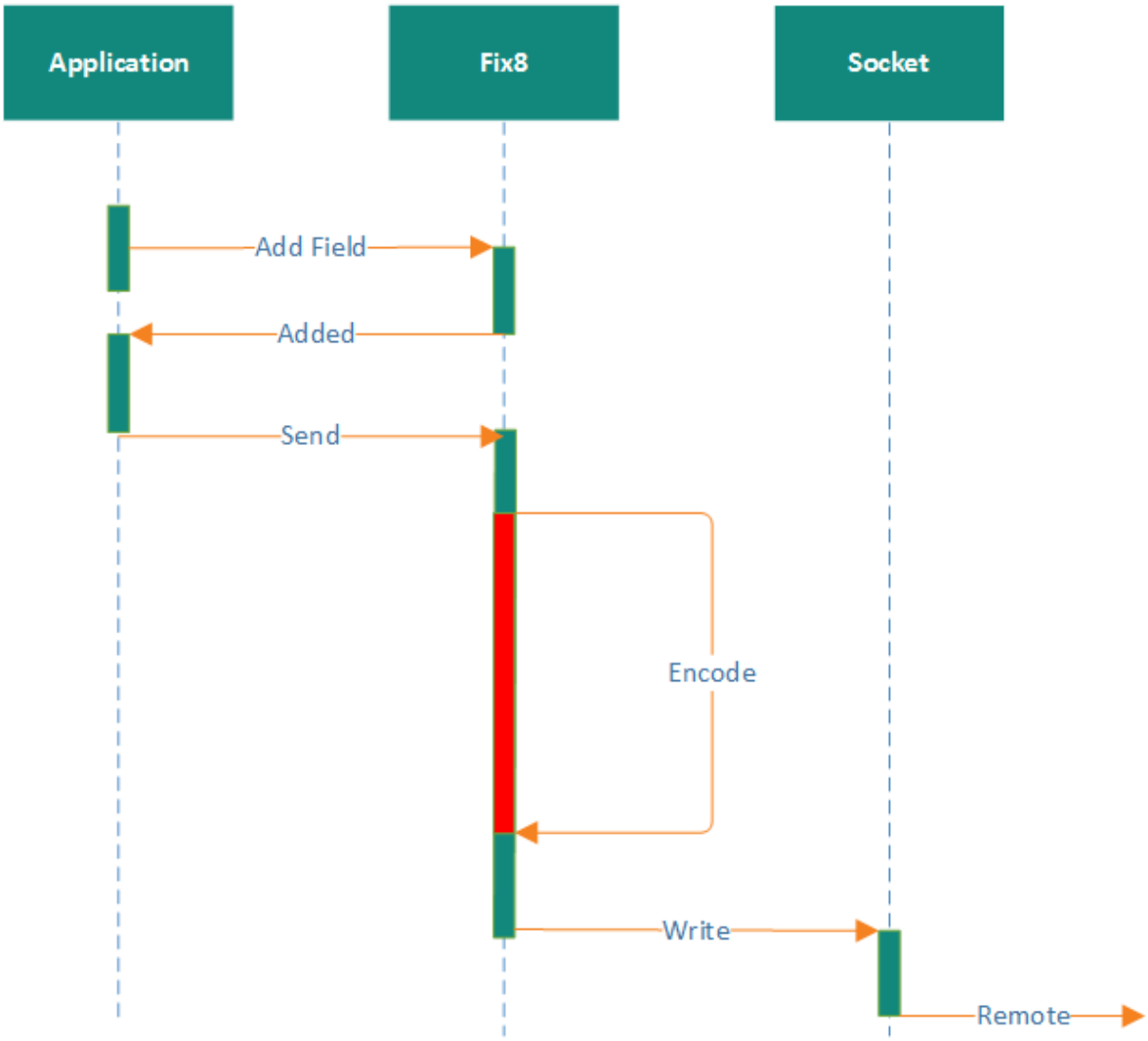
5. **Compile your application.** Compile all the source in your application. Your compiler will optimise the generated code.

