An Introduction to XML

What it is, Why it's Used, How to Deal With It on Z

Joe Bostian
jbostian@us.ibm.com

What is XML?

eXtensible Markup Language is a means to:

- Describe data in a platform-neutral way
- Separate data content from its format
- Make data format available for others to discover and use

```
<?xml version="1.0">
<person id="Big Boss">
   . . .
</person>
```

XML and HTML are descendents of SGML

SGML is a descendent of GML

GML was developed at IBM in the 1960's

- A set of tags for the text formatter "SCRIPT"
- SCRIPT was the central component of DCF (Document Composition Facility)

A GML Example

```
.*------------------------ Title Page ---------------------
:gdoc sec='Optimal Parallel Merging and Sorting' gmlver=3.
:docprof ldrdots=yes toc=0123456 headnum=NO.
:frontm.
:titlep.
:title.Investigating an Optimal Parallel Algorithm for
:title.Merging and Sorting Using &sqrt.N Processors
:date.&date
:author.Joe Bostian
:address.
:aline.Parallel Algorithm Design
:aline.66.622
:aline.Spring 92
:aline.Prof. Kaltofen
:eaddress.
:etitlep.
.*-------------------------------------------
:toc.
.*--------------------------- Body ------------------------
:body.
.*-------------------------------------------
 . . .
```

- This is how I wrote reports in 1992 for a Master’s class
- Ran this through the SCRIPT processor to generate a printer-specific binary
- Syntactically, XML and HTML are not that different from this

Specific Design Goals for XML

These are the specific design goals for XML, as described by W3C:

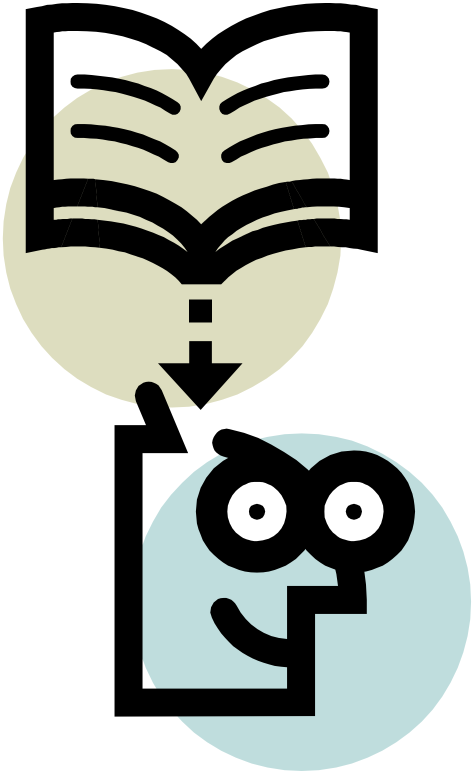
- XML shall be straightforwardly usable over the Internet.
- XML shall support a wide variety of applications.
- XML shall be compatible with SGML.
- It shall be easy to write programs which process XML documents.
- The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
- XML documents should be human-legible and reasonably clear.
- The XML design should be prepared quickly.
- The design of XML shall be formal and concise.
- XML documents shall be easy to create.
- Terseness in XML markup is of minimal importance.

See http://www.w3.org/TR/2006/REC-xml-20060816/ for more information

Why a Text Based Markup Language?

The key design goals of XML are what has made it extraordinarily successful

- Straightforwardly usable over the internet
  - Works with URI/URL standards for locating local and remote resources
- XML processing programs will be easy to write
  - (Sort of) There are lots of tools and packages to assist with XML creation, consumption, and translation
- XML documents will be easy to create
  - (Really easy) Any text based editor can be used on any platform in any language



Basic idea - enable communication through a clear description of a common foundational language

Related Developments Have Been Key

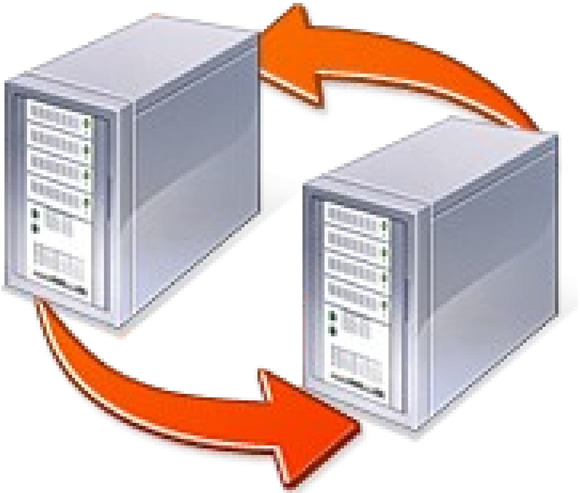
Description and publication of data forms

- Document Type Definitions (DTDs)
  - Part of the XML spec
  - The original means for describing valid XML docs
- Schemas
  - A more powerful, comprehensive specification (www.w3.org/XML/Schema)
  - The preferred method for describing valid XML docs

The ability to transform XML to other arbitrary forms

- Stylesheets – outline how to generate an arbitrary document from an XML document

These allow users to make what they need of the information they have



How Widespread is XML Usage?

