Canonical Standards and the use of TypeCode

Kevin Himka  Boeing Enterprise Information Architect
Jeff Rice  Boeing BDS Project Architect
Jim Wilson  OAGi
Ralph Hertlein  VP Hertlein Group
What We Do Today

- **Commercial Airplanes**
  - Boeing 7-series family of airplanes leads the industry.
  - Commercial Aviation Services supports carriers worldwide.

- **Defense, Space & Security**
  - World’s largest manufacturer of military aircraft.
  - Global Services & Support provides training, maintenance, and other services to government customers worldwide.
  - World’s largest provider of commercial and military satellites and major service provider to NASA.
  - Large-scale systems integration and support; develop networking technology and solutions.

- **Boeing Capital Corporation**
  - Financing solutions focused on customer requirements.

Connect and protect people globally
Global Boeing

- Products and services support to customers in more than 150 countries
  - Revenue in 2015: $96 billion
  - 70 percent of commercial airplane revenue historically from customers outside the United States

- Manufacturing, service and technology partnerships with companies around the world
  - Contracts with more than 20,000 suppliers and partners globally

- Research, design and technology-development centers and programs in multiple countries

- Approximately 160,000 Boeing employees across the United States and in more than 65 countries

Partnering worldwide for mutual growth and prosperity
Enterprise Focus –
SOA Based Services with XML Schema

- Standardized, tailored schemas to produce enterprise XML messages that are:
  - Consistent, repeatable, predictable, canonical
  - Optimized for tools, processes and clarity
  - Fully documented for reuse, maintenance and upgrades
  - Adaptable to be responsive to business needs
  - Improved project startup
  - Support Service Oriented Architecture Strategy
- Governed to assure this continues to provide payback over time
- Required by certain government agencies
So What is a @typeCode

W3C Recommendation since 1998
- Attribute of an element (thing)
- Adds detail to a less specific thing
- Provides “optional” specific information
- Compensation for differences
- Excellent programming choice and control
- Allows for more detailed data
Why Standardize and use typecodes…?

- This means something to someone somewhere but we don’t know so just map it?
- Is Name1 equal to the suppliers contact first name or the name of the supplier itself?
- Does AddressLine1 include FirstName, MI, LastName or LastName, FirstName, MI or?
- Is the document referenced here the purchase order or the requisition?
- Is this the dateTime that the shipment is ready to pick up, the time the shipper actually picked it up or when the packaging team released this to be picked up?
- Is this the address of the Shipper’s main office, local office or their dock?
- Who should this Shipper get in touch with to allow access to the dock?
- Is this message an approval to purchase or just to get approval to purchase?
Attributes add optional details and more programming control to XML data

Canonical PO without @typeCodes

• Who’s description is this – the sender’s? (guess I will have to go look that up)
• What does this code mean – who owns it?
• Dimension Measure – what unit?
• Guess this is an Army document?
• That is a great “indicator” – it says true – and that means what? – I will have to go look that up…
• WOW – a time – what does that mean – 1 must mean an hour or is it a day or week? And I will assume that there is no time zone involved – right?
Attributes add optional details and more programming control to XML data

Canonical PO with @typeCodes

```xml
<Description typeCode="SystemADescription">this part is used to replace some special stuff</Description>
<Note typeCode="SystemAOrderLineText">call me when you get this</Note>
<Classification typeCode="SystemBClassificationCodings">
  <Codes>
  </Codes>
</Classification>
<Dimensions>
  <Measure unitCode="BOX." typeCode="SystemBAplicationUnitOfMeasure">23</Measure>
</Dimensions>
<DocumentReference typeCode="SystemABuyersContract">
  <ID typeCode="SystemABuyersContractNumber">CN3838ARMY777</ID>
  <LineNumberID typeCode="SystemABuyersContractLineNumber">393</LineNumberID>
</DocumentReference>
<ManufacturingParty>
  <CAGEID schemeID="SystemBManufacturersCAGE">39FR908</CAGEID>
</ManufacturingParty>
<Extension typeCode="ApplicationReference">
  <Indicator typeCode="SystemBFractureCriticalIndicator">true</Indicator>
  <Time typeCode="SystemBDockToStockLeadTimeDays">1</Time>
</Extension>
```

What do attributes called @typeCodes DO?

