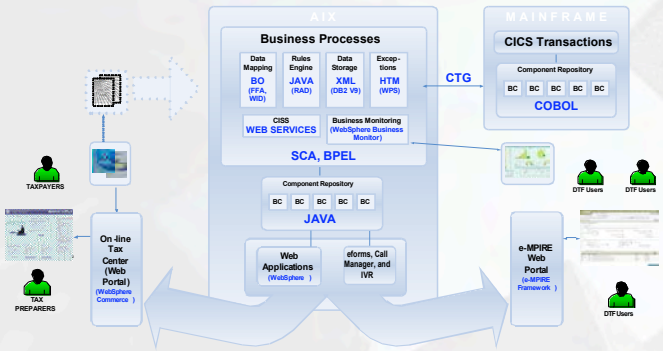
2009 FTA Technology Conference and Exhibition

# XML Facilitated Web Pattern Development at New York State Taxation and Finance

James Lieb, Director – Common Services and Case Management
NYS Department of Taxation and Finance



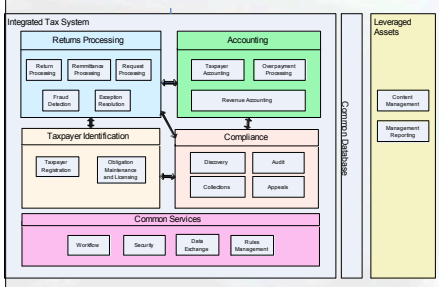
## Business Modernization Overview




- JAVA Business Components
- Business Processes
  - Data Mapping - BO
  - Data Storage - XML
  - Rules Engine - Java
  - Exceptions - WPS
- CICS Transactions
- Paper Input
- Business Monitoring
- Internal/External Web
- eforms, Call Manager (VOIP), and IVR

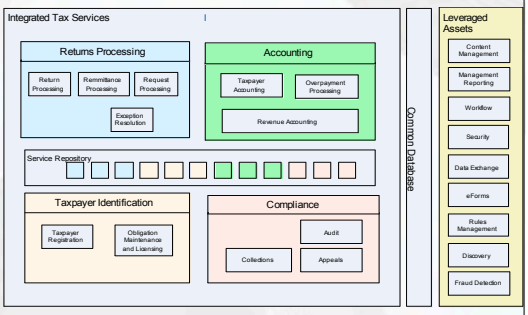
**TRANSACTIONAL**

# Integrated Tax Systems to Integrated Tax Services




For our Modernization we started with COTS solutions as a starting point but ..

Instead evolved into an SOA solution that leveraged existing assets and was more open to integrating industry standard products for common services.








# Returns Processing Summary

## Operational XML → Next Generation DB Design


| 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 limitations

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

# XML's Transformation

## Technological Convergence changes view of XML

| | Traditional | Strategic |
|---|---|---|
| Usage Patterns | • Data sharing<br>• Data transfer | • Transactions<br>• Execution format<br>• Persistence protocol |
| Technologies | • Parsers (SAX, DOM) | • Web Services (WSDL)<br>• XQuery, XPath<br>• DOM/SDO<br>• XML/Relational DBMS<br>• eForms<br>• AJAX, REST, RIA, Atom |
| Business Issue Resolved | • Simplified Collaboration | • Expanded channels<br>• Enabled partners<br>• Customer empowerment<br>• Faster solution delivery |





## Layers of Transactional XML Domain Model


- 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

# Transactional XML: Computations

Correcting Information during processing
- Math errors
- Credit claims
- Fraud

***Tag based solution***

```
<Wages "xxx"/>
<WagesComputed "xxx"/>
<WagesComputedReason "xxxx" />
```

***Attribute based solution***

```
<Wages Captured="xxx", Computed="xxx", ComputedReason="xxxx" />
```

**Attribute based advantages**
- Reduce tags
- More efficient processing map
- Reduced DB storage
- Easier integration with Rules Engine





# Transactional XML: Exceptions

Describing a document that has some processing issues
- Critical processing issues
- Informational issues
- Mechanism for getting work stopped/reviewed/corrected

