EDI, XML & JSON Implementation: beyond the basics

Matt Bemis
Associate Registrar
wbemis@usc.edu
USC University of Southern California

Jerry Bracken
Software Engineer
jeraldbracken@gmail.com
BYU BRIGHAM YOUNG UNIVERSITY

Doug Holmes
Acting Manager, eTranscripts
doug@ouac.on.ca
OUAC Ontario Universities' Application Centre

Alex Jackl
President & CEO
alex@bardicsystems.com
bardic systems

PESC

SPRING 2019 DATA SUMMIT
MAY 7 – 10, 2019 | WASHINGTON, D.C.

Overview

The focus of this workshop will be touch points beyond basic implementation of these three methods/payloads in the exchange of student transcript data. We have focused this content with the assumption that attendees understand the fundamental basics of EDI and XML data transmissions.

What we will be highlighting are the areas people generally don't talk about, or only learn about through obstacles that present themselves after implementation.

The panelist will provide their insights on how they've overcome these obstacles.

AGENDA

- GENERAL OBSERVATIONS
- EDI
- XML
- JSON

PESC

GENERAL OBSERVATIONS

- Educational Electronic Data Exchange Usage:
  - 94% of institutions receive some form of electronic transcripts (including PDF)
  - 81% of institutions send some form of electronic transcripts (including PDF)

GENERAL OBSERVATIONS

- Usage (continued):
  - 10-12 million exchanges a year of EDI or XML education documents (transcripts, requests, acknowledgements, test score, admission applications)
  - Roughly 15 million PDF transcript exchanges

GENERAL OBSERVATIONS

Consider the handling of:

- Electronic transcripts for students who have not yet applied
- Transactions from trading partners not yet established
- Data transmissions for students who applied but have not yet been admitted

GENERAL OBSERVATIONS

Consider the handling of:

- Duplicate data transmissions, so that coursework not *loaded* multiple times

- Double transcription of coursework so that coursework not *counted* multiple times
  - Transfer credits (same transcript)
  - Institutional agreements (multiple transcripts)

GENERAL OBSERVATIONS

Code values:

- Why crosswalking code sets is critical to the consumption of electronic transcript data:
  - Institutional identifiers (e.g., FICE, ACT, GEO Code)

GENERAL OBSERVATIONS

Reporting:

- Change of data instances for administrative review (where a grade has changed, units for the course have changed, etc.)
  - Recognizing true change versus duplicate transmission (e.g., document-level compare, field-by-field, etc)

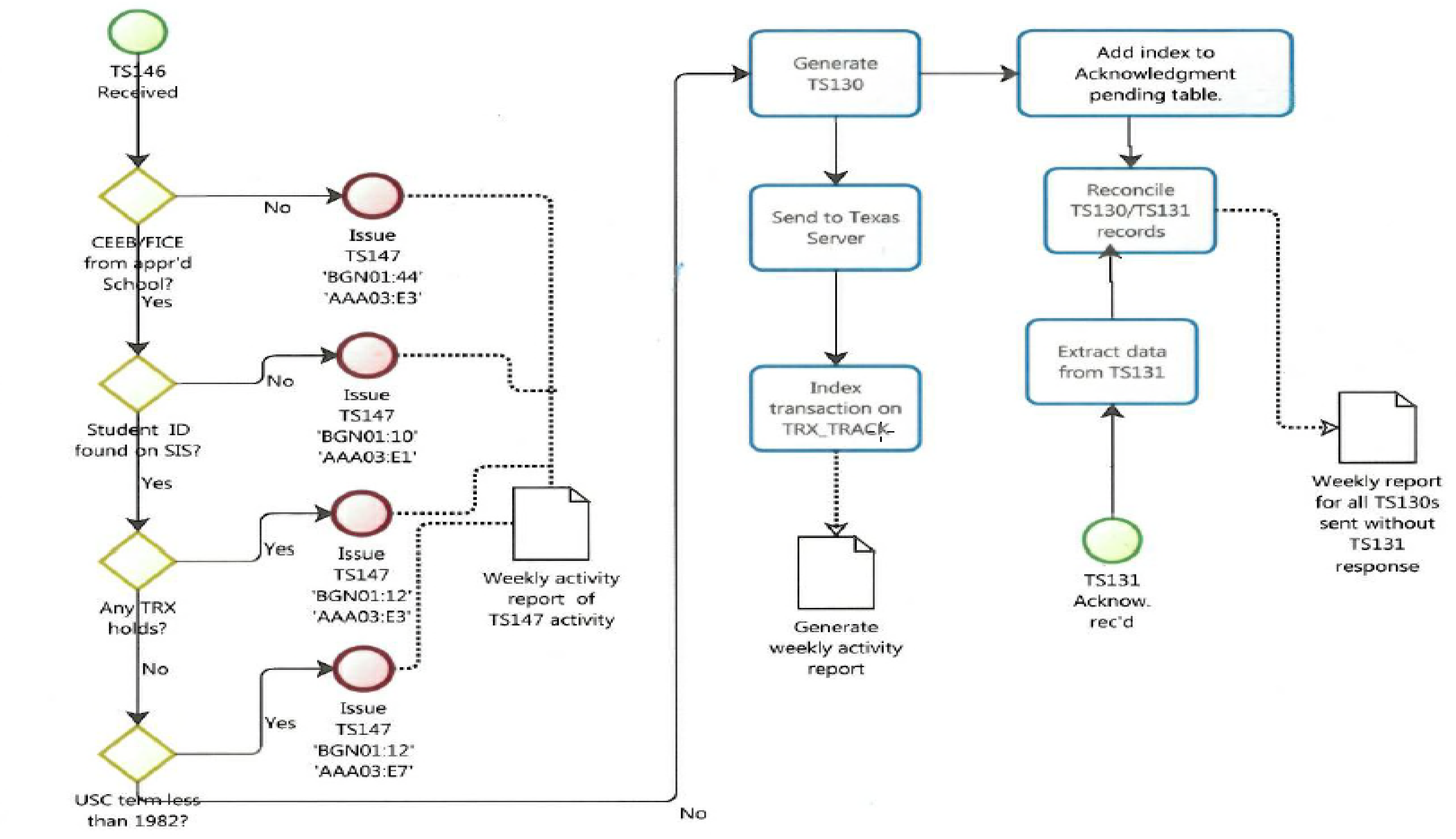
- Weekly/monthly reports for electronic transcript activity