- Some of these things come from SystemB and some from SystemA
- I now know what that Measure really means
- Yes that is a SystemA Contract
- Remove the ASS_U_ME
- I can select via occurrence number or actual @typeCode
- Guess that indicator means this item will need to be inspected for fractures
- I have a FRAGILE system that does not use attributes – guess I will have to ask AI to transform to non-standard format for me
- Most canonical components and elements are designed with @typeCodes (attributes) or similar
- TypeCoding affords creation and use of clearer semantic meaning, business intent, enterprise application differentiation and audit trails
- TypeCoding permits leveraging the canonical standard (OAGIS) in a timely process while still following the enterprise needs – typeCoding also exposes semantic that should go back…
- Governance team standardizes the @typeCoding
- Enables positive programming – Know vs. Assume – 1=?
How it All Works Together…

Standard Guideline Spreadsheet – Drives easy entry, tracking and updates

1. `<Extension typeCode="BoeingShipmentType">
   <Code typeCode="BoeingShipmentTypeCode">MS</Code>
   <Text typeCode="BoeingShipmentTypeDescription">MISCELLANEOUS</Text>
</Extension>

2. `<Extension typeCode="BoeingShipmentResponsibility">
   <Text typeCode="BoeingShipmentResponsibleProgran">777</Text>
</Extension>

3. `<Extension typeCode="MisusedBESSyFields">
   <Text typeCode="MisusedBESSyDescriptionField">?GOLD SEG. CODE</Text>
</Extension>

Sample XML File derived from the GEFEG Guideline with some manual adjustments

GEFEG Model – Saves and manages the entire process from the spreadsheet imports to the manual entries with Notes and Note contents

Imported into and exported from the GEFEG Model to populate values and turn on elements

Tailored xsd is derived from the GEFEG Model – from the spreadsheet imports
Definitions & Usage of the spreadsheet

1. Attribute Value List – Actual possible expected mapping values listing @typeCode, @schemeID, etc… detailed typing of the canonical element – MUST USE – offers full XPath control

2. Example – Sample data for testing Provides real world sample values to guide X-Mapping – permits easier creation of sample XML for test and validations

3. Enterprise Business Terms – Overall enterprise canonical model business term Canonical meaning in the Boeing Enterprise – Boeing Enterprise generic value at a business level (not dB or app)

4. Business Process Specifics – Comments concerning this use in this process - also details for application specific guidance such as FlexFields
Tracking Your Work –
Recommended Organization!

Set up the following directory structure on your machine based on the Service, BOD and on the date and time of next review cycle:

- **BOD1 generic name**
  - Review date
    - GEFEGZefs
    - HTMLDocs
    - SprdShts-CSVs
    - TailoredSchema
    - XMLExamples
  - Review date
  - Same as above

Example for Organization
Mapping Canonical Logic (OAGIS)

- Semantic logic first – typeCoding second

- Item –
  - any product, service or offering
  - ID – how to identify from the sending system point of view
  - ItemIDSet – how to identify from associated systems point of view using @attributes
  - ID Group – how to identify from other generic and standard specific points of view
  - Classification – generic grouping
  - Specification – how to define the item
  - Etc.