6. **Link against the Fix8 Runtime Library.** Your application is then dynamically linked against the Fix8 Runtime library.

7. **Your final application.** Use the supported XML configuration system to prepare your interface for connection.

D. Summary of Features (All Versions)

1) **High performance.** Fix8 is the fastest C++ Open Source FIX framework.
   a) For the same message[1], Fix8 encodes 3.2 times faster and decodes 3.0 times faster averaging 3.0 times over *Quickfix* and reduces encode latency by 69% and reduces decode latency by 67%. The following sequence diagram indicates where the encode measurement is made.



iii. Encode measurement sequence

   b) On typical hardware, client `NewOrderSingle` encode latency is now 1.0µs, and `ExecutionReport` decode 1.9µs.

[1] Test environment was Fedora release 18 (Spherical Cow); Linux Version 3.9.2-200.fc18.x86_64 GNU/Linux; 32 core Intel(R) Xeon(R) CPU E5-2690 @ 2.90GHz 20480 KB Cache; 128GB RAM. GCC version 4.7.2; Poco version 1.4.6; gperftools 2.0 (for tcmalloc); quickfix version 1.13.3; tbb version 4.2 (tbb42_20131003oss); 185691 BogoMIPS Total.

2) **Flexibility.** Fix8 supports any FIX version or custom variant:

a) Fix8 supports standard FIX4.X to FIX5.X and FIXT1.X.
b) Custom FIX variants can be easily used. For example, if you use a modified FIX 4.4 with custom tags or messages, Fix8 can use your custom FIX instead of the standard FIX 4.4. If you need to add new fields, just edit your XML schema and Fix8 takes care of the rest.
c) Multiple FIX variants can be used in the same application - for example your application could initiate FIX 4.4 and FIX 4.2 connections.

3) **Portability.** Fix8 runs on many different platforms, including:

a) Linux
b) OSX
c) Windows (32/64, VS2012, VS2013, VS2015, VS2017)
d) Intel IA32, x64-64, AMD64, Itanium, PowerPC, ARMv7 and ARMv8-A.

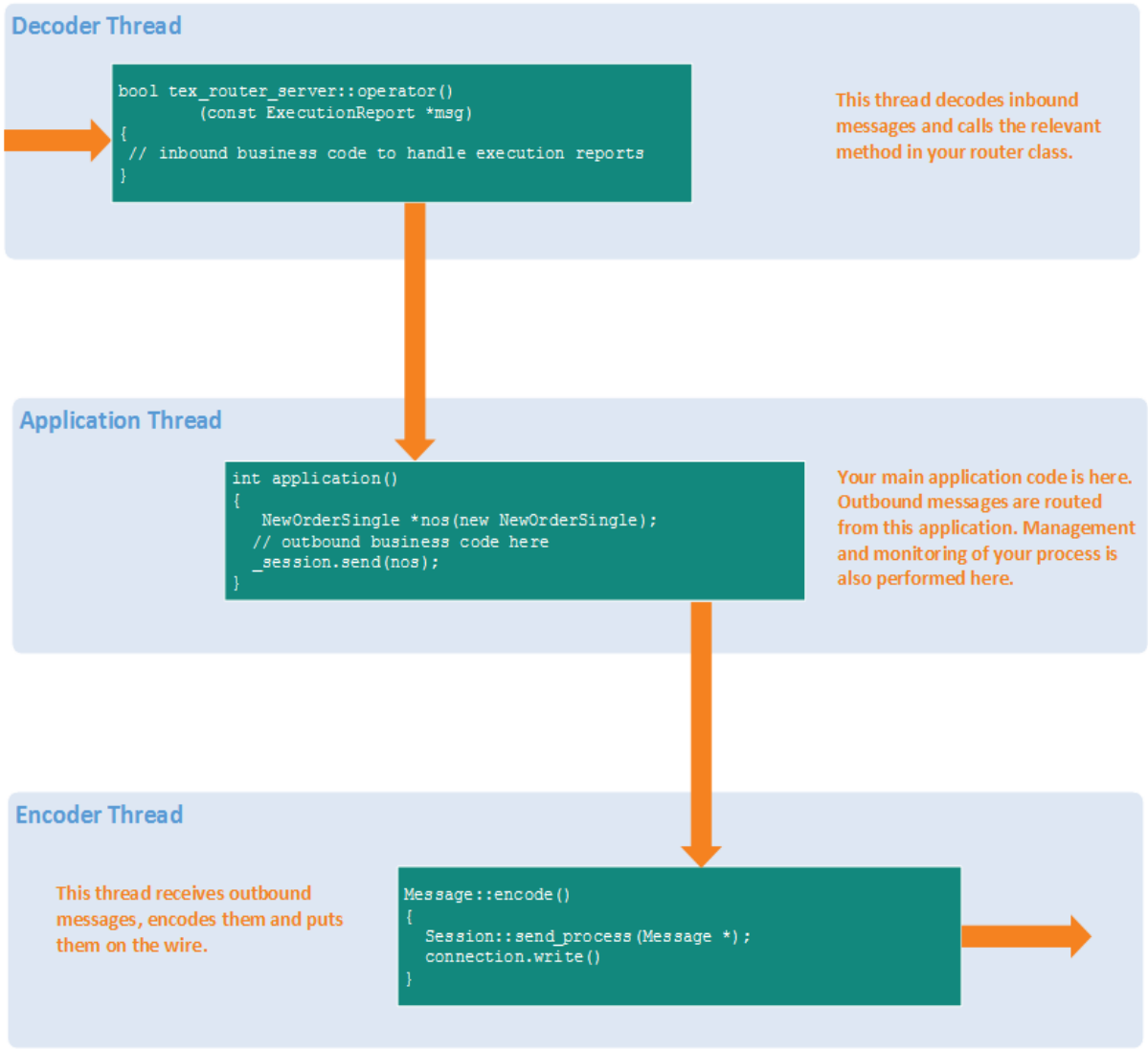
4) **Functionality.** Fix8 is a complete C++ FIX framework including:

