Center for Technology in Government

Rethinking the Database

Using XML databases to align with business processes and enhance capabilities

GTC EAST

The New York Digital Government Summit

September 21-24, 2009

University at Albany

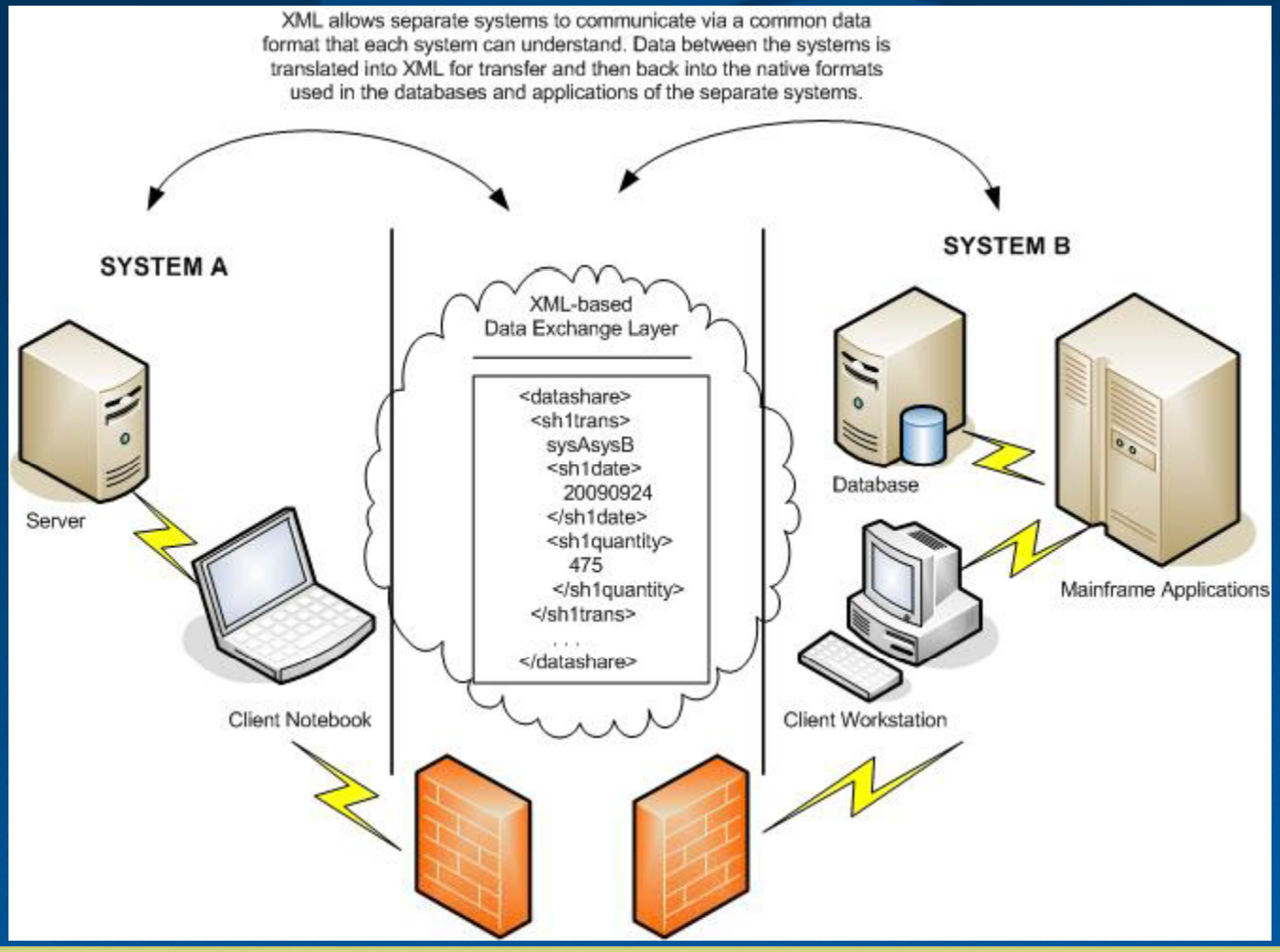
Components of XML

- **XML** - encoding documents electronically
- **XSL** - transforming and rendering XML documents
- **XPath** - addressing the parts of an XML document
- **XML Database** – storing collections of XML data
- **XQuery** - querying collections of XML data
- **SQL/XML** - querying XML within SQL
- **XForms** - interfacing with XML data
- **XRX** - coming soon to a database near you

XML for Data Exchange

- **Data exchange -** XML enable platform-independent data exchange among applications.
- XML serves to "glue together" or mediate a common data layer between two separate and already existing programs.
- Typically, XML-based messaging services (such as SOAP) enable different applications to communicate.

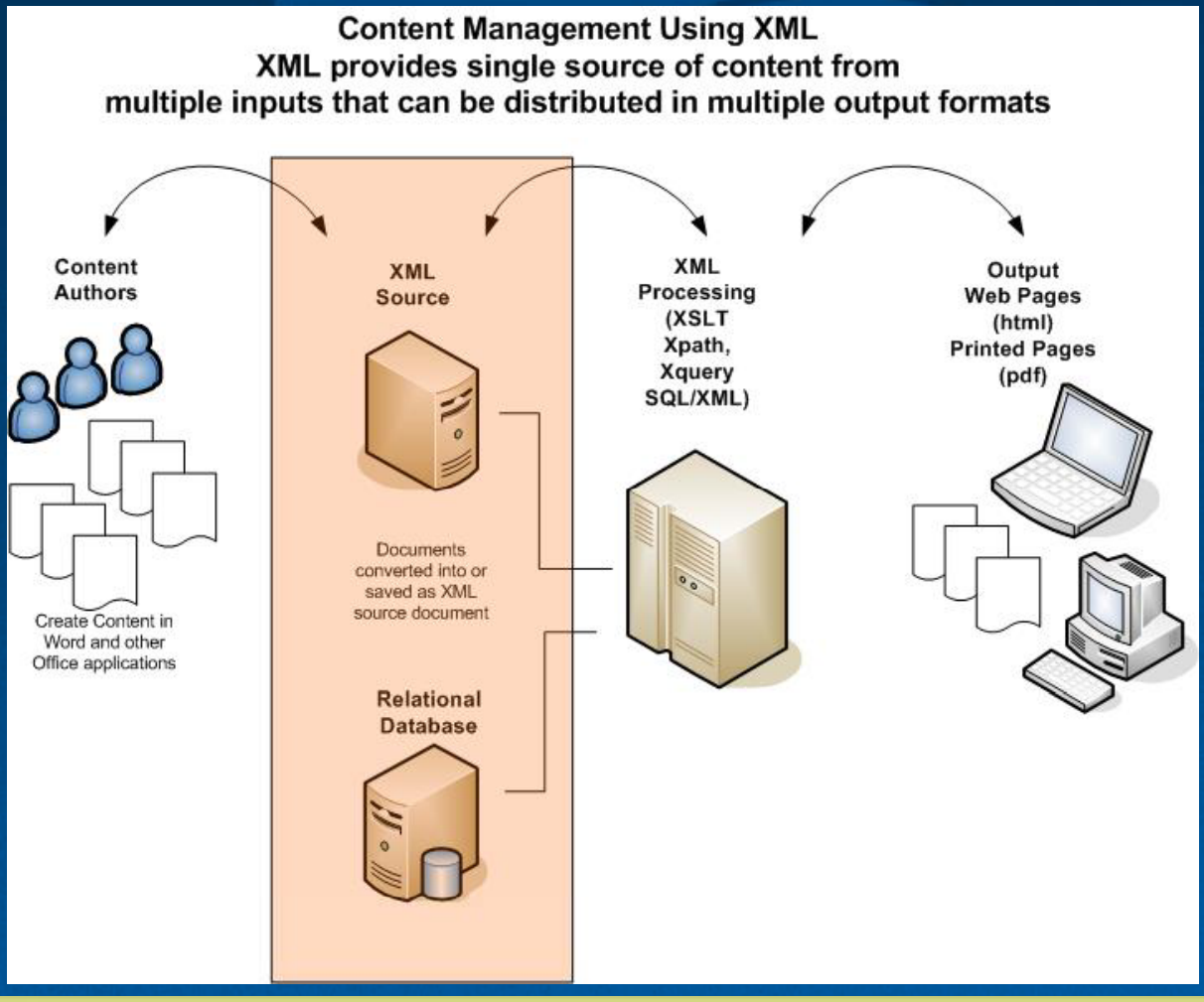
XML for Data Exchange



XML for Content Management

- **Web content management -** usually implemented as a Web application, for creating and managing HTML content.
- Most systems use a database to store content. Content is frequently, but not universally, stored as XML, to facilitate reuse and enable flexible presentation options.
- A presentation layer displays the content to regular Web-site visitors based on a set of templates. The templates are sometimes XSL files.

XML for Content Management



XML for Syndication: RSS

- **RSS** is a simple XML format used to syndicate headlines.
- It is used by websites that publish new content regularly and provides a list of headlines with links to their latest content.
- Content such as news feeds, events listings, project updates, and most recently podcasting, video and image distribution can all be distributed by RSS.

