XCC Speed Talk

Sam Neth, Lead Engineer, Server

Agenda

- Introduction to XCC
- Address common issues
- Current investigations
- Questions

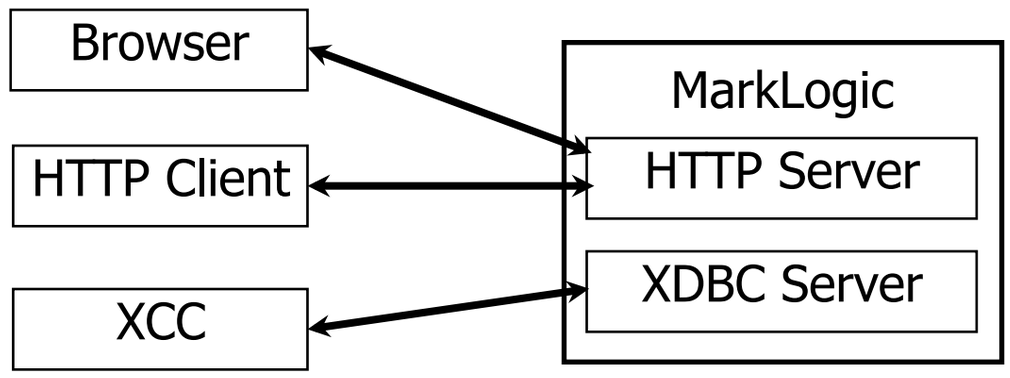
Disclaimer - Forward Looking Statements

- All statements describing future releases and capabilities, estimated release dates, and content are plans only, and MarkLogic is under no obligation to develop, include or make available, commercially or otherwise, any specific feature or functionality in any MarkLogic product.

- Information is provided for general understanding and informational purposes only, and is subject to change at the sole discretion of MarkLogic in response to changing customer requirements, market conditions, delivery schedules and other factors.

- Information should not be distributed without written permission from MarkLogic.

What is XCC?



- Client library for Java and .NET apps to talk to MarkLogic server
  - Roughly equivalent to a JDBC/ODBC driver
  - Proprietary XDBC protocol; talks to XDBC server

- .NET version is a thin C# layer
  - Java classes converted to CLR by IKVM (also used by Saxon.net)
  - IKVM upgrade in 4.2; much faster, with .NET 4.0 support

Why should I use XCC?

- Great for adding MarkLogic to existing Java or .NET architecture
  - Maps to traditional persistence and search modules
  - Language, Datatypes, Internal/External, Aggregation, Expertise, Control
- Some features you may care about
  - Manages connections and handles errors
  - Marshals data to/from scalar or DOM types
  - Streaming content insertion with System Entity Resolution
  - Transactional multi-document inserts
  - Java version supports SSL
- Future
  - Client-side transaction control in 5.0

Why might I not use XCC?

- Access to MarkLogic from platforms other than Java and .NET
- Direct SOA integration of MarkLogic
- Load management requirements
- The alternative; single-tier web services over HTTP
  - REST or SOAP services are common
  - JSON support in server
  - ml-json library used from JavaScript for conference app
  - Active Document library exposes MarkLogic to Ruby
  - There are other open source libraries out there

Load Balancing

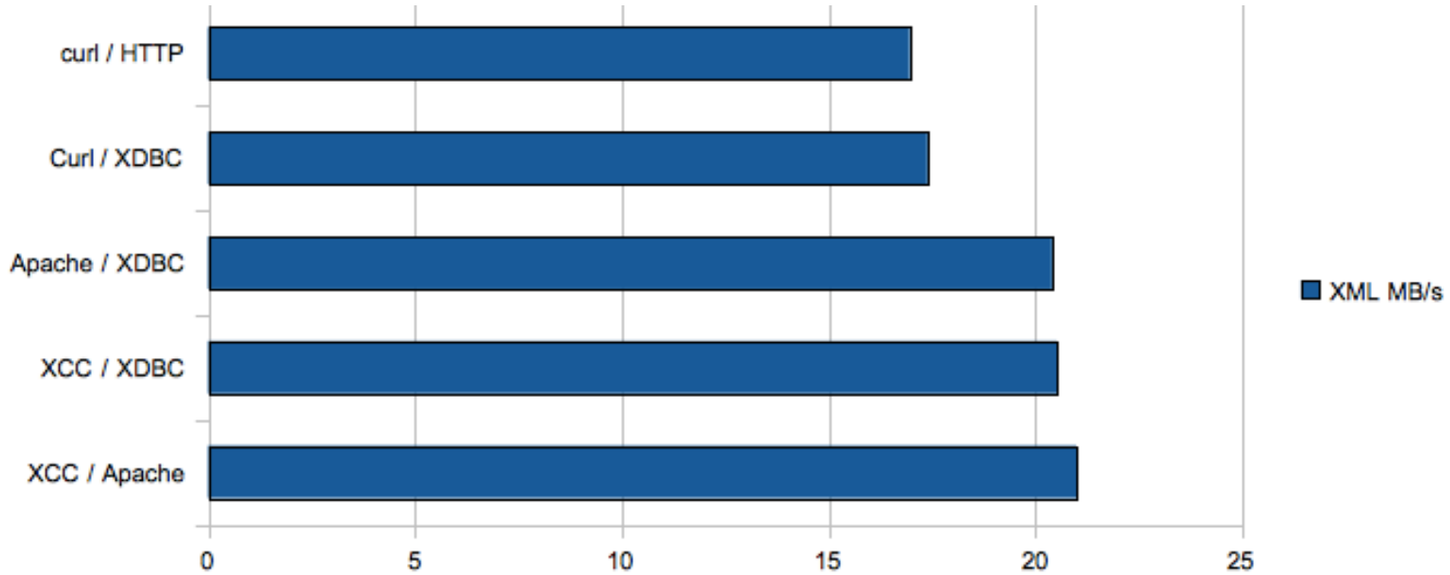
- Protocol is HTTP-like but not fully compliant
  - Callbacks for System Entity resolution
  - Connection management can result in isolation failure
  - No support for cookies