a) Connection classes for client and server sessions (including SSL)
b) Native support for the standard FIX field types (e.g. Amt, LocalMktDate, Currency, Qty, UTCTimeStamp). The underlying C++ types are also available for each FIX field type (such as bool, float, char, string).
c) Verbose FIX message printer which can generate human readable logs and reports
d) Multi-session server/client managers that help you control and manage multiple servers or multiple client connections within an application
e) Asynchronous logger which provides fast, reliable non-blocking logging to multiple programmable log targets. The logger is highly configurable with fine grained verbosity.
f) Asynchronous message persister provides robust retransmission capability. Support for Redis, Aerospike, MongoDB, Memcached and BerkeleyDB is also provided.
g) Built-in application specialisation. Fix8 is designed to allow applications to specialise critical behaviour to suit special purposes. For example, if you need to handle logins differently, or if you need to provide store and forward capabilities.
h) Easy to use XML based configuration system providing structured session and application setup. The following is an example client session configuration

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<fix8>
   <default role="initiator"
            fix_version="1100"
            ip="127.0.0.1"
            session_log="session_log_file"
            protocol_log="protocol_log_file"
            login_retry_interval="3000"
            reset_sequence_numbers="false"
            connect_timeout="3"
            heartbeat_interval="10"
            tcp_nodelay="true"
            always_seqnum_assign="false"
            process_model="threaded"
            enforce_compids="false"
            login_retries="5"
            tabsize="3"
            persist="file0" />

   <session name="TESTA"
            sender_comp_id="TESTA_TEX"
            target_comp_id="TEX_SERVER"
            port="11001"
            active="true" />

   <session name="TESTB"
            sender_comp_id="TESTB_TEX"
            target_comp_id="TEX_SERVER"
            port="11001"
            active="true" />

   <persist name="file0"
            type="file" dir="./run"
            use_session_id="true"
            rotation="5"
            db="client" />

   <log     name="session_log_file"
            type="session"
            filename="./run/testframe_client_session.log"
            rotation="5"
            flags="timestamp|sequence|thread"/>