RSS Reader

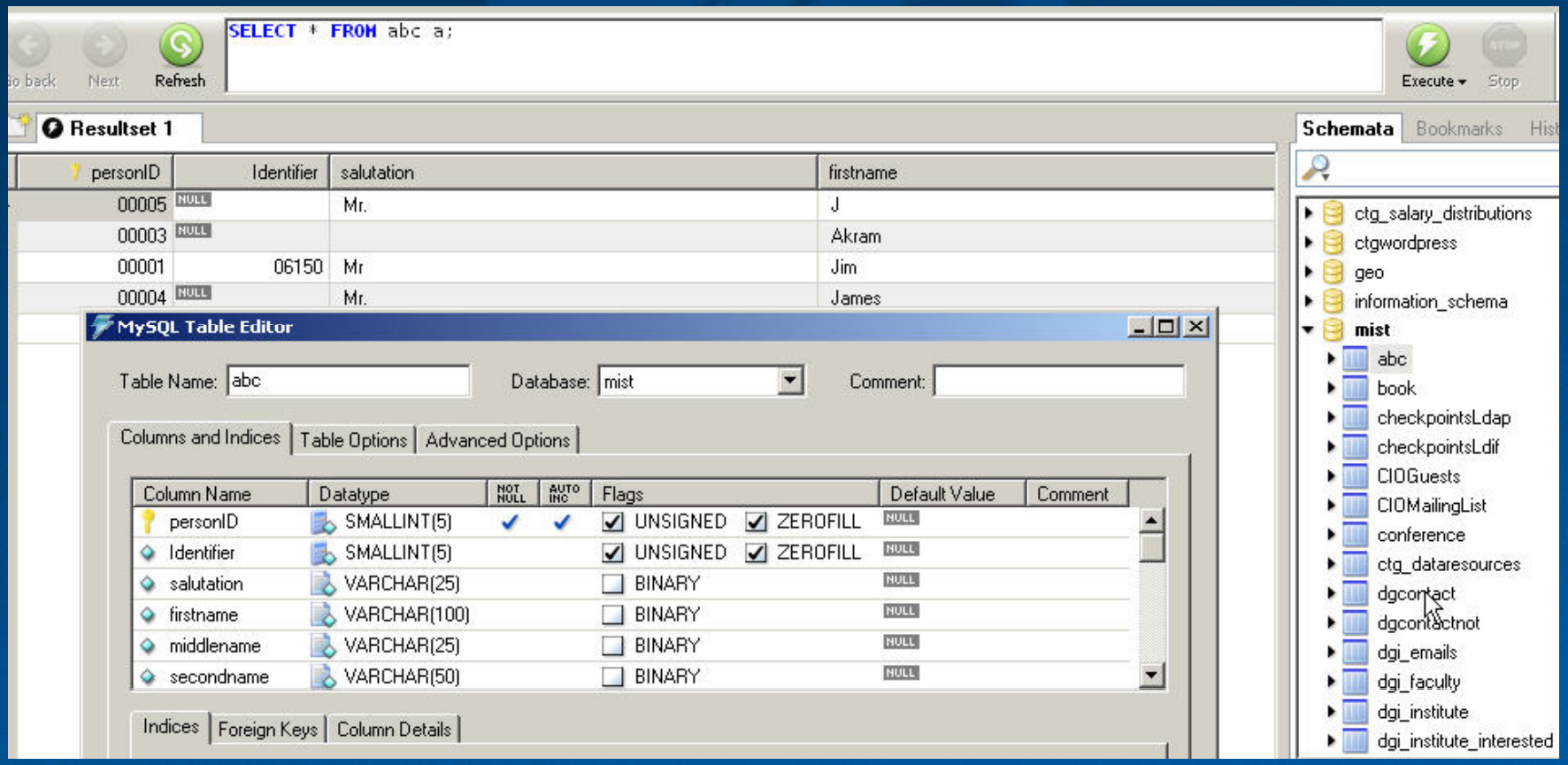

Subscribe to this feed using
Sage
Always use Sage to subscribe to feeds.
Subscribe Now
CTG All-in-One Page
A look at the most recent CTG updates in various categories
Center for
Technology in Government
Home Page
Tuesday, August 04, 2009 4:15 PM
LATEST NEWS : Rick Howard, Oregon CIO for Human Services Department, authors opinion article linking Health IT to CTG's interoperability framework ♦ 2008 ANNUAL REPORT: Government Worth Having ♦ TECHNOLOGY TRANSFER: Advanced Web Technologies ♦ POPULAR DOWNLOADS: Advancing Return on Investment Analysis for Government IT: A Public Value Framework ♦ SPOTLIGHT ON: University at Albany-SUNY Appoints Theresa Pardo Director of CTG ♦ PROJECT HIGHLIGHT: Highlights from the 2009 iGov Research Institute in Seattle ♦
Most Recent Web News
Tuesday, August 04, 2009 2:17 PM
2008 ANNUAL REPORT:"Government Worth Having";PROJECT HIGHLIGHT:"Highlights from the 2009 iGov Research Institute in Seattle";VERIFY/UPDATE YOUR CONTACT INFORMATION:"Within the next week, we will be sending out a request via this CTG Web News distribution list asking you to verify and update your contact information using a form on our CTG Web site.";
Most Recent Publication
Monday, August 31, 2009 1:00 AM
International Digital Government Research: A Reconnaissance Study (1994 - 2008) - UPDATED -- Today, digital government (DG) research is being conducted all over the world. Most of this work is focused within the geographic and political contexts of individual countries. However, given the growing influence of global economic, social, technical, and political forces, the questions embedded in digital government research are now expanding to international dimensions. A

Databases: Relational vs. XML

- Relational – no hierarchy or significant order; based on two-dimensional tables. Used for storing and querying data.
- XML – hierarchical and sequential; based on trees in which order matters. Used for exchanging and displaying data.

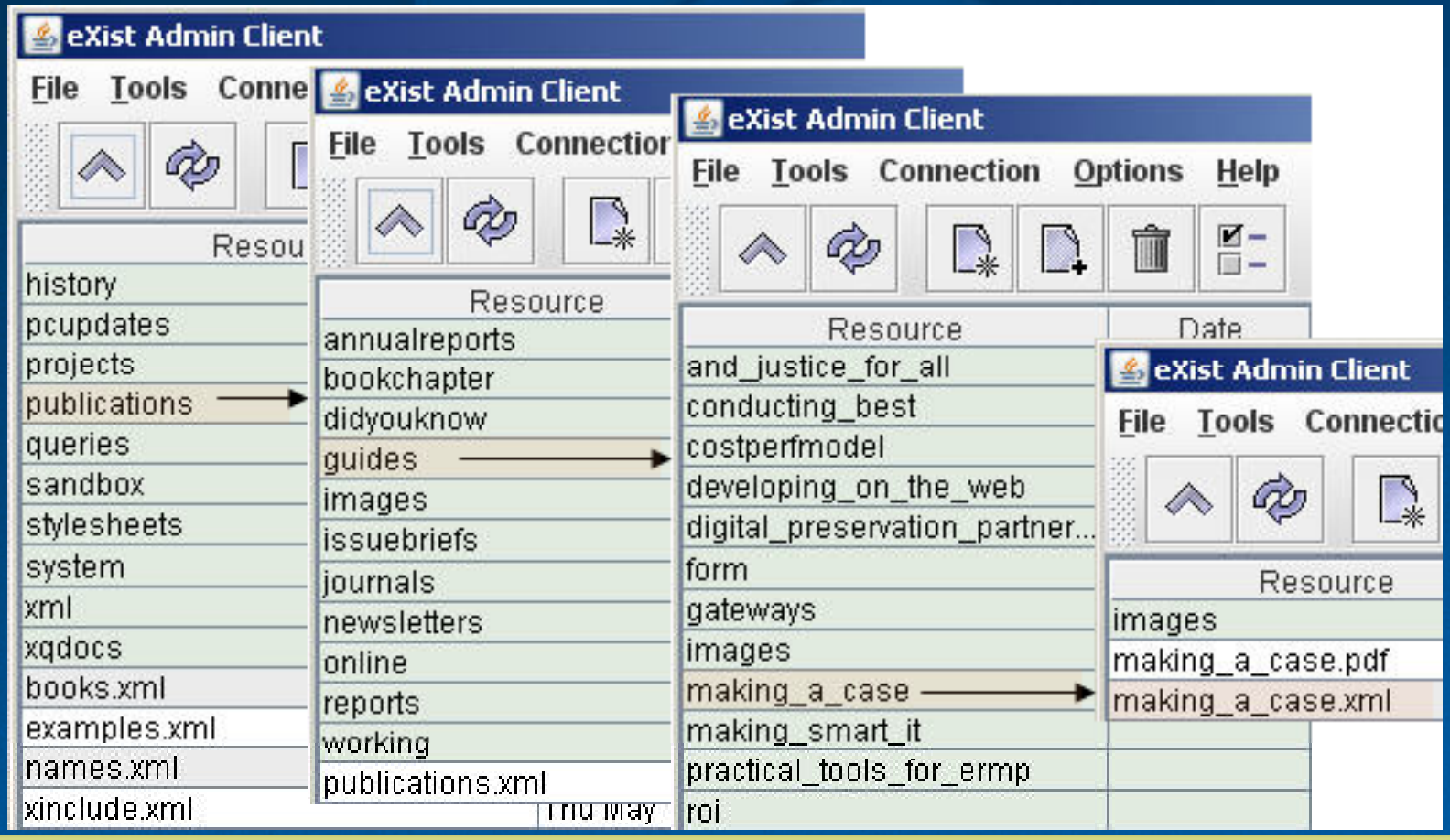
Relational Data

(tables, columns, rows, keys)



XML Data

(collections, files)



A Complicated Relationship

Data exchange between XML, applications, and databases is not a simple, one-step event.