***Exception Segments***

```
<filing>
  <taxpayer>
  </taxpayer>
  <forms>
     <w2 id="512">
     </w2>
  </forms>
  <exceptionsSeg>
    <exception issue="xxx", severity=1/>
    <exception issue="yyy", severity=0/>
  </exceptionSeg>
</filing>
```

**Exception Segment advantages**
- State persisted with business object
- Facilitates exception handling
  - UI
  - Process routing
- Easy integration with Rules Engine

# Transactional XML: History

***History Segments***

```
<filing>
  <taxpayer>
  </taxpayer>
  <forms >
     <w2 id="512", efctvDate="2009-06-29">
     </w2>
     <w2 id="512, efctvDate="2009-01-16">
     </w2>
  </forms>
</filing>
```

**History Segment advantages**

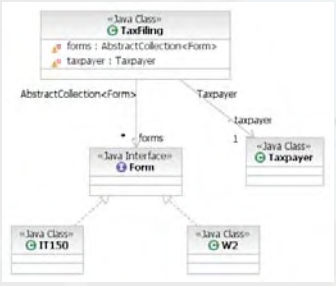
- All changes within the business object
  - Maps to the Audit Folder
- Facilitates return adjustments
- Works with rules engine
- Easy to compare forms and filings by date
- Facilitates off line use





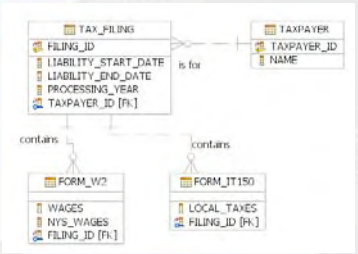
# XML Execution Model - Traditional

## Object-Relational Mapping




OR
Tools/Frameworks

Mapping Info



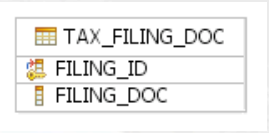

Processing Model –Object Classes

Relational DB

# XML Execution Model - Strategic

## XML Mapping

```
<?xml version="1.0" encoding="UTF-8"?>
<filing>
    <taxpayer>
        <name>Joe</name>
        <id>123123123</id>
    </taxpayer>
    <forms>
        <w2 id="512">
            <fedWages claimed="70000" computed="70000"/>
            <nyWages claimed="70000" computed="70000"/>
        </w2>
        <it150 id="342">
            <localTaxes claimed="6700" computed="6700"/>
            <salesTax claimed="3500" computed="3500"/>
        </it150>
    </forms>
</filing>
```


Processing Model - DOM/SDO

Object-Relational XML DB

### Model Benefits

- Easy to process
- Easy to persist
- Easy to query
- Easy to share

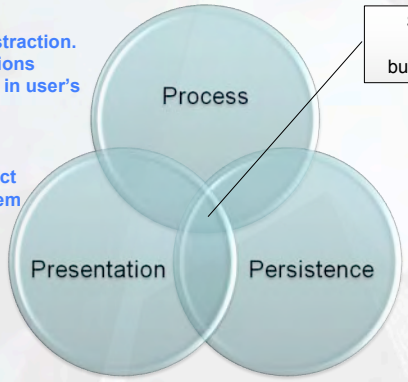




# Facilitating Business Alignment

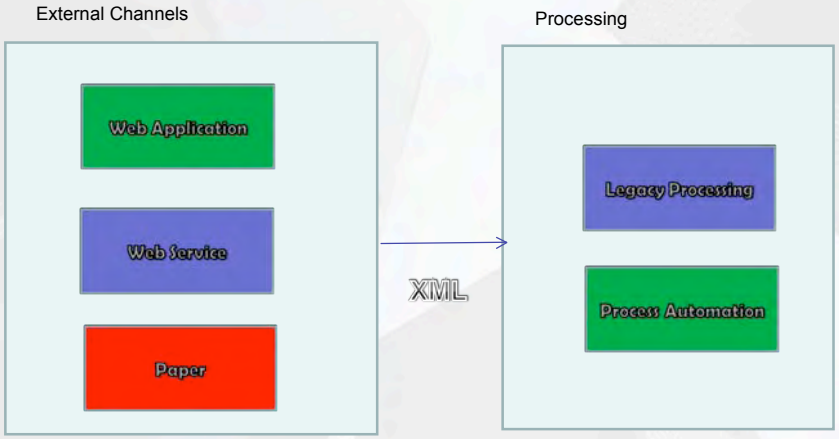
**Eliminates Technical Abstraction. All processes, functions and reports "configured" in user's terms.**

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




Sharing the same Business defined business object (XML)

# Development Enablement
## Establishing Patterns








# 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?

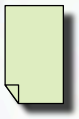
Web Application

*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….