- If you need LBFO, do it in your Java/.NET application
  - External LBFO (F5, etc) is discouraged
  - Managing multiple ContentSource objects is a common solution

- Possible futures:
  - Full HTTP-compliance to support external LBFO
  - Built-in client LBFO management of multiple servers

Is XCC fast?

- Document retrieval speed comparison; local on slow Linux box



- Slower if parsed to DOM
  - Experimented with compressed trees
  - XML parsing *very* fast these days
- Looking at moving from NIO to single-threaded sockets.

Writing fast XCC applications

- Use the latest package
- Use streaming result sequences

```
request.setCacheResult(false)
```

  - Careful to read fully and close Session objects to avoid leaking sockets

Writing fast XCC applications

- Always reuse ContentSource, reuse Session when convenient.

```
// THIS

ContentSource cs = ContentSourceFactory.newContentSource(“xcc://localhost:8888”);
Session session = cs.newSession();

for (...) {
  AdhocQuery rq = session.newAdhocQuery(query);
  session.submitRequest(rq);
  ...
}

// NOT THIS !!

for () {
  ContentSource cs = ContentSourceFactory.newContentSource(“xcc://localhost:8888”);
  Session session = cs.newSession();
  AdhocQuery rq = session.newAdhocQuery(query);
  session.submitRequest(rq);
  ...
}
```

Writing fast XCC applications

- Reuse DocumentBuilder

```
// THIS

DocumentBuilder db = DocumentBuilderFactory.newInstance().newDocumentBuilder();
while (rs.hasNext()) {
  Element el = ((XdmElement)rs.next().getItem()).asW3c(db);
}

// NOT THIS

while (rs.hasNext()) {
  Element el = ((XdmElement)rs.next().getItem()).asW3c();
}
```

Compact Sequences

- Implementation detail!
- Before / After

```
(1, 2.0, 'three') =>

--f41610d7053f1eac
Content-Type: text/plain
X-Primitive: integer

1
--f41610d7053f1eac
Content-Type: text/plain
X-Primitive: float

2.0
--f41610d7053f1eac
Content-Type: text/xml
X-Primitive: string

three
--f41610d7053f1eac--
```

```
(1, 2.0, 'three') =>

--f41610d7053f1eac
Content-Type: application/vnd.marklogic.sequence

i1
f2.0
s5:three
--f41610d7053f1eac
```

Result Type Support

- In 4.0 (2008) some changes were made to align with XQuery 1.0
- Currently all nodes are returned as Element (even documents)
- Hadoop Connector uses XCC, drives some new requirements
- A few possibilities for 5.0:
  - Broader and differentiated support of node types (e.g. Attribute)
  - Geo types (Point, Polygon, etc.)
  - Maps (map:map to java.util.Map)
  - Query and other types as XML

Version Compatibility

- Objective is universal forwards and backwards compatibility
- Always use the latest from your major version if possible
- Testing focuses on matching versions; keep server up to date too!
- Maven repository can help you keep your XCC client up to date
  - http://developer.marklogic.com/maven2/

- XCC 4.2
  - Java 1.5+ / .NET runtime 3.0+
  - Support for .NET framework 4.0

- XCC 4.1
  - Java 1.5+ / .NET runtime 1.1+