It involves many processing steps and translations of the data into totally different formats.

```
<SalesOrder>
   <Number>1234</Number>
   <Customer>Gallagher Industries</Customer>
   <Date>29.10.00</Date>
   <Line Number="1">
      <Part>A-10</Part>
      <Quantity>12</Quantity>
      <Price>10.95</Price>
   </Line>
   <Line Number="2">
      <Part>B-43</Part>
      <Quantity>600</Quantity>
      <Price>3.99</Price>
   </Line>
</SalesOrder>
```

can be mapped to these objects:

```
   object SalesOrder {
      number = 1234;
      customer = "Gallagher Industries";
      date = 29.10.00;
      lines = {ptrs to Line objects};
   }            /   \
               /     \
              /       \
object Line {          object Line {
   number = 1;             number = 2;
   part = "A-10";          part = "B-43";
   quantity = 12;          quantity = 600;
   price = 10.95;          price = 3.95;
}                      }
```

and then to rows in these tables, which are linked through a primary key / foreign key relationship:

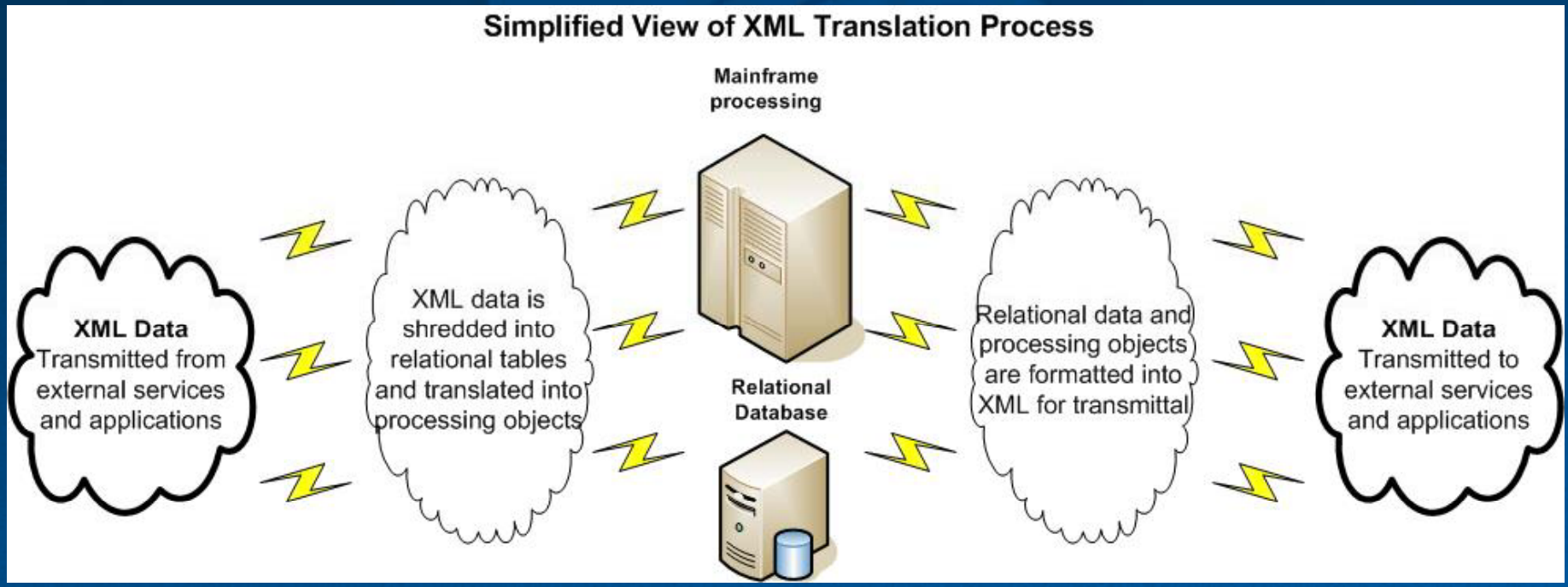
```
SaleOrders
----------
Number   Customer                Date
------   --------------------    --------
1234     Gallagher Industries    29.10.00
...      ...                     ...
...      ...                     ...


Lines
-----
SONumber   Line   Part   Quantity   Price
--------   ----   ----   --------   -----
1234       1      A-10   12         10.95
1234       2      B-43   600        3.99
...        ...    ...    ...        ...
```

Data arrives in an XML format.

XML document is mapped to processing objects.

And also mapped to rows and tables in relational database.

A Complicated Relationship



Simplifying the Relationship

- Storing data as XML (in native XML or XML-enabled databases) eliminates the process of translating data back-and-forth into various formats.
- Data is received, stored, and processed as XML.
- Eliminates multiple translation steps (along with their development times and their possibilities for errors).

The Real Difference

"XML is by definition self-describing data. Build the database around that structure not the other way around. The implementation is far from being that simplistic. This basic concept however – leverage XML's self-describing and hierarchical nature to manage it – is the very foundation of an XML database."

(from http://bigmenoncontent.com/2009/05/28/xdb-matters/)

Two Immediate Benefits

By storing XML data as XML in XML database:

- Simplified storage
- New query capabilities

New Query Capability

Query

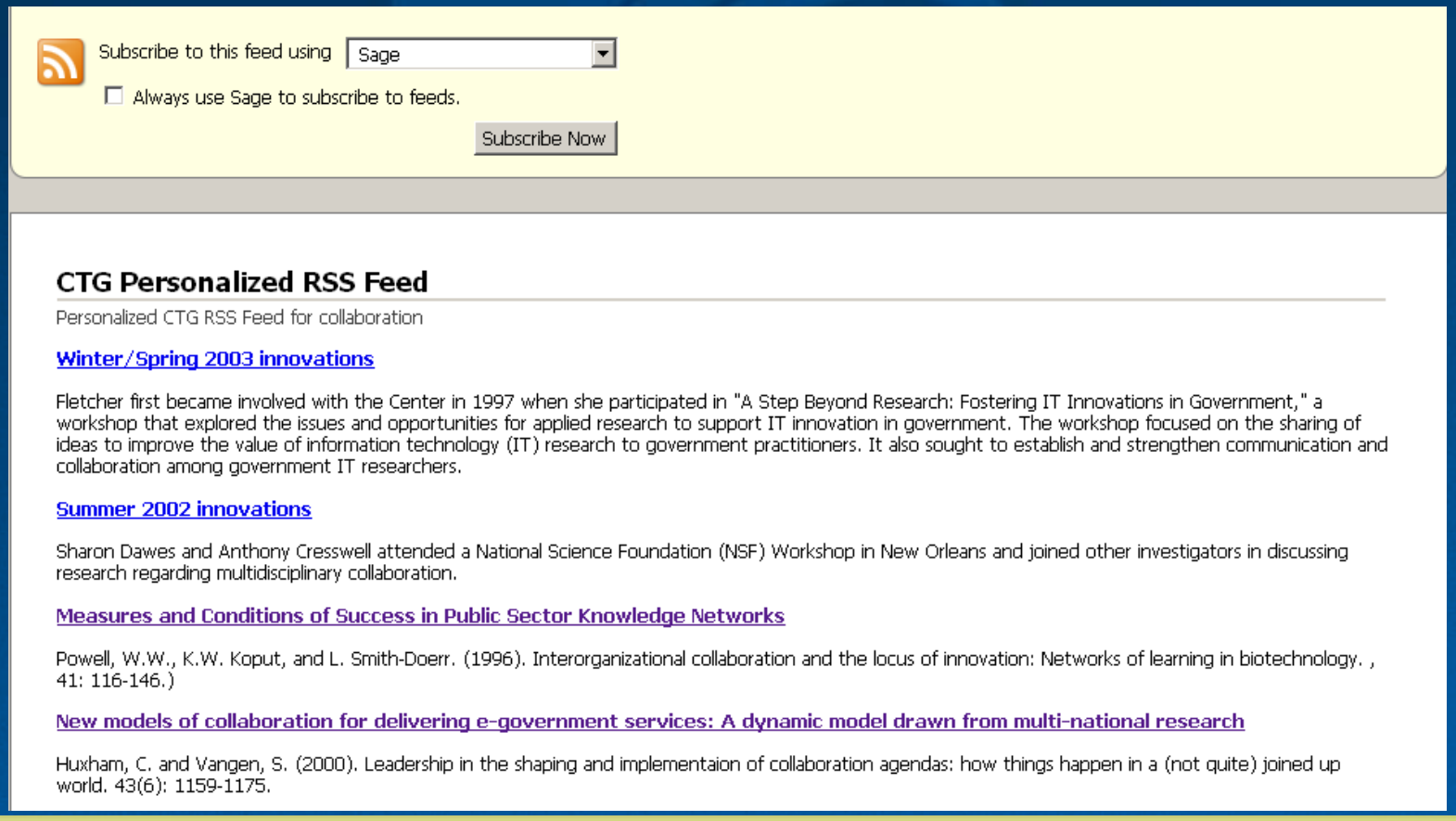
Create a personalized RSS feed based on keyword(s) that an individual wants to track within CTG's Website.

How?
Since all content on website is stored in XML database, it is all available to query. An XForms interface enables visitors to submit their own terms of interest for building a personalized RSS.

```
xquery version "1.0";
declare namespace request="http://exist-db.org/xquery/request";
let $param1 := request:get-parameter("param1","collaboration")
(:Finds all pubs in 'publications' collection that contain the word 'collaboration' to create RSS feed:)
let $col := "publications/"
return
<rss version="1.0">
<channel>
<title>CTG Personalized RSS Feed</title>
<link>http://www.ctg.albany.edu</link>
<description>Personalized CTG RSS Feed for {$param1}</description>
<ttl>240</ttl>
{
let $z :=collection($col)/book//chapter[contains(para, $param1)]
let $title := $z/ancestor::book
for $book1 in distinct-values($title/title)
let $l := $title[title = $book1]//parent::book/@label
let $i := $title[title = $book1]//parent::book/@id
let $d := $title[title = $book1]/bookinfo/pubdate/text()
let $f := $title[title = $book1]//chapter//para[contains(text(), $param1)][fn:last()]/text()

return
<item>
<title>{$book1}</title>
<link>{ concat("http://www.ctg.albany.edu/publications/",$l,"/",$i)}</link>
<description>{$f}</description>
<pubdate>{$d}</pubdate>
</item>
}
</channel>
</rss>
```

New Query Capability

Results of Query



New Query Capability

Query

Create a list of CTG publications grouped by main author.