A Form

has a UI object



has a object enforcing form rules

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

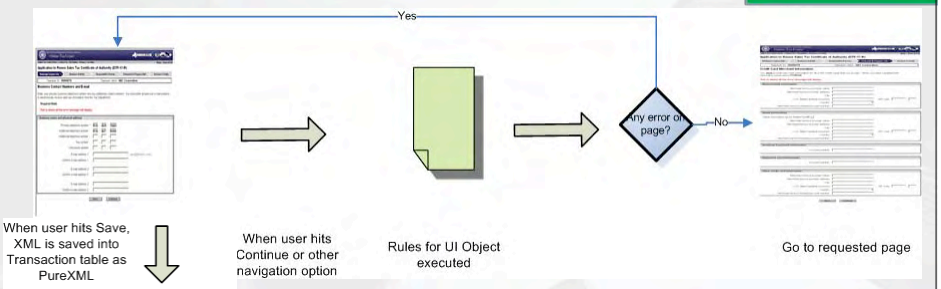
has an XML segment in the transaction XML

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





Web Application



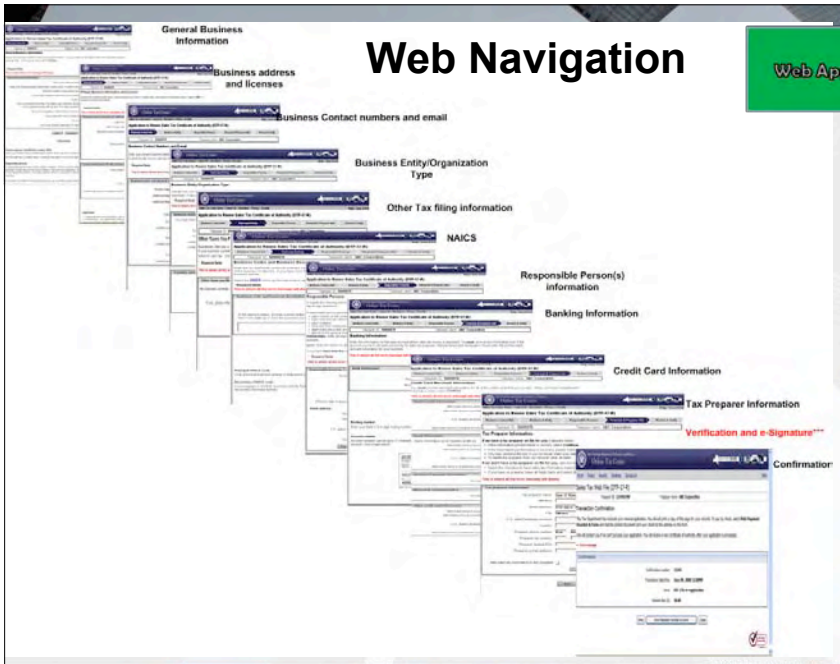

```
<transaction>
  <form formid = "a'>
    <field1>134</field1>
    <field2>abc</field21>
  </form>
  <exceptions>
    <context>a</context>
    <errNbr>17</errNbr>
  </exception>
</transaction>
```