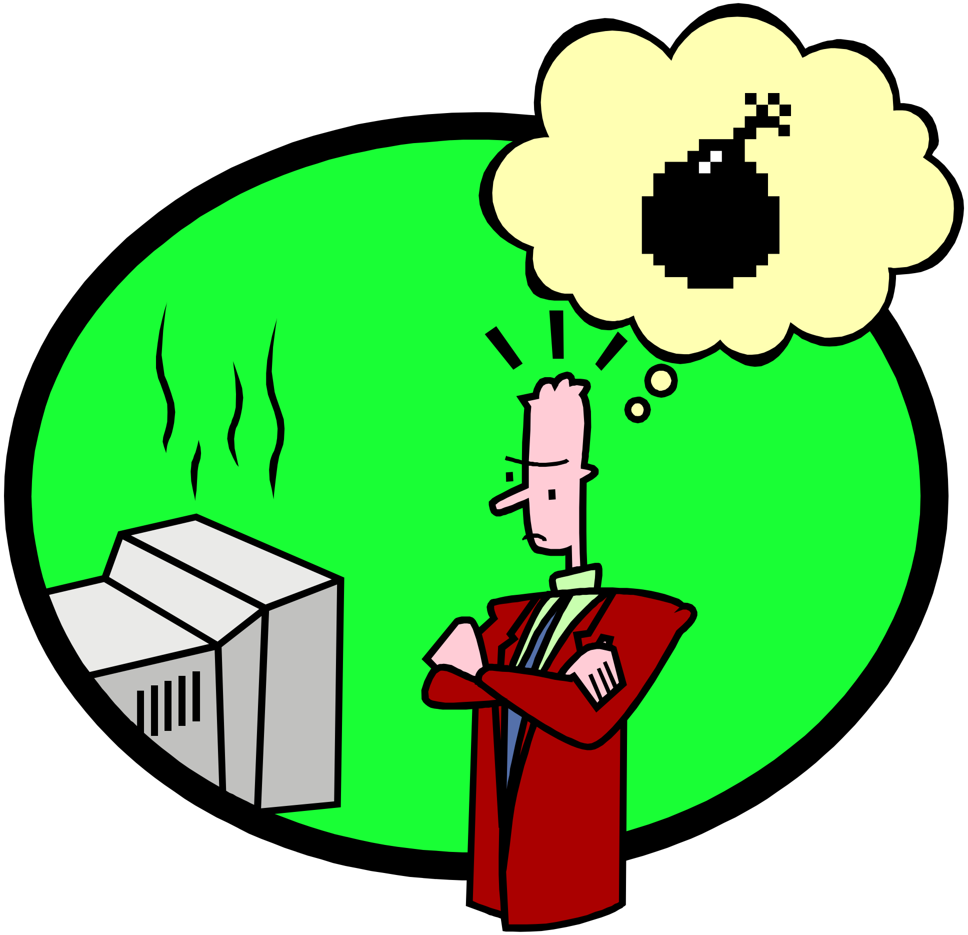
- Dozens of standards defined to describe how web based services interoperate

  (www.oasis-open.org/home/index.php)
- XML is often used as the internal form for documents
  - (xml.openoffice.org/)
  - (office.microsoft.com/en-us/products/HA102058151033.aspx)
- XML usage/management features are now key for most databases
  - (www-306.ibm.com/software/data/db2/xml/)
  - (www.oracle.com/technology/tech/xml/xmldb/index.html)
  - (www.microsoft.com/sql/prodinfo/overview/whats-new-in-sqlserver2005.mspx)
- XML is used for many/most web pages

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

- Many, many, **many** more standards defined by specific industries
  - Banking, Automotive, Insurance, etc.

So, XML is all Good, Right?