How?
Since all content on website is stored in XML database, individual documents can be queried at the node level and combined, grouped and sorted into results that are difficult if not impossible to achieve without XQuery.

```
xquery version "1.0";
declare namespace request="http://exist-db.org/xquery/request";
(:Finds all pubs in 'apublications' collection for each individual Author and lists their publications with author in author order :)
let $param2 := "guides"
let $col := "publications/"
let $col2 := concat($col,$param2)
return
<set>
 {
   let $a := collection($col)/book
   for $name in distinct-values($a//author)
order by $name
   return
      <book>
      <bookinfo>
         <author>
           { $name }
         </author>
         {
            for $b in collection($col)/book
            where some $ba in $b//author
                satisfies ($ba = $name)
            return
            <title label=" { concat("http://www.ctg.albany.edu/publications/",$b/@label,"/",$b/@id)} ">{$b/title/text()} </title>
         }
      </bookinfo>
      </book>
 }
</set>
```

New Query Capability

Results of Query

Resources

- **XML Databases - The Business Case**
  http://www.cfoster.net/articles/xmldb-business-case/
- **Ronald Bourret, Consulting, writing, and research in XML and databases**
  http://www.rpbourret.com/xml/
- **Introduction to Native XML Databases**
  http://www.xml.com/pub/a/2001/10/31/nativexmldb.html
- **A comparison of XML-enabled and native XML data management techniques**
  http://xml.sys-con.com/node/104980?page=0,0
- **Feature Comparison: EMC Documentum xDB vs. Oracle XML DB & IBM DB2 pureXML**
  https://community.emc.com/docs/DOC-2999

Contact

- **Jim Costello**
  Center for Technology in Government
  jcostello@ctg.albany.edu
  518-442-3812

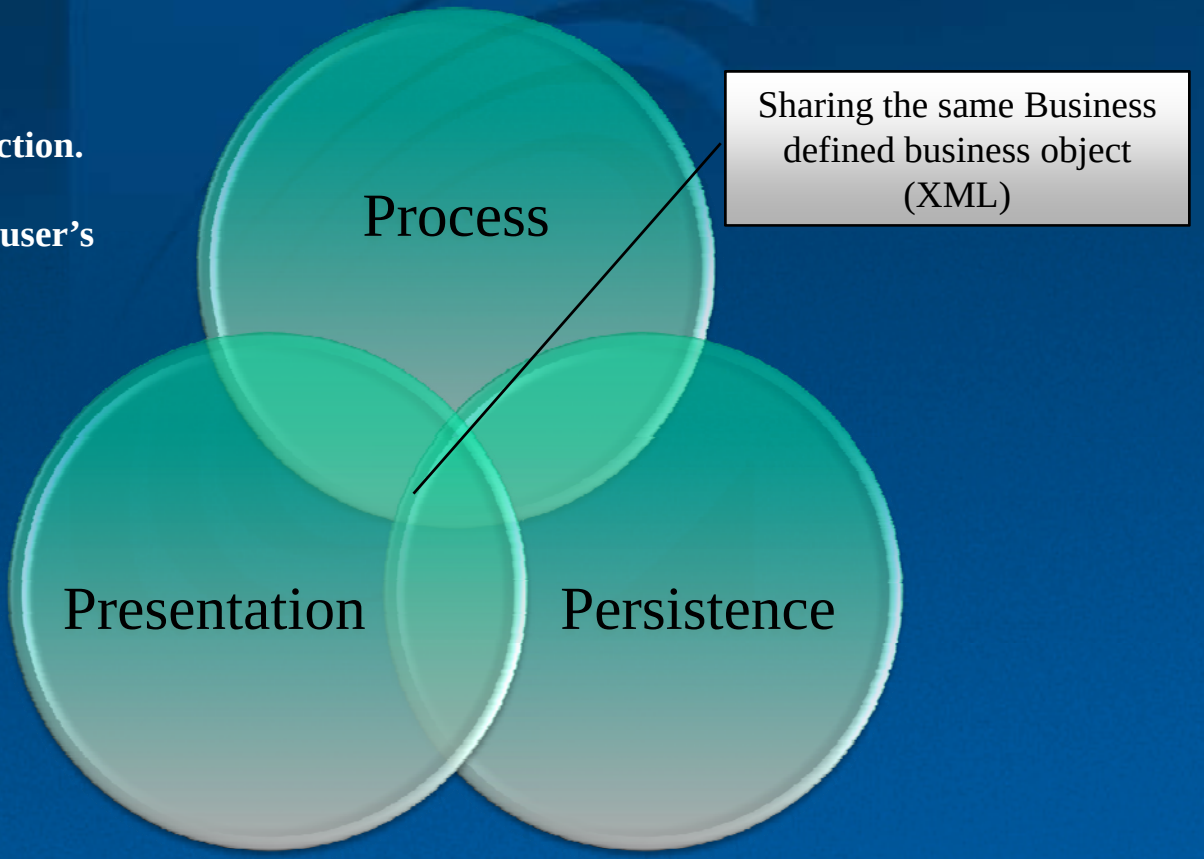
Real World Large Scale Deployment

- **New York State Taxation and Finance**

Facilitating Business Alignment

Eliminates Technical Abstraction. All processes, functions and reports "configured" in user's terms.

The same core Business Object Is leveraged by all of the system components .



Returns Processing

Operational XML – Next Generation DB Design

IT-201

W2

IT-150

col1	col2	col3	col4	col5	…	col1000
134	NULL	11/23/05	NULL	NULL	…	NULL
NULL	276	NULL	NULL	Yes	…	NULL
12	NULL	NULL	99.99	NULL	…	NULL
NULL	NULL	NULL	123.23	NULL	…	No

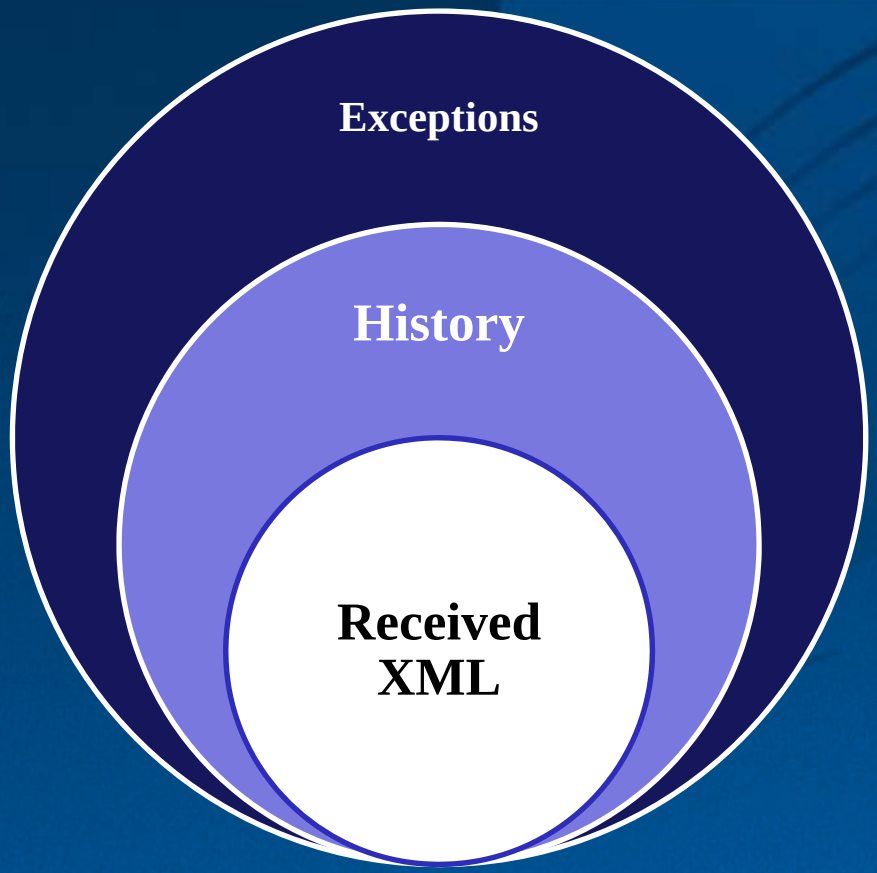
```
<filing>
  <form formid = 'IT201'>
    <wages>134</wages>
    <date>11/23/05</date>
  </form>
  <form formid = 'W2'>
    <wages>278</wages>
    <jointTP>Yes</jointTP>
  </form>
</filing>
```

Relational Table by Form
- 3600 tables required
- Difficult to get filing context
- Made Rules engine, display difficult
- Much IO

Generalized Relational Table
- Needed DB to translate fields
- Sparsely populated
- Performance issues
- Rules engine limitatations

XML Solution
- Business object based (the audit folder)
- Keeps business context
- Robust rules processing
- Can leverage XML tooling