GENERAL OBSERVATIONS

Requesting transcripts:

- Important to understand the "request for transcript" transaction sets, and available responses when the requests cannot be fulfilled

EDI Electronic Transcript fulfillment workflow


TS146 Received
CEEB/FICE from appr'd School?
No
Yes
Issue TS147 'BGN01:44' 'AAA03:E3'
Student ID found on SIS?
No
Yes
Issue TS147 'BGN01:10' 'AAA03:E1'
Any TRX holds?
Yes
No
Issue TS147 'BGN01:12' 'AAA03:E3'
USC term less than 1982?
Yes
No
Issue TS147 'BGN01:12' 'AAA03:E7'
Weekly activity report of TS147 activity
Generate TS130
Send to Texas Server
Index transaction on TRX_TRACK
Generate weekly activity report
Add index to Acknowledgment pending table.
Reconcile TS130/TS131 records
Extract data from TS131
TS131 Acknow. rec'd
Weekly report for all TS130s sent without TS131 response

GENERAL OBSERVATIONS

Matching transcripts:

- Unmatched, or partially matched, students
  - Request vs. transcript
  - Requested vs. unsolicited

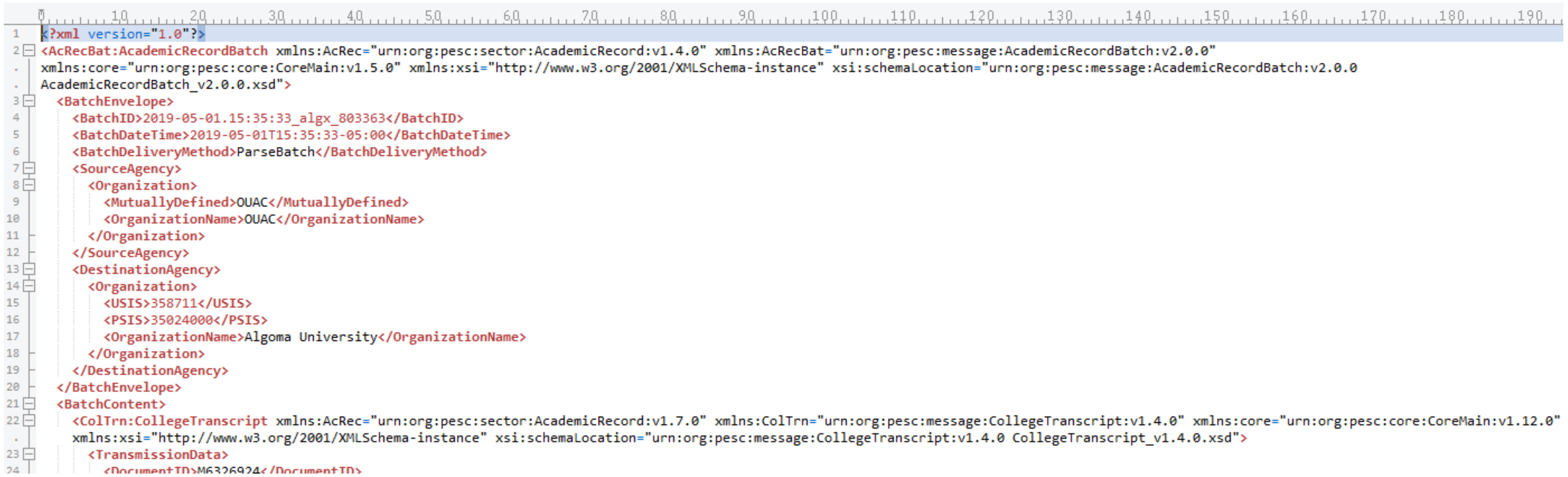
GENERAL OBSERVATIONS

Reconciling transcript acknowledgements:

- Reconciling outbound electronic transcript fulfillment, with inbound acknowledgment responses

XML

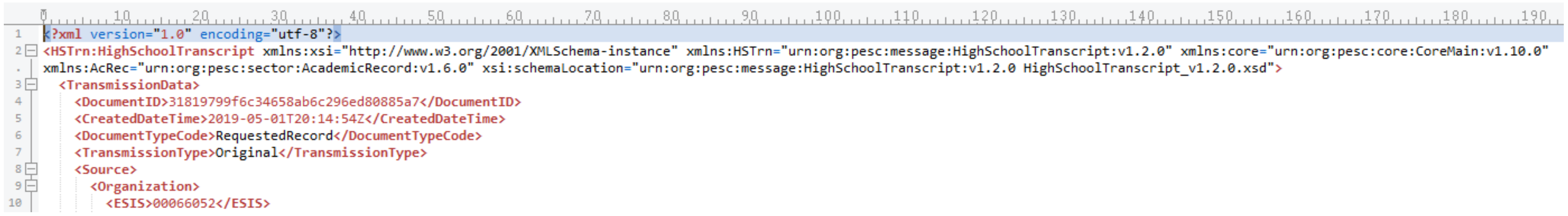
- **Batched documents?**



```
<?xml version="1.0"?>
<AcRecBat:AcademicRecordBatch xmlns:AcRec="urn:org:pesc:sector:AcademicRecord:v1.4.0" xmlns:AcRecBat="urn:org:pesc:message:AcademicRecordBatch:v2.0.0"
xmlns:core="urn:org:pesc:core:CoreMain:v1.5.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="urn:org:pesc:message:AcademicRecordBatch:v2.0.0