   <log     name="protocol_log_file"
            type="protocol"
            filename="./run/testframe_client_protocol.log"
            rotation="5"
            flags="append|inbound|outbound|direction|sequence"/>
</fix8>
```

iv. Sample Client XML configuration

5) **Robust configurable threading.** Fix8 supports a *pipelined*, *threaded* and *coroutine* (fibres) based threading model. The following diagram shows the *threaded* model:



v. *Threaded* threading model

6) **FIX Capability.** Fix8 provides full FIX capabilities, including:

a) Support for field and value domain validation
b) Support for mandatory/optional field assertion, field ordering and well-formedness testing
c) Retransmission and standard session semantics
d) Support nested components and repeating groups to any depth. The Fix8 compiler and runtime library take the pain out of using repeating groups.

7) **Community Support.** Both the open source and commercial versions of Fix8 benefit from the wide community user base. This provides timely bug reports and fixes, production testing and burn-ins with new versions, and a responsive support

knowledge base.

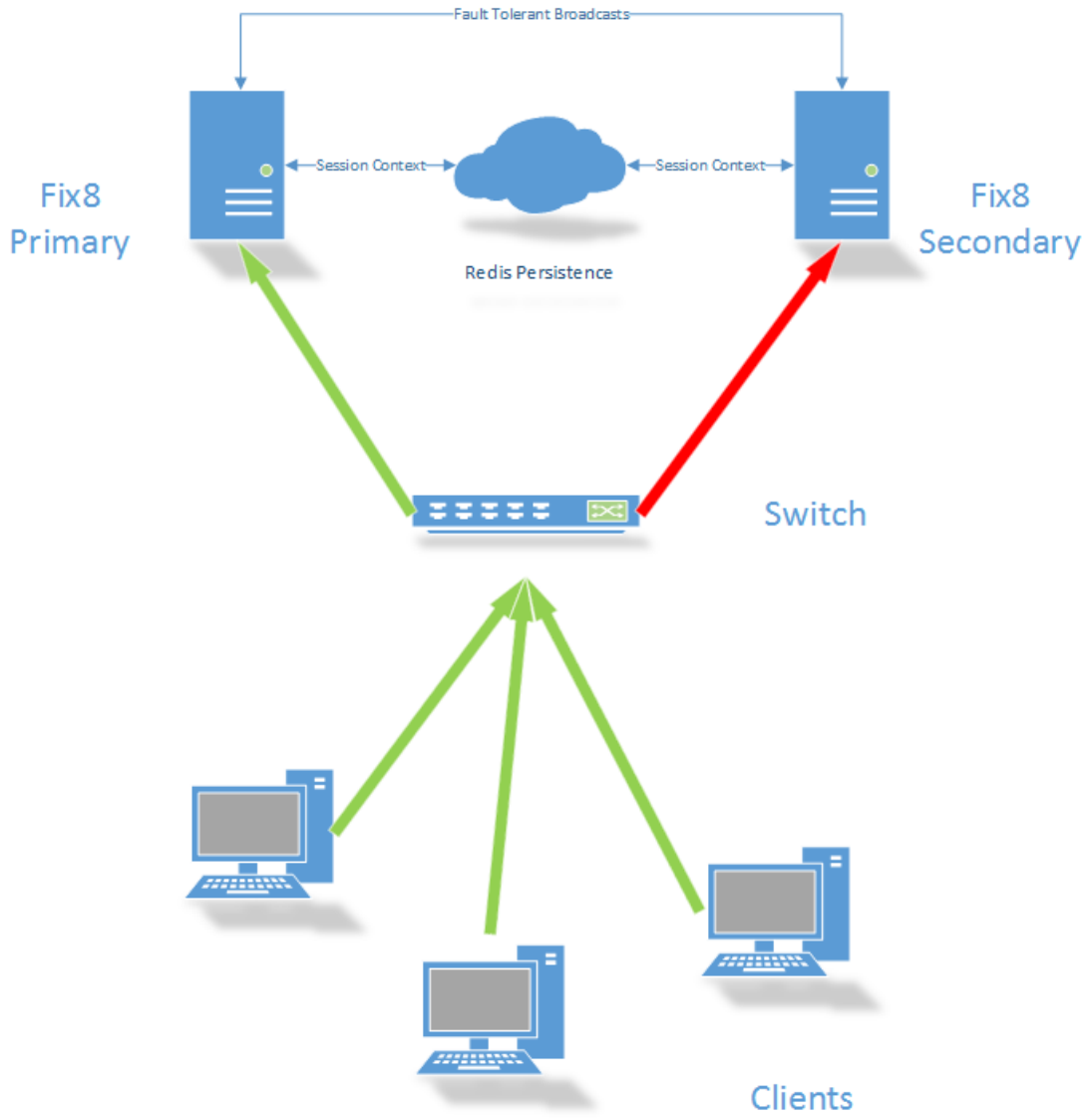
8) **Customisable Behaviour.** Fix8 provides an extensive set of hooks in the standard Session for you to implement your own customised behaviour or to extend the default behaviour provided by the framework. The following table shows which behaviour can be customised:

FIX8::Session method	Description	Override
handle_logon	Client or Server login	Optional
handle_logout	Client or Server logout	Optional
handle_heartbeat	Heartbeat received	Optional
handle_resend_request	Resend Request received	Optional
handle_sequence_reset	Sequence Reset received	Optional
handle_test_request	Test Request received	Optional
handle_reject	Reject received	Optional
handle_admin	Handle an admin message (e.g. Logon)	Optional
handle_application	Handle an application (e.g. ExecutionReport)	Required
state_change	A session state has changed (e.g waiting for logon to logon received)	Optional
modify_outbound	Enable a final modification of any outbound FIX message	Optional
authenticate	Provide an inbound authentication (e.g. login credentials)	Optional
recover_seqnums	Handle recovery of sequence numbers	Optional
generate_logout	Create a logout message	Optional
generate_heartbeat	Create a heartbeat message	Optional
generate_resend_request	Create a resend request message	Optional
generate_sequence_reset	Create a sequence reset message	Optional
generate_test_request	Create a test request message	Optional
generate_reject	Create a reject message	Optional
generate_business_reject	Create a business reject message	Optional
activation_check	Check if a session or login schedule event has elapsed	Optional
process	Process an inbound message of any type	Optional

vi. Fix8 Session overridable methods

E. Summary of Features (Fix8Pro)

1. **High Availability (HA).** Fix8Pro offers a robust HA solution - known as Managed Resiliency, especially for sell-side users. Managed resiliency uses a cloud based persistence model. With a typical HA setup, a primary server offers connections to inbound clients. If this primary fails overs, clients will reconnect to a secondary (or subsequent) server and come on-line and resume with their previous client session contexts. ✪



vii. HA Architecure

2. **Integrated Middleware Support (IMS).** Our IMS feature provides a generic middleware interface seamlessly integrated into Fix8Pro. You can use your current enterprise middleware layer with Fix8Pro. Drivers for specific 3rd party middleware components are being developed. Fix8MT will develop additional drivers on demand. Currently 0MQ (zeromq) is supported. ✪

3. **Developer and Production Support.** Fix8 Market Technologies provides a subscription based support model for both developers and production systems.