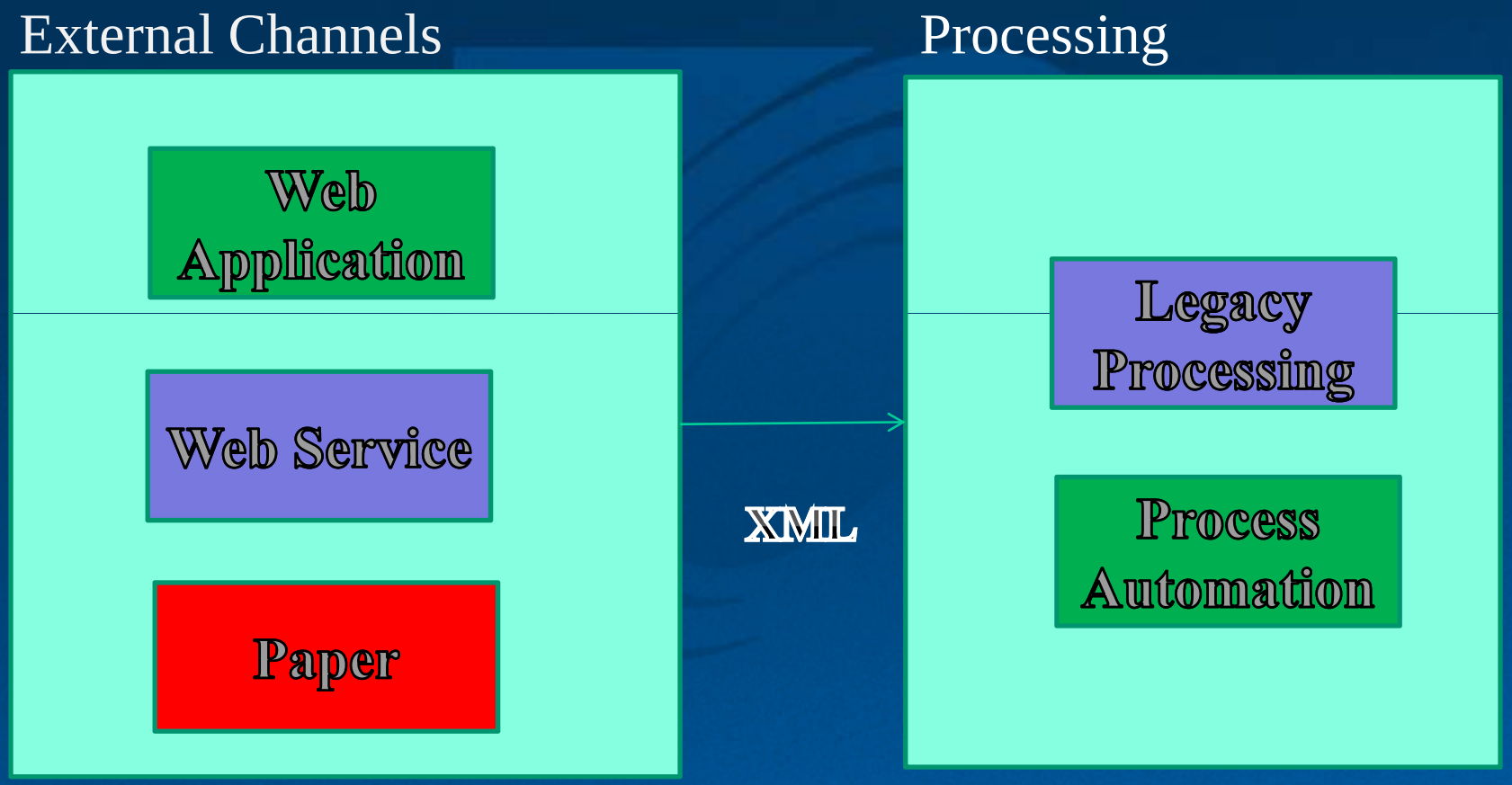
Transactional XML Layering



- Exceptions
  - Quality or Condition of Data
  - Auto Routing
  - Customer Service
- History
  - What, Why and Who of all Changes
  - Auditability
  - Exception Resolution
- All relevant data in one place
  - Less I/O
  - Data Integrity
  - Enable Transaction Processing

Development Enablement

Establishing Patterns



Web Application

Web Development challenges

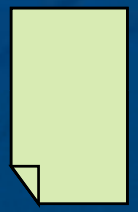
- **Develop quicker**
- **Reuse “segments” of web apps**
- **Consistent features (print, return to application)**
- **Consistently defined navigation patterns**
- **Track users usage of application**

What is a Web Application?

Web Application

A series of form based UI objects that create a transaction for processing.

Then each of these “forms” can be designed exactly the same way….



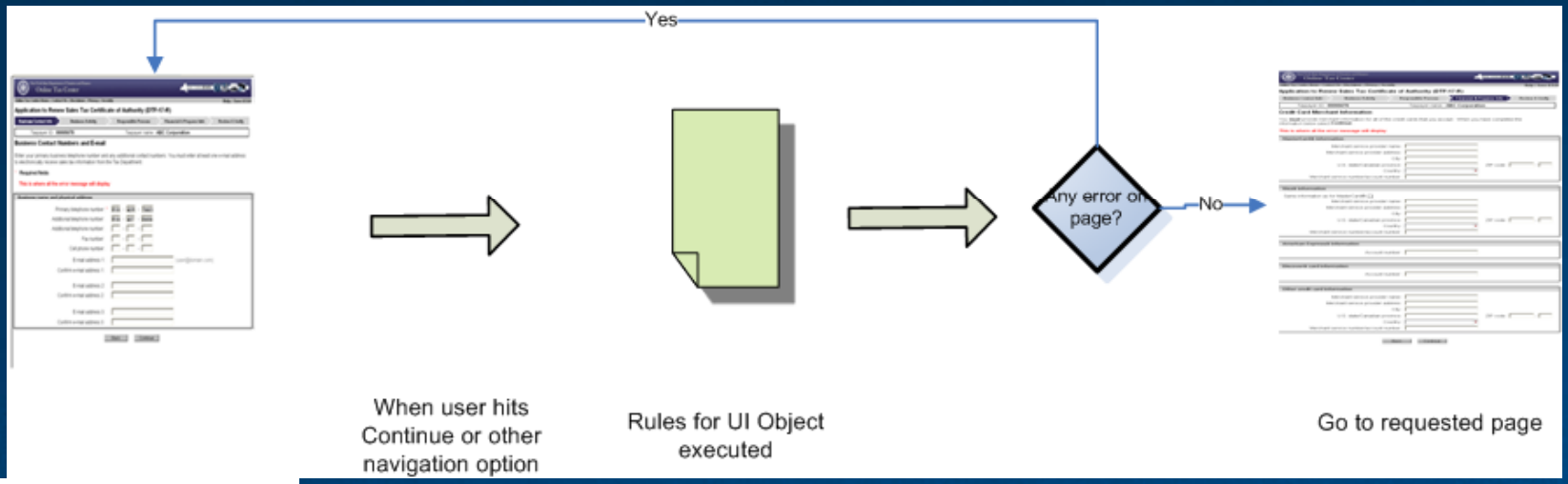
```
<transaction>
  <form formid = “a’>
    <field1>134</field1>
    <field2>abc</field21>
  </form>
  <exceptions>
    <context>a</context>
    <errNbr>17</errNbr>
  </exception>
</transaction>
```

A Form has a UI object has an object enforcing form rules has an XML segment in the transaction XML

In this way pages can be coded separately and following the same pattern and integrated into a web application.

Web Navigation Pattern

Web Application



When user hits Save, XML is saved into Transaction table as PureXML

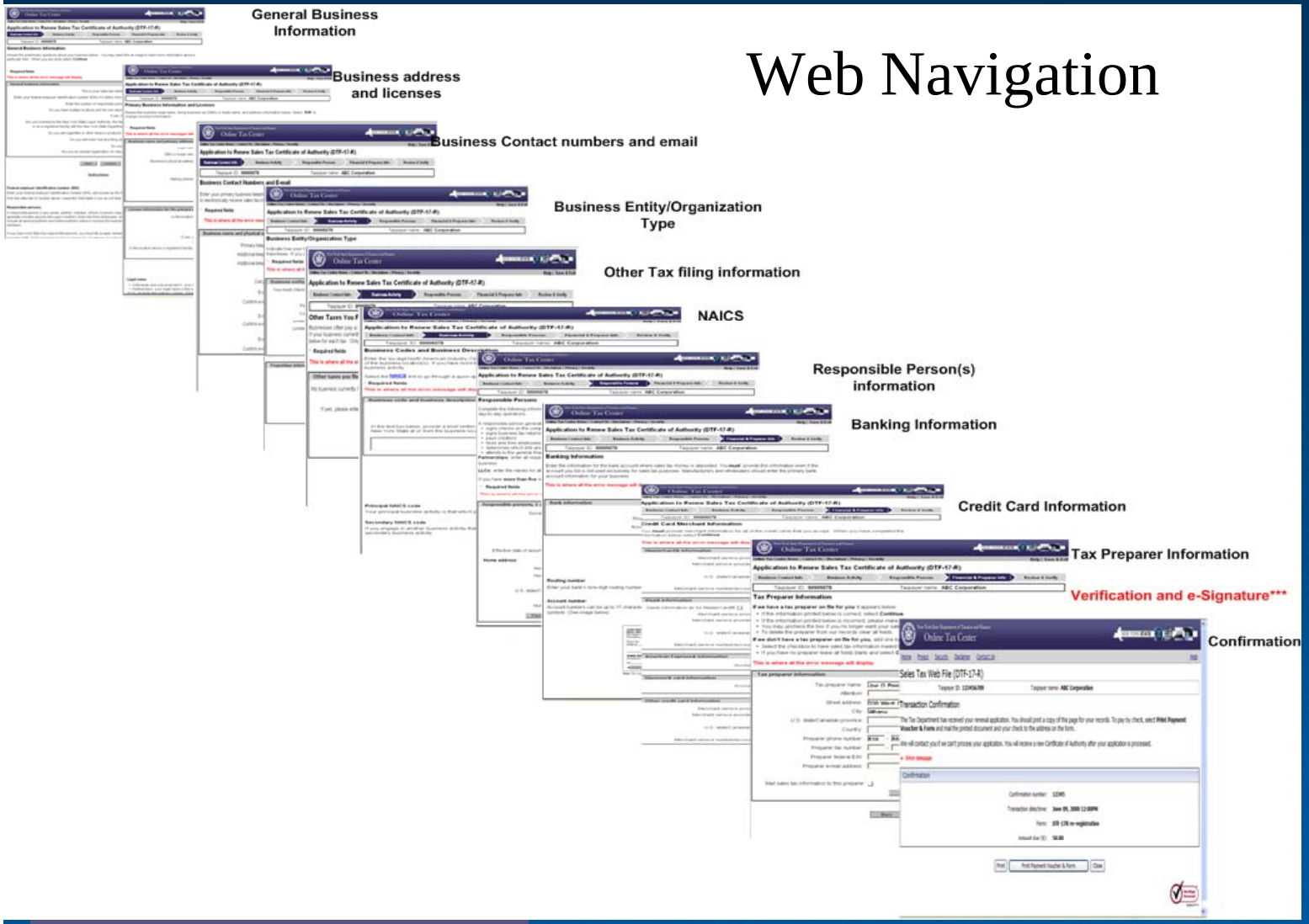
```
<transaction>
  <form formid = "a">
    <field1>134</field1>
    <field2>abc</field21>
  </form>
  <exceptions>
    <context>a</context>
    <errNbr>17</errNbr>
  </exception>
</transaction>
```