Remember that

“Terseness in XML markup is of minimal importance”

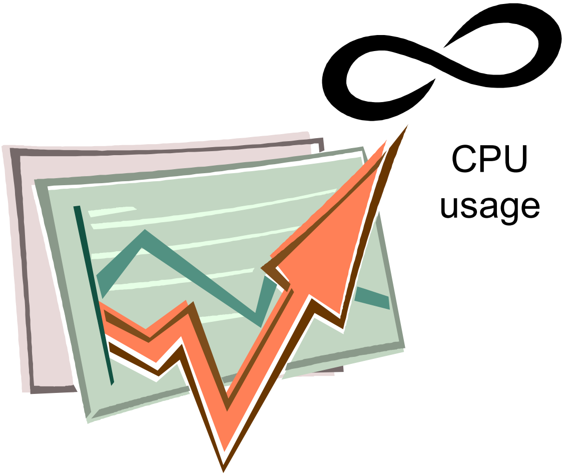
?

It costs a lot of CPU cycles to handle text data

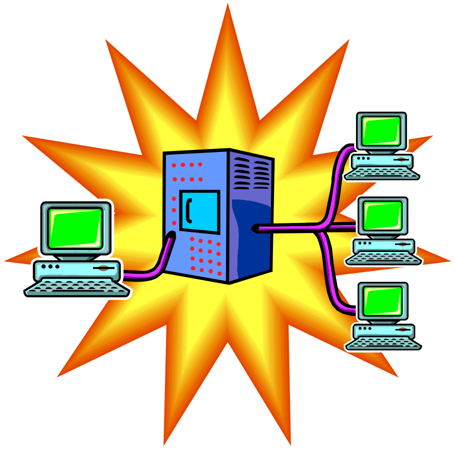
- Often requires conversions between encodings
- Data has to be parsed before it can be used

If Money and Time Were No Object …

- Before any data is read, it would first be parsed
- When any data is written, it would first be serialized (converted to XML form)
- Everyone would be happy to throw more servers on the pile forever



Since Money and Time Are Huge Objects …



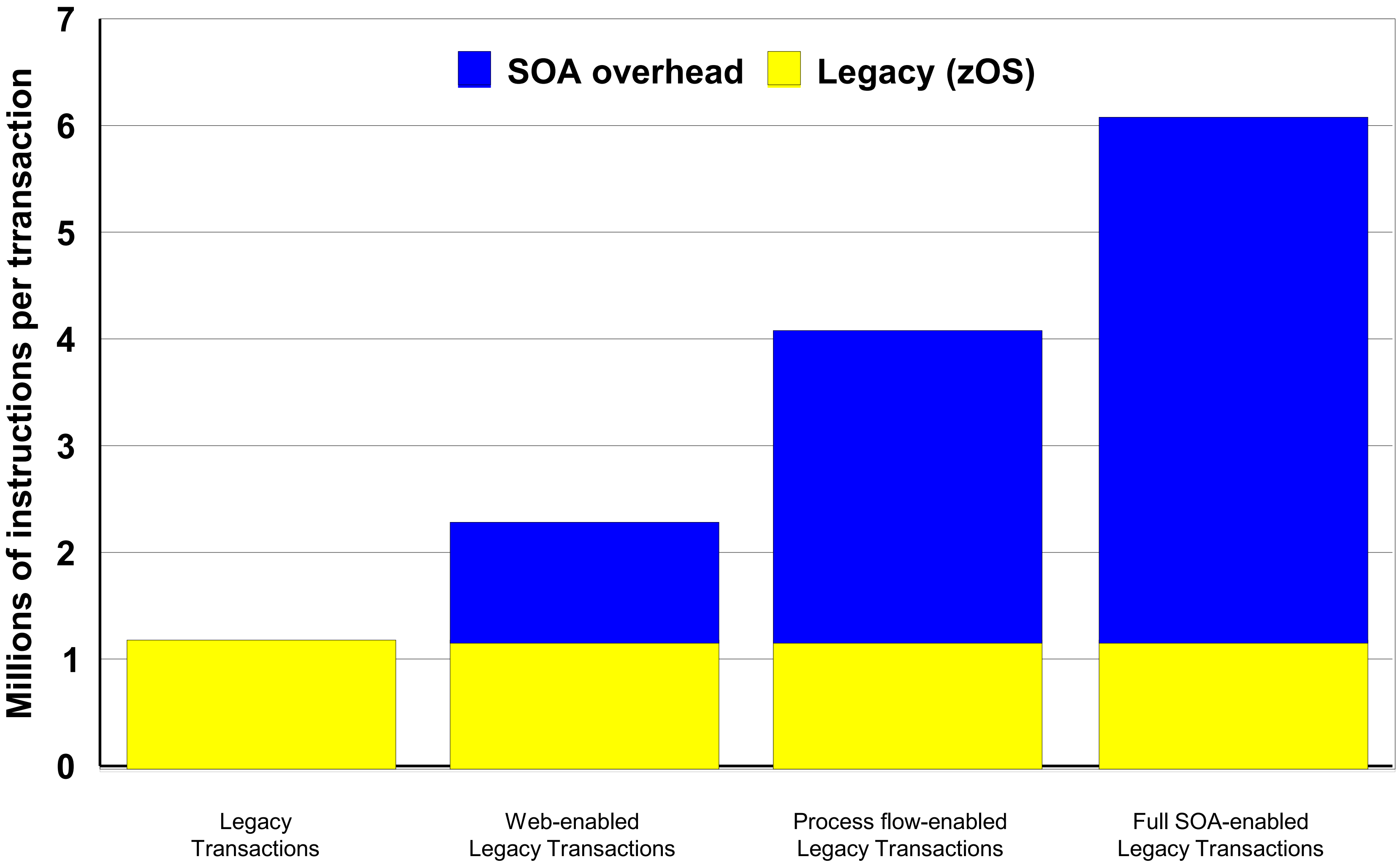
We have to find a way to process XML at a lower cost