4. **Access to All Bugfixes and Upgrades.** As part of the subscription product, all updates to Fix8Pro such as bug fixes and maintenance releases will be pushed out to subscribers.

5. **Pre-built Platform Specific Binaries.** These will built and supplied for your specific platform, ready for use in your environment.

6. **Scriptable Test Framework and Certification Tools.** A fully XML scriptable test framework provides a reliable and robust way to ensure quality and FIX conformance. Compose or capture business scenarios and create test cases. These scripts can be incorporated into a complete regression test suite. Both client and server modes are supported. Simple sell-side simulations can als be created to test client interfaces. This feature can also be used for FIX certification and compliance, especially for sell-side users.

   The following example XML script demonstrates a client testcase that sends a `NewOrderSingle` and expects `ExecutionReports` in reply. Specific fields and values of the replies are specified in the script. This script also demonstrates our embedded Python support for field testing.

✪Roadmap feature

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<fix8>
   <testcases name='clientset1' ns='FIX44' libdir='.libs' confdir='./'
test_spacing='2' message_spacing='2' message_timeout='5'>
      <session name='TESTA' role='initiator' config='testcase_client.xml' />


      <globvars>
         <var name='GLOBTEST01' value='%{OUT:TESTA:OrderQty:01}'/>
         <var name='PYTHONTEST01' value='
def test_field(a):                                    \n
   print &quot;Parameter was: {}&quot;.format(a)       \n
   if %{OUT:TESTA:Price:01} - a == 0:                 \n
      return True                                     \n
   else:                                              \n
      return False '/>
         <var name='HIS01' value='AUTOEXECPRIV' />
      </globvars>


      <testcase name='order_with_fills_01'>
         <message action='send' type='NewOrderSingle' session='TESTA'>
            <fields>
               <field name='Price' value='34.5' />
               <field name='OrderQty' value='100' />
               <field name='Symbol' value='BHP' />
               <field name='HandlInst' value='1' />
               <field name='OrdType' value='2' description='LIMIT'/>
               <field name='Side' value='1' description='BUY'/>
               <field name='TimeInForce' value='1' description='GOODTILLCANCEL'/>
               <field name='TransactTime' value='!'/>
               <field name='ClOrdID' assign='TESTA:ClOrdID:01'
                           value='testa_clordid_01' />
            </fields>
         </message>
         <message action='expect' type='ExecutionReport' session='TESTA'>
             <fields>
               <field name='OrderID' assign='TESTA:OrderID:02'/>
               <field name='ExecType' value='0' description='NEW'/>
               <field name='OrdStatus' template='OrdStatus-new'/>
               <field name='LeavesQty' value='100' operator='<='/>
               <field name='OrderQty' value='100' />
               <field name='CumQty' value='0' />
               <field name='AvgPx' value='34.5' />
            </fields>
         </message>
         <message action='expect' type='ExecutionReport' session='TESTA'>
            <fields>
               <field name='OrderID' value='%{TESTA:OrderID:02}'/>
               <field name='ExecType' >
                  <item result='pass' value='F' description='TRADE' />
               </field>
               <field name='OrdStatus'>
                  <item result='pass' value='2' description='FILLED' />
               </field>
               <field name='LeavesQty' value='0' />
               <field name='CumQty' value='100' />
               <field name='AvgPx' value='34.5' />
            </fields>
         </message>
      </testcase>
   </testcases>
</fix8>
```