Web Navigation highlights

- All pages coded exactly the same
- Single XML table allows restart
- Navigation patterns enforced (Wizard, conditional, etc.)
- Process server allows for externalization of navigation
- Common error handling
- Can use previous filing to start new transaction

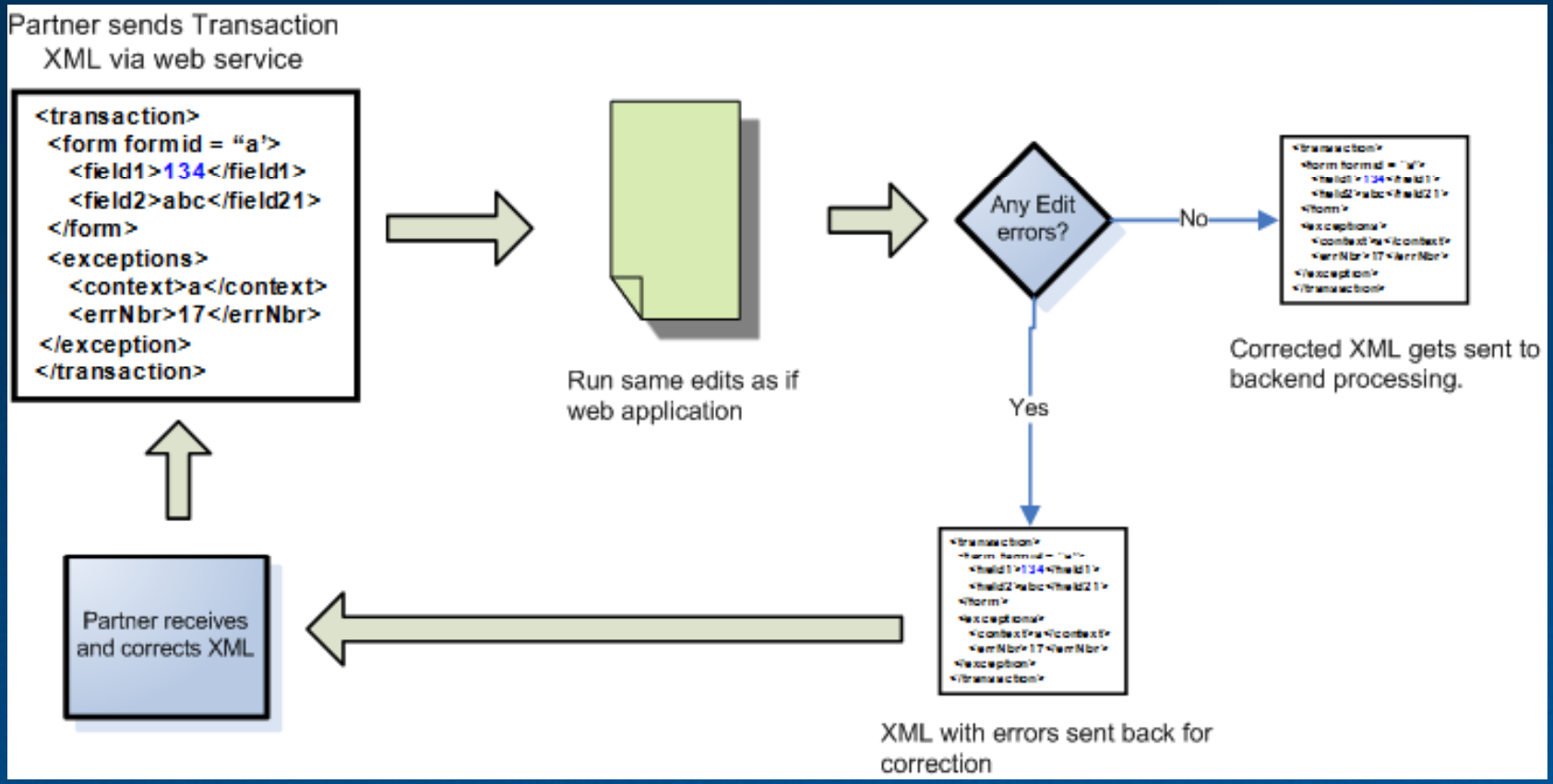
Web Application

Web Navigation



Web Service Pattern

Web Service

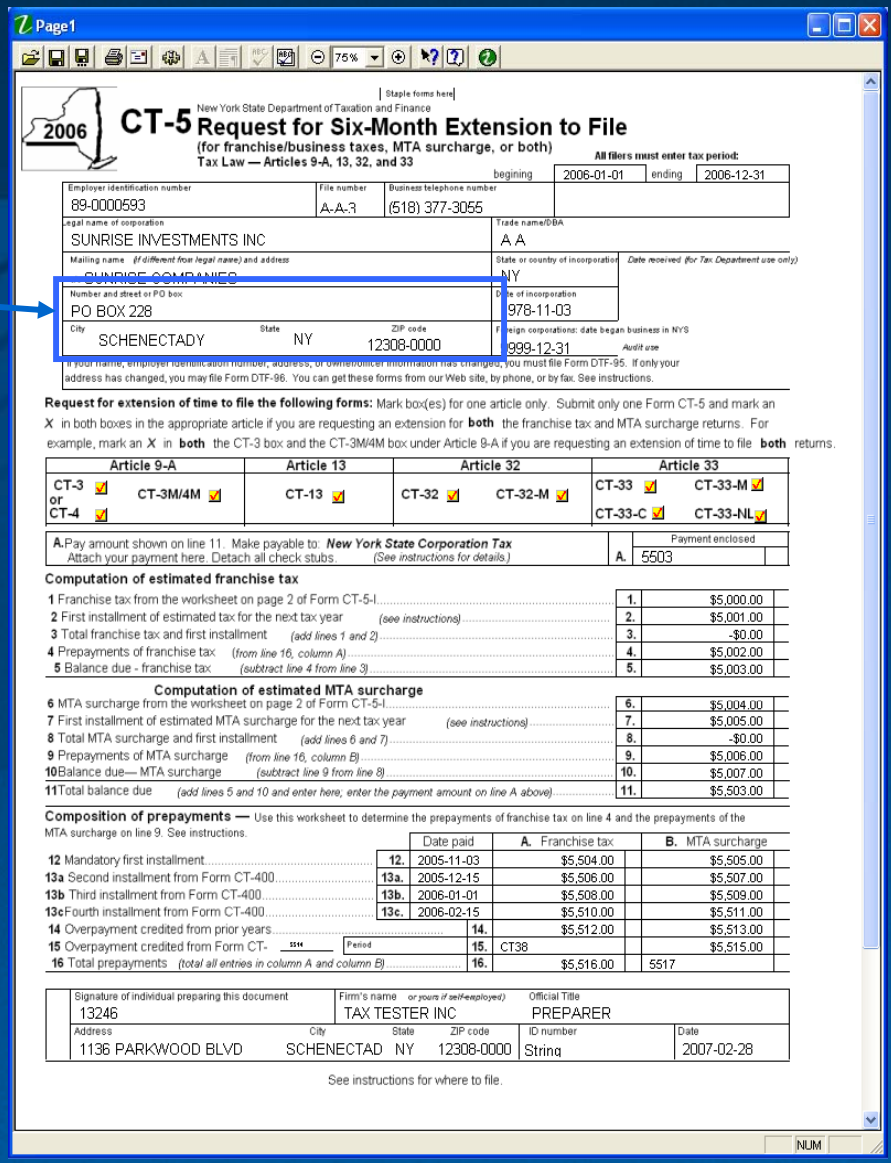


Web Service Process highlights

- Same pattern serves all web services
- Leverages same rules as web
- Once web is established no additional coding required

XML within CM

```
<USAddress>
  – <Address>
      <AddressLine1>PO BOX 228</AddressLine1>
      <City>SCHENECTADY</City>
      <State>NY</State>
      <ZIPCode>123080000</ZIPCode>
      </USAddress>
  – </Address>
</USAddress>
</StateOfIncorporation>
<HdrCode>
  – <FederalReturnFiledOther>String</FederalReturnFiledOther>
      <FilerClassificationCode>AA3</FilerClassificationCode>
      <FormType>CT5</FormType>
      <ReturnTypeCode>CT5</ReturnTypeCode>
      <SoftwareDeveloper>
      <DeveloperName>
<BusinessNameLine1>Sunrise Investments Inc</BusinessNameLine1>
<BusinessNameLine2>A A</BusinessNameLine2>
```