- Use cheaper cycles
- Minimize cycle usage to process XML

Especially important on Z

XML and SOA Cycle Consumption

Example: Price Quote / Price Change transaction in various environments



The Cost Problem on z/OS

XML overhead threatens to make the total cost of ownership unworkable

- Legacy transactions become too costly
- New applications are never developed for the platform

Goal - minimize the total cost of XML processing on the platform

- Components of the z/OS product
- Middleware running on z/OS
- Applications

Approach – create best-of-breed XML processors, and make them available everywhere on the platform

- Additionally, reduces install and maintenance overhead

Current Parsing Solutions

XML Toolkit for z/OS

- SAX – Simple API for XML
  - Generates a stream of events that are returned to the caller as the document is parsed
- DOM – Document Object Model
  - Builds a structured tree of nodes representing the document
  - Caller traverses the tree for needed information

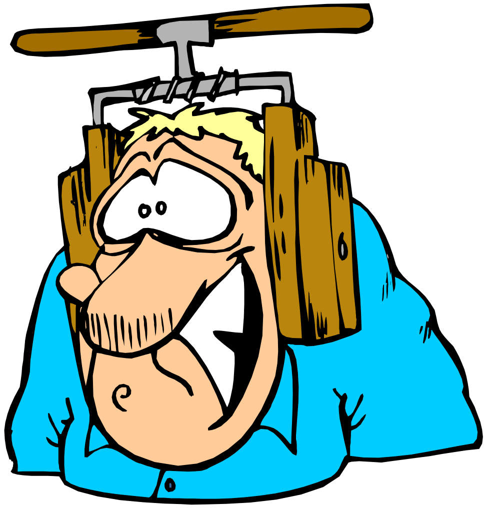
C/C++ interfaces are installed via the z/OS XML Toolkit, while the Java interfaces have been integrated with Java 4 (and later)

Both packages also include the XSLT stylesheet processor

Current Parsing Solutions …

- Enterprise COBOL and PL/I - XML Parse statement
  - Provides access to an integrated XML parser
  - High-performance
  - Primitive functionality
  - Unique interface

- Home-grown parsing solutions
  - Not that hard to do, but are hard to do right



The z/OS XML System Services Solution

A high performance parsing solution integrated with the BCP

- Non-validating
- Handles very large documents
- Small memory footprint
- Works in task mode and other esoteric MVS environments
  - SRB, cross-memory mode, etc.
- Unique buffer-in/buffer-out interface
  - Allows documents to be parsed in segments
- C/C++ and assembler interfaces

Avoids shortcuts in favor of performance

- XML 1.0 and 1.1 documents supported
- Robust well-formedness checking
- Entity support (internal DTDs)
  - Complete DTD grammar is tolerated
  - Returns unresolved entity references in the parsed data stream
- Full namespace support

z/OS XML System Services …

Provides additional useful features

- Comment stripping
- Tokenize significant whitespace (instead of returning character data)