## 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 Navigation

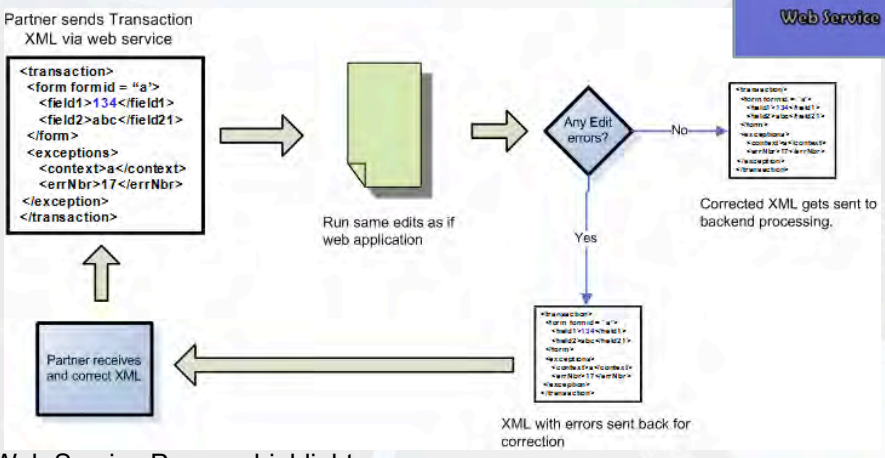
Web Application





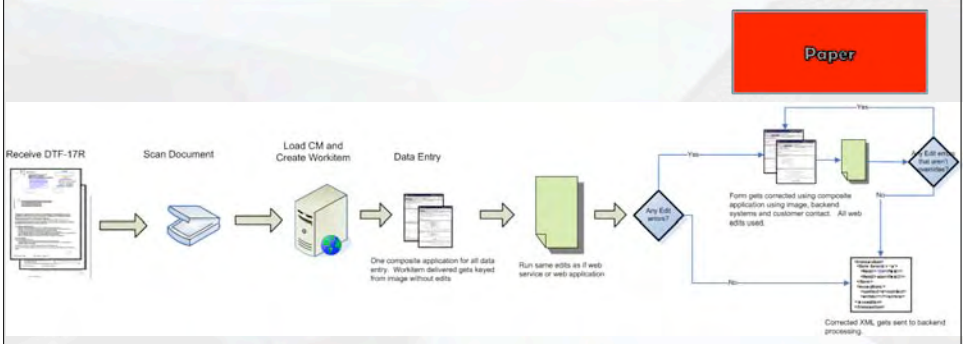



Web Service




## Web Service Process highlights

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




## Paper Process highlights

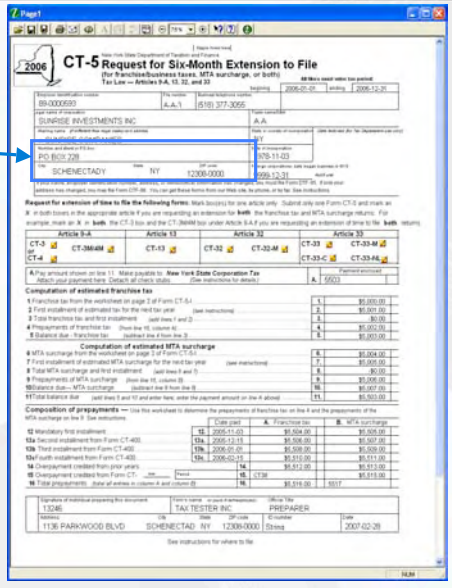
- Same pattern serves all paper processes
- Leverages the same form Interfaces
- Leverages same XML database
- Leverages same rules as web
- Has built in capabilities to have different acceptance rules for paper