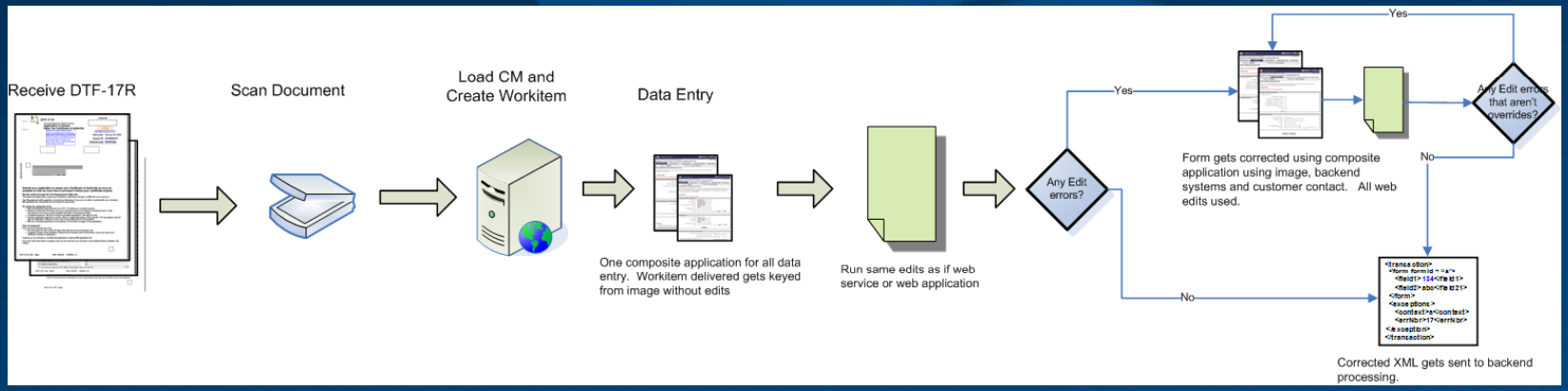


Receive XML and show the document in the form (integrated with our EDMS)

- **user view is independent of channel!**
- **all data received is stored in one table**
- **form can be used as input and for correction**
- **PureXML solution uses advanced indexing**

Paper Process Pattern

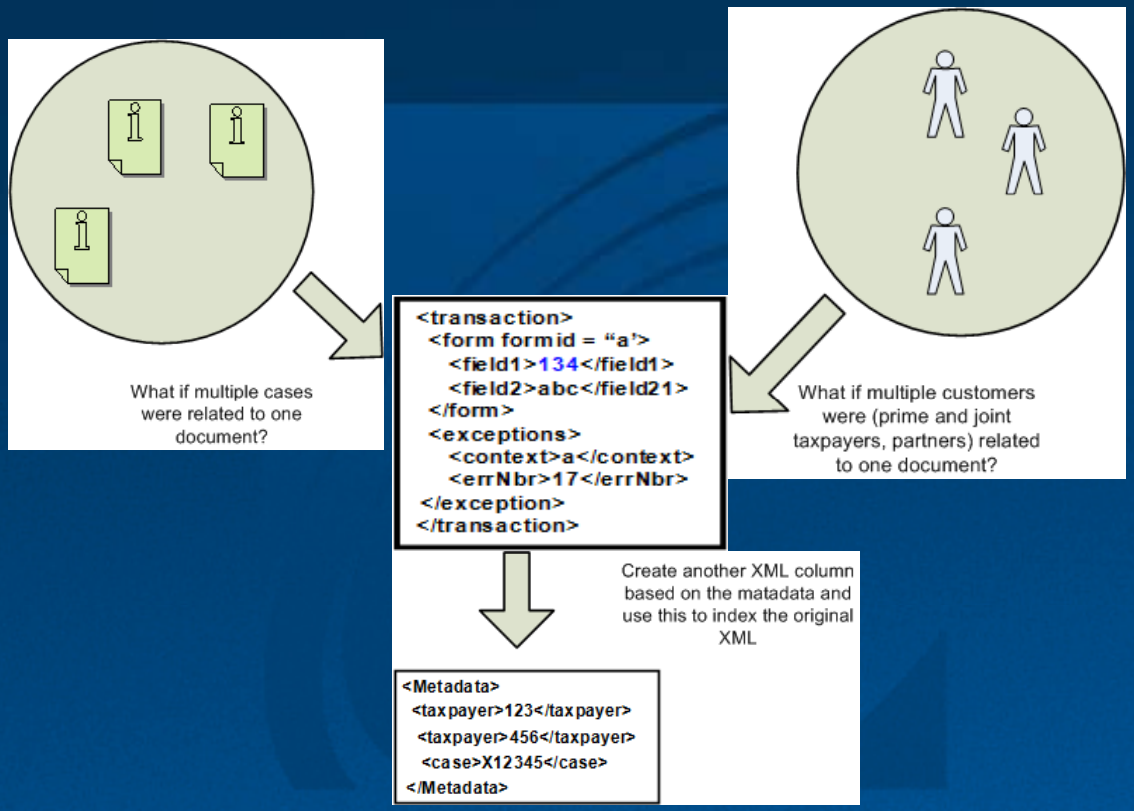
Paper



Paper Process highlights

- Same pattern serves all paper processes
- Leverages the same form Interfaces
- Leverages same XML database
- Leverages same rules as web
- Has built in capabilities to have different acceptance rules for paper

XML Indexing

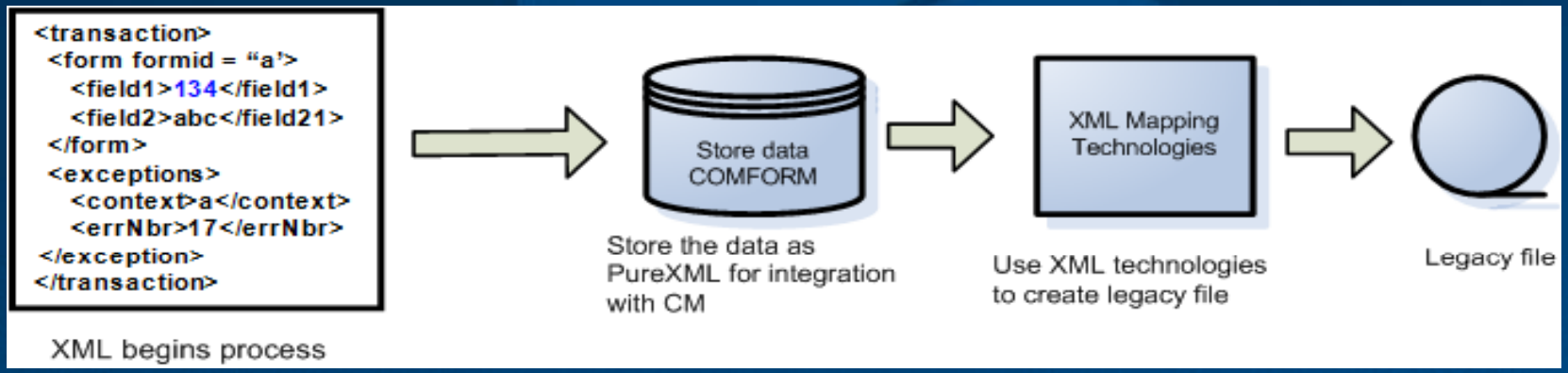


XML indexing highlights

- Allows for a many to one indexing scheme
- Index fields can be added on fly
- Supports optional index

Legacy Processing

Legacy Integration

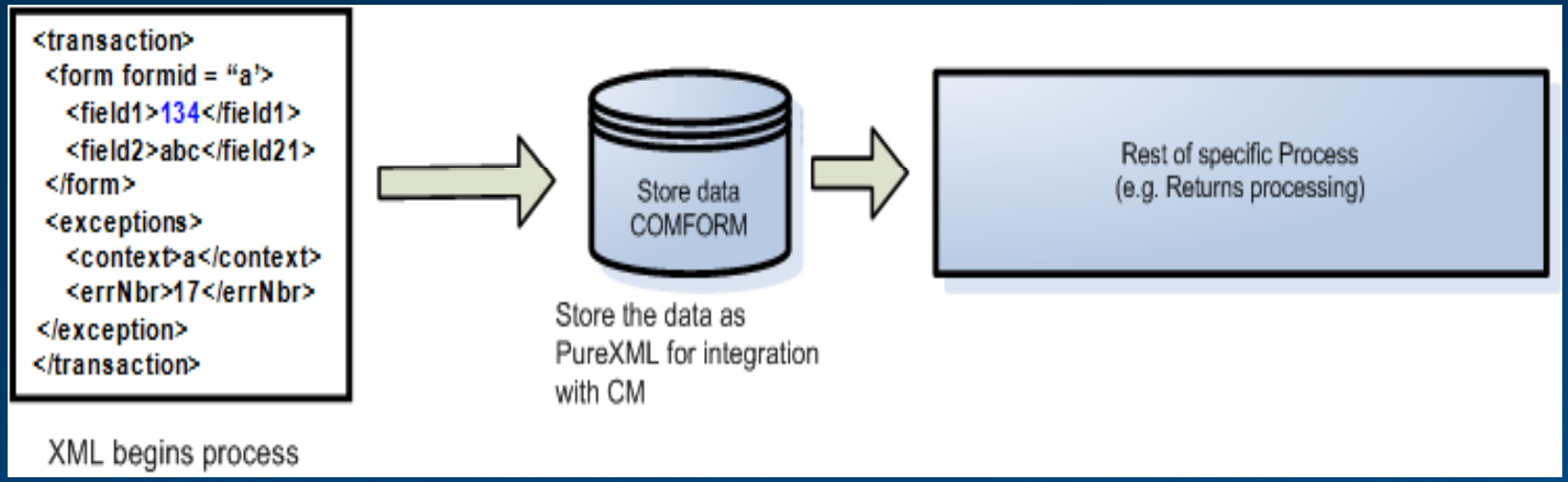


Legacy Integration highlights

- Allows for modernizing data capture while keeping legacy processing
- Single XML table stores all received documents and integrated with CM
- Many mapping services are re-usable