- Extension –
  - Enterprise specific point of view
  - Needs to be grouped via agreement and defined by @attributes and @typeCodes
RULE 1 – Reuse First

- Refer to what has been mapped already – look to others for confirmation and reuse
- Only add if not there already – make sure
- Define by agreeing – not alone
- Use other standards and models as reference
- When possible – recommend back to standards body for inclusion in future versions
Rule Two - Communicate

- Define carefully and ask others
- Be ready to modify prior to use
- Acceptance better than ownership

<table>
<thead>
<tr>
<th>Enterprise&quot;Speak&quot;</th>
<th>Canonical&quot;Speak&quot;</th>
<th>@typeCoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part</td>
<td>Item OR ItemInstance (Item is conceptual, ItemInstance is real)</td>
<td>/@typeCode=RawMaterial, Part, Assembly</td>
</tr>
<tr>
<td>Supplier</td>
<td>PartyMaster, SupplierParty or Party (When offered in the canonical or typed)</td>
<td>/@typeCode=EnterpriseSupplier or /@typeCode=SAPSupplier</td>
</tr>
<tr>
<td>Discount</td>
<td>Allowance</td>
<td>/@typeCode=EnterpriseContractDiscount</td>
</tr>
<tr>
<td>BEMSID</td>
<td>PartyMaster/ID</td>
<td>/@typeCode=BEMSID</td>
</tr>
<tr>
<td>Company Name</td>
<td>BuyerParty/Name</td>
<td>/@typeCode=eMDMPartyName</td>
</tr>
<tr>
<td>Blanket Order</td>
<td>Quantity</td>
<td>/@typeCode=BlanketOrderTotal</td>
</tr>
</tbody>
</table>
## Canonical Mapping Details - 1

Reference prior mappings for ANY mapping – REUSE!!!!! –

**example:**

<table>
<thead>
<tr>
<th>Part number</th>
<th>GetShipment/DataArea/Shipment/ShipmentItem/ItemInstance/ID/@typeCode=EnterprisePartNumber</th>
</tr>
</thead>
</table>

Follow semantic standard meaning –

<table>
<thead>
<tr>
<th>All of this Shipment</th>
<th>GetShipment/DataArea/Shipment/ShipmentHeader/ID/@typeCode=EnterpriseShipmentIdentification</th>
</tr>
</thead>
<tbody>
<tr>
<td>All of this Container</td>
<td>GetShipment/DataArea/Shipment/ShipmentUnit/ID/@typeCode=EnterpriseContainerIdentification</td>
</tr>
</tbody>
</table>
Use attributes to permit fuller semantic meaning at the enterprise level

- OAGIS attributes permit more complete semantic definition from an enterprise level
- Use full, clear naming – NO ABBREVIATIONS – need clarity for future users
- /ShipmentHeader/ID – whose ID is this? – Enterprise, Application, some form of Party (Shipper, ShipTo, ShipFrom…)

<table>
<thead>
<tr>
<th>Attribute to use</th>
<th>Preferred Order</th>
<th>Examples</th>
</tr>
</thead>
</table>
| /@typeCode               | First                      | • EnterpriseShipmentIdentifier  
                          |   • ERPShipmentIdentifier  
                          |   • ShippersShipmentIdentifier |
| /@schemeID               | When no @typeCode exists   | Same as @typeCode                                                        |
| /sequenceNumber          | Last resort                | • 1, 2, 3, 4, …                                                           |
Standards Vocabulary vs Enterprise vs Application vs Conventions

- STANDARD WINS, then Enterprise, then conventions, then application
- Business reasoning WINS – not dB
- Examples:

<table>
<thead>
<tr>
<th>Enterprise”Speak”</th>
<th>Canonical”Speak”</th>
<th>@typeCoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part</td>
<td>Item OR ItemInstance (Item is conceptual, ItemInstance is real)</td>
<td>/@typeCode=RawMaterial, Part, Assembly</td>
</tr>
<tr>
<td>Supplier</td>
<td>PartyMaster, SupplierParty or Party (When offered in the canonical or typed)</td>
<td>/@typeCode=EnterpriseSupplier or /@typeCode=SAPSupplier</td>
</tr>
<tr>
<td>Discount</td>
<td>Allowance</td>
<td>/@typeCode=EnterpriseContractDiscount</td>
</tr>
<tr>
<td>BEMSID</td>
<td>PartyMaster/ID</td>
<td>/@typeCode=BEMSID</td>
</tr>
<tr>
<td>Company Name</td>
<td>BuyerParty/Name</td>
<td>/@typeCode=eMDMPartyName</td>
</tr>
<tr>
<td>Blanket Order</td>
<td>Quantity</td>
<td>/@typeCode=BlanketOrderTotal</td>
</tr>
<tr>
<td>Quantity</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Use multi-level type coding to provide greater clarity