# XML within CM

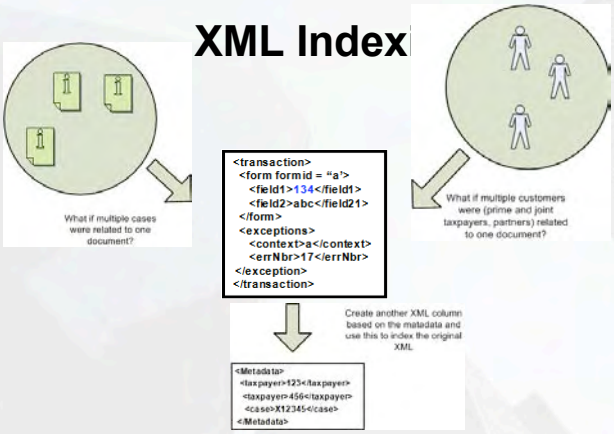
```
<USAddress>
  – <Address>
      <AddressLine1> PO BOX 228 </AddressLine1>
      <City> SCHENECTADY </City>
      <State> NY</State>
      <ZIPCode> 123080000 </ZIPCode>
      </USAddress>
  – </Address>
</USAddress>
</StateOfIncorporation>
<HdrCode>
  – <FederalReturnFiledOther>String</FederalReturnFiledOther>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




**Receive XML from the IRS 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**

# XML Index



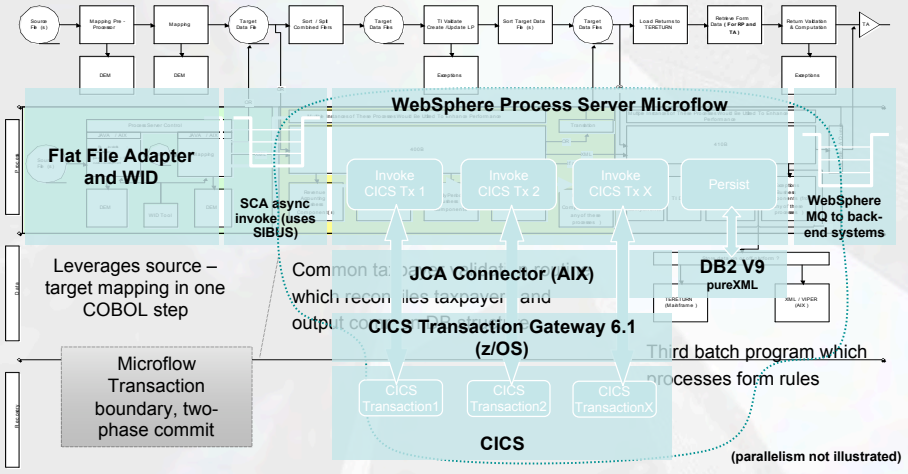

## XML indexing highlights

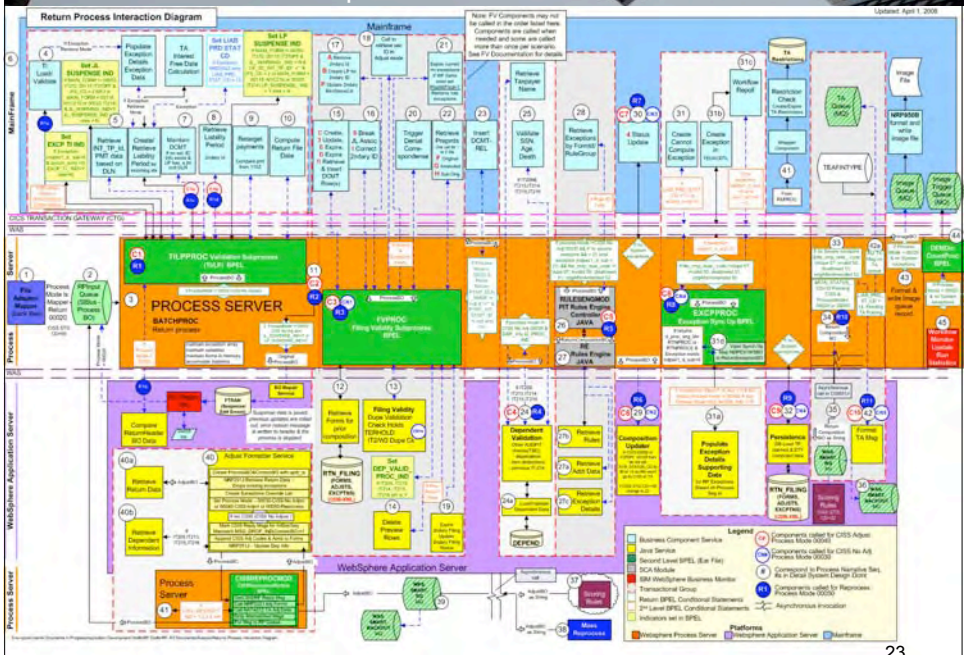
- Allows for a many to one indexing scheme
- Index fields can be added on fly
- Supports optional index





# e-MPIRE R3 Returns Processing










# NY State Tax 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

## What's next?

- 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)





# Questions?