viii. Sample client test cases

F. Requirements

	Supported or Minimum Required	**Recommended**
Hardware Architecture	Intel IA32 Intel x86-64 AMD64 Intel Itanium PowerPC ARMv7 ARMv8-A	**Intel x86-64** **AMD64**
Operating System	Linux 32/64 OSX Windows (32/64, VS2012, VS2013, VS2015, VS2017)	**Linux 64 bit** **Suggested distros…** **Centos** **Ubuntu** **Debian** **RHEL** **Scientific** **Fedora** **Suse**
Memory	Minimum 256mb	**+8gb**
Networking	Minimum 10BaseT SolarFlare 10G/40G	Gigabit Ethernet (GbE) 10GBase-X 40GBase-X
C++ Compiler	C++11 required	GCC 4.9 Clang 3.4 Intel C++ 14.0
3rd Party Libraries	Poco FastFlow TBB (Intel) Performance Tools (Google) Google Test (Google) Redis Memcached Aerospike MongoDB BerkeleyDB (Oracle) Doxygen OpenOnload	Poco TBB Performance Tools

G. Contact Us

Email

Fix8 Market Technologies Pty Ltd

Level 5, 155 Clarence Street, Sydney NSW 2000 Australia

ABN 29167027198

+61 2 8091 3811

[Fix8 Open Source]

[Fix8 Market Technologies]

[Fix8 Professional]

Version 1.4 20170620