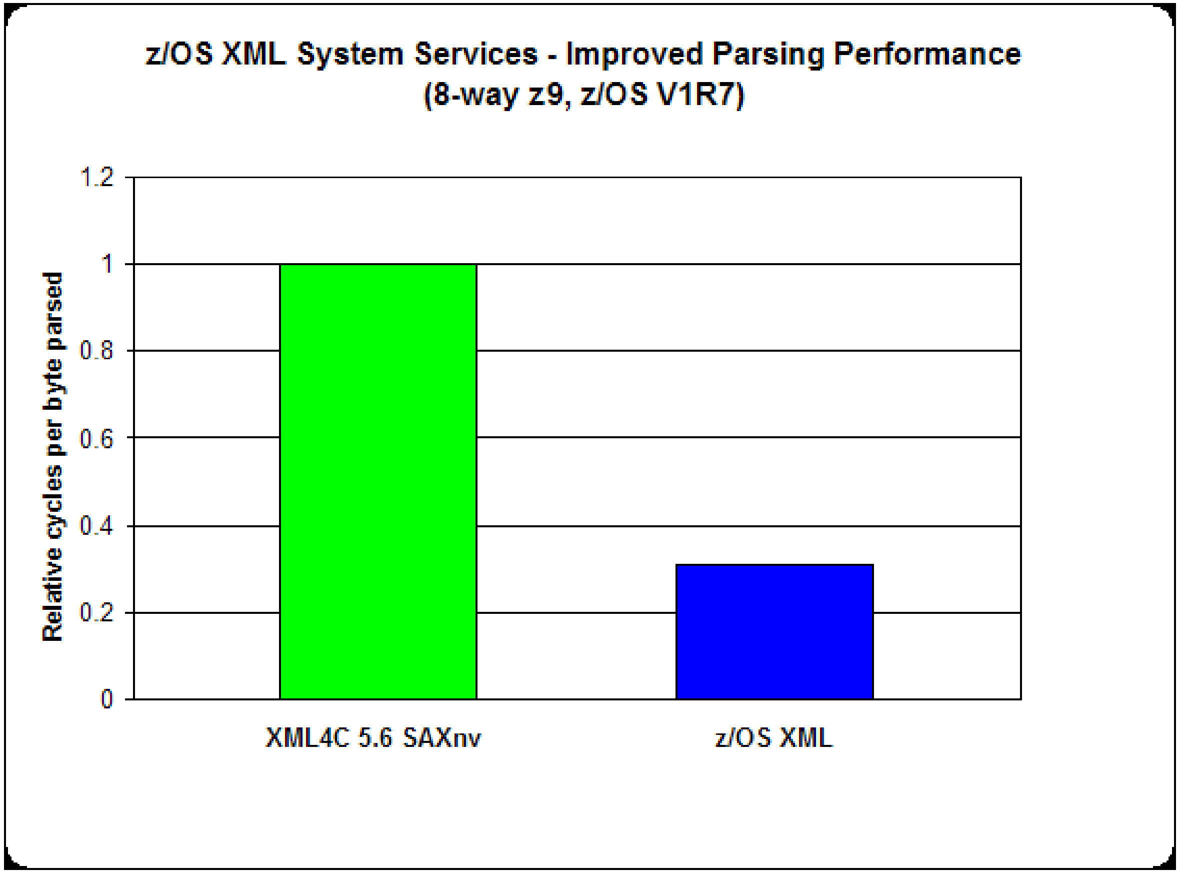
Processing is offloaded to specialty engines

- zSeries Application Assist Processor (zAAP)
    - First implemented in z/OS V1R9, rolls back to V1R8 and V1R7
- z9 Integrated Information Processor (zIIP)

Working to implement additional features now

- Validation by schema during parse
- Extend the toolkit to enable use of XML System Services under the covers
    - Provides offload for SAX and DOM users

Performance



Considerations:

- SAX and z/OS XML use fundamentally different processing models
- SAX-based apps require significant change to use z/OS XML

See www-03.ibm.com/servers/eserver/zseries/zos/xml/perform/

Performance …

No need for callers to transcode strings returned in the parsed data stream

- z/OS XML Toolkit transcodes and returns everything in UTF-16

Have parsed 500 MB documents in the lab

- Buffer spanning allows a caller to trade time for space
  - Large buffers allow the parse to move faster
  - Small buffers reduce overall memory footprint
- Can theoretically parse documents of unbounded size

Summary

XML has achieved the acceptance that was intended for it

- It is fairly easy to use and develop
- It enables easier communication between endpoints in a web services environment
- It simplifies conversion of data to other forms

It's easy to burn a lot of cycles processing XML

- Everything must be parsed before it can be used

Reducing the cost of XML processing is a high priority for z/OS

- Cost-performance is enhanced through the use of specialty engines
- We continue to focus on reducing the number of cycles required for parsing
- Always looking to increase the number of environments where XML System Services is used

Reference

See our web page:

www-03.ibm.com/servers/eserver/zseries/zos/xml/

Your source for:

- The z/OS XML System Services User's Guide
- Links to other parser related information