- Two level: *Military Contract Back Order Amount* in PurchaseOrder
  - ContractReference/@typeCode=MilitaryProcurementContract
  - ContractReference/Quantity/@typeCode=BackOrdered

- Three level: *CrossDock Primary Contact, Primary Cell Phone Number* in Shipment
  - Party/@typeCode=CrossDockProvider
  - Party/Contact/@typeCode=DockPrimaryContact
  - Party/Contact/TelephoneCommunication/@typeCode=PrimaryCell

Remember – full XPaths control is the goal for all possible cardinalities (0..n) at multi levels requires path control
## Multi-level Attribute Mapping

**Multi–level typing spreadsheet example:**

Two Levels - multiple use

<table>
<thead>
<tr>
<th>XPath</th>
<th>Cardinality</th>
<th>Use</th>
<th>Attribute Values</th>
</tr>
</thead>
</table>
| AcknowledgeMoveProduct/DataArea/MoveProduct/MoveProductLine/DocumentReference | 0..n         |        | 1. TransactionReference  
2. OutVoucher  
3. AlternateVoucher  
4. PickTicket             |
| AcknowledgeMoveProduct/DataArea/MoveProduct/MoveProductLine/DocumentReference/@typeCode | required   |        |                                                       |
| AcknowledgeMoveProduct/DataArea/MoveProduct/MoveProductLine/DocumentReference/ID | 0..1         |        | 1a. ERPTransactionReference  
1b. ERPTransactionReversalReference  
2. ERPOutVoucher  
3. ERPAlternateVoucher  
4. ERPPickTicket             |
# Multi-Level Assignment - Results Review

**Table:**

<table>
<thead>
<tr>
<th>XPath</th>
<th>Occurrence</th>
<th>Use (Property of attribute)</th>
<th>Attribute Value List</th>
<th>Example</th>
<th>Enterprise Business Terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>NotifyShipment/DataArea/Shipment/ShipmentHeader/Party</td>
<td>0..n</td>
<td></td>
<td>1. ApplicationUsers 2. SignatoryParties</td>
<td></td>
<td>Non-semantic parties involved within the Shipment - This element is not mapped</td>
</tr>
<tr>
<td>NotifyShipment/DataArea/Shipment/ShipmentHeader/Party/@typeCode</td>
<td></td>
<td>required</td>
<td></td>
<td></td>
<td>Identifies the different types of additional parties that are involved in a Shipment: 1. Users from the involved applications with this process 2. Parties responsible for contractual commitments</td>
</tr>
<tr>
<td>NotifyShipment/DataArea/Shipment/ShipmentHeader/Party/Contact</td>
<td>0..n</td>
<td></td>
<td>1. SAPUserContact 2. ContractSignOffBy 3. ContractSignOffFor</td>
<td></td>
<td>Provides guidance concerning shipment contact information and how to use - this element is not mapped</td>
</tr>
<tr>
<td>NotifyShipment/DataArea/Shipment/ShipmentHeader/Party/Contact/@typeCode</td>
<td></td>
<td>required</td>
<td></td>
<td></td>
<td>Defines the acceptable types of Contacts for this message</td>
</tr>
<tr>
<td>NotifyShipment/DataArea/Shipment/ShipmentHeader/Party/ContactID</td>
<td>0..n</td>
<td></td>
<td>1. 12945693 2. 11139697 3. 02233121</td>
<td></td>
<td>1. - Sending systems user ID information 2. - ShipmentAuthorizationSignOffBy 3. - ShipmentAuthorizationsSignOffFor</td>
</tr>
<tr>
<td>NotifyShipment/DataArea/Shipment/ShipmentHeader/Party/ContactID/@typeCode</td>
<td></td>
<td>required</td>
<td>1. SAPUserBEMSID 2. ContractSignOffByBEMSID 3. ContractSignOffForBEMSID</td>
<td></td>
<td>Defines the acceptable types of IDs</td>
</tr>
<tr>
<td>NotifyShipment/DataArea/Shipment/ShipmentHeader/Party/Contact/EmailAddressCommunication</td>
<td>0..n</td>
<td></td>
<td>1. SAPUserEmailAddress 2. ContractSignOffByEmailAddress 3. ContractSignOffForEmailAddress</td>
<td></td>
<td>This element is not mapped</td>
</tr>
<tr>
<td>NotifyShipment/DataArea/Shipment/ShipmentHeader/Party/Contact/EmailAddressCommunication/@typeCode</td>
<td></td>
<td>required</td>
<td></td>
<td></td>
<td>Identifies the different possible types of Email address that can be used or are used</td>
</tr>
<tr>
<td>NotifyShipment/DataArea/Shipment/ShipmentHeader/Party/Contact/EmailAddressCommunication/EmailAddressID</td>
<td></td>
<td>required</td>
<td>1. <a href="mailto:user182949@boeing.com">user182949@boeing.com</a></td>
<td></td>
<td>Email address(es) for the Contact being described by the @typeCoding on this Party/Contact</td>
</tr>
<tr>
<td>NotifyShipment/DataArea/Shipment/ShipmentHeader/Party/Contact/EmailAddressCommunication/EmailAddressID/@schemeID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Defines the acceptable list of email address types</td>
</tr>
</tbody>
</table>
A Word About Component Level Extensions