Process Automation Pattern

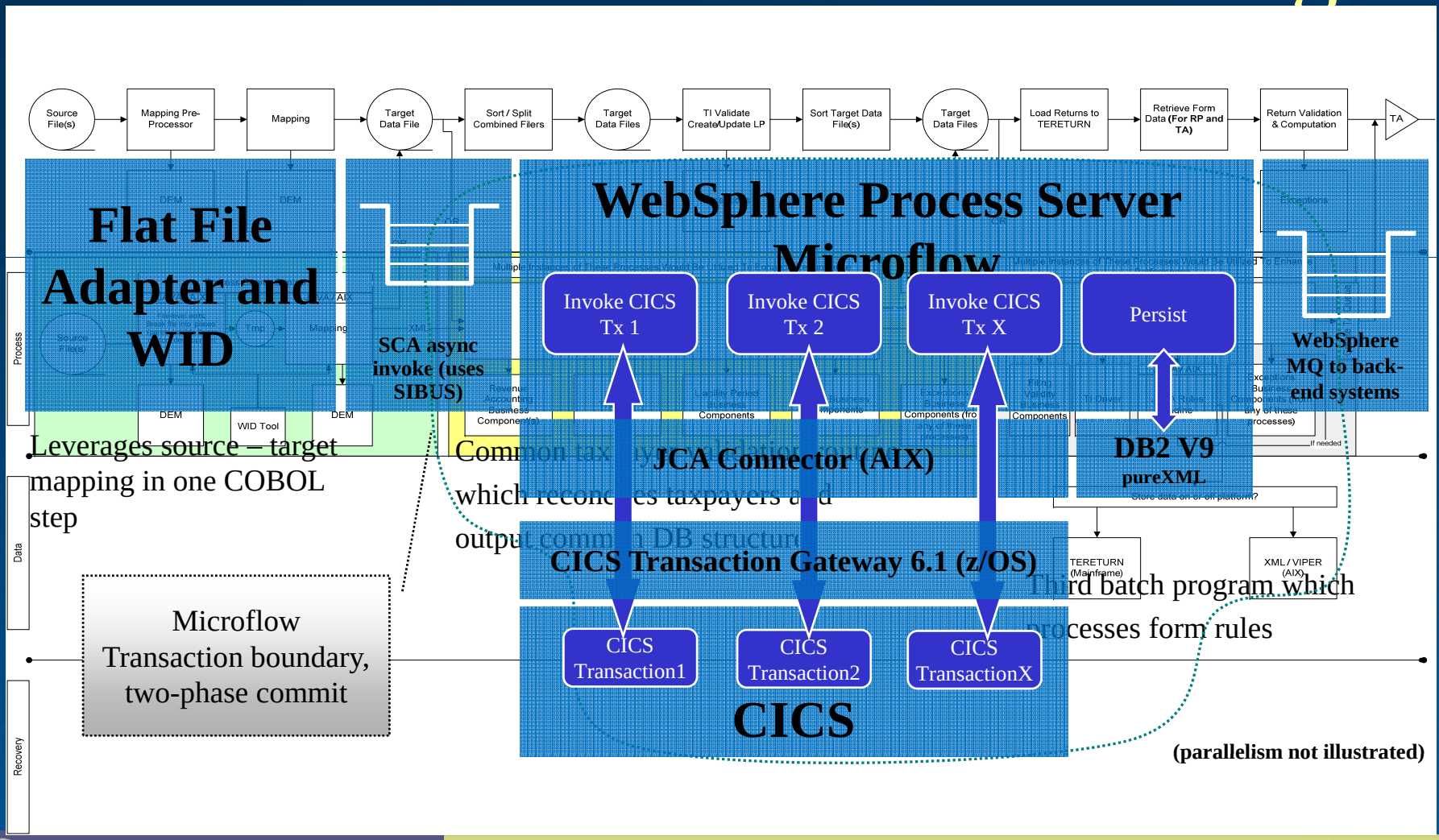
Process Automation



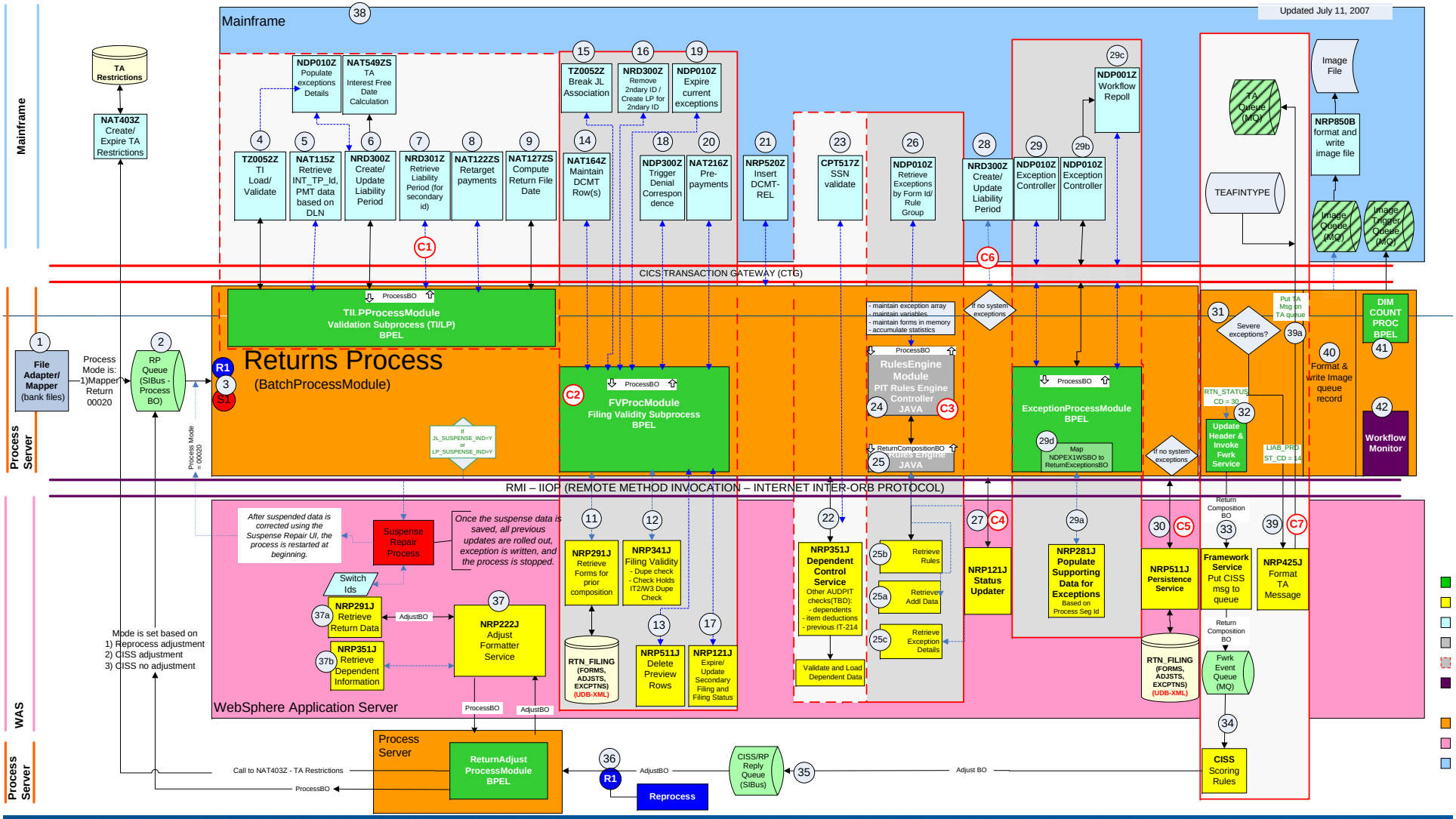
Process Automation highlights

- Can be integrated with UI for complete inline processing
- Single XML table stores all received documents and integrated with CM
- Process can be reported on through Monitor
- Operational data system becomes XML

e-MPIRE R3 Returns Processing



Returns Process (BatchProcessModule)



NY State Tax XML/SOA Processing

- PIT (Personal Income Tax)
  - 11M returns processed
  - Peak in April: 390,000+ per day
  - Up to 14,500 different data elements for the filings (60% Electronic)
  - 6M Refunds ($4.9 B), direct deposit up 13.1%, checks down 6%
  - Electronic extensions up 160% (439,000)
- Corporate Tax
  - IRS ELF Program – 2007 - 32,317, 2008 -193,977
  - Peak month: 100,000 returns in April 2009
  - Peak day: 20,000 returns
- Sales Tax
  - 1Q2009 : 60,000 on the Web
  - 1Q2009 : 400,000 from partners
- Withholdings Tax
  - 50,000 web filings of XML
- STAR Property Tax Rebate Application (Tax Refund)
  - 3.5 web applications in a 3 month period

NYS Tax at Direction

- **Convert other Subsystems (Domains) to XML**
  - **Simplify conversion**
  - **Map data structures closer to the business**
  - **Leverage the rules engine**
- **Expand the use of web navigation with integration into operational XML**
- **Incorporate more XML enabled tools to speed delivery and improve product**
- **Leverage the XML data in new ways (AJAX, REST, RIA)**

Contact

- **Jim Lieb**
  NYS Department of Taxation and Finance
  james_lieb@tax.state.ny.us