- Standard extension “group” appears at all components
- Relies totally on @typeCoding to differentiate
- @typeCoding requires some form of user (enterprise) standardized pattern
- Can not point to other component elements unless @typeCoding offers some pattern
- Requires a patterned model for usage patterns such as security, db additional detail, application interchange, application extensions, etc.
So @typeCodes do add value!

- Are optional – but can be trans-mapped and coded to for FRAGILE systems
- Adds tremendous value to the message use and to troubleshooting
- Defines the “owner of the data at the data level”
- Better defines the use of the date at the data level and reduces assumptions
- Better leverages the applications adaptor to add business value

- Requires a bit more and different programming
- Can easily be learned and tools have existed for creation and use for decades
- Enables serious use of business rule based processing
- Enables enterprise flexibility and adaptability
Challenges & Considerations – Mapping an Enterprise Canonical

- Mapping for an Enterprise – not Point-to-Point (PtP)
- Maximize reuse of all of the canonical (OAGIS)
  - Patterns for Message, Errors, Business Nouns, Components & Fields
  - Patterns for WS packaging
- Non-canonical - accommodating legacy and “tribal knowledge”
  - Non-standard semantics
  - Differentiating enterprise differences
  - Defining clarity beyond “personal abbreviations” (tmstss…)
  - Lost business meaning or worse business impact
Recommendations

- Continue with:
  - Attribute type coding for more meaning and origin of data
  - Governance and standardization of coding and uses
  - Advance more reliable and consistent coding and processes

- Advance with:
  - Coding knowledge for XPath and languages * tools that support
  - Working with MIRO and other vendors using to extract best practices
  - Assure AI can expand expertise to application areas (they use xsl today!)
  - Driving non-enterprise variations back into standard
Canonical Integration Only Works via a Process

Canonical Message (OAGIS) Moving to Canonical System

Canonical System is a journey – not single step

Progress report

- Points along the journey include:
  - Canonical message design – structure
  - Canonical message practical use – attributes
  - Canonical message reuse – consistency
  - Adaptation for application “fragility” and consistent reuse – adapters – in production
  - Consistent interpretation – in production
  - Canonical enterprise data – master data – in work
  - Data cleansing on the fly – in design
  - Enterprise interpretation utilities – in design