AcademicRecordBatch_v2.0.0.xsd">
  <BatchEnvelope>
    <BatchID>2019-05-01.15:35:33_algx_803363</BatchID>
    <BatchDateTime>2019-05-01T15:35:33-05:00</BatchDateTime>
    <BatchDeliveryMethod>ParseBatch</BatchDeliveryMethod>
    <SourceAgency>
      <Organization>
        <MutuallyDefined>OUAC</MutuallyDefined>
        <OrganizationName>OUAC</OrganizationName>
      </Organization>
    </SourceAgency>
    <DestinationAgency>
      <Organization>
        <USIS>358711</USIS>
        <PSIS>35024000</PSIS>
        <OrganizationName>Algoma University</OrganizationName>
      </Organization>
    </DestinationAgency>
  </BatchEnvelope>
  <BatchContent>
    <ColTrn:CollegeTranscript xmlns:AcRec="urn:org:pesc:sector:AcademicRecord:v1.7.0" xmlns:ColTrn="urn:org:pesc:message:CollegeTranscript:v1.4.0" xmlns:core="urn:org:pesc:core:CoreMain:v1.12.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="urn:org:pesc:message:CollegeTranscript:v1.4.0 CollegeTranscript_v1.4.0.xsd">
      <TransmissionData>
        <DocumentID>M6326924</DocumentID>
```

XML

- Individual documents?



```xml
<?xml version="1.0" encoding="utf-8"?>
<HSTrn:HighSchoolTranscript xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:HSTrn="urn:org:pesc:message:HighSchoolTranscript:v1.2.0" xmlns:core="urn:org:pesc:core:CoreMain:v1.10.0"
xmlns:AcRec="urn:org:pesc:sector:AcademicRecord:v1.6.0" xsi:schemaLocation="urn:org:pesc:message:HighSchoolTranscript:v1.2.0 HighSchoolTranscript_v1.2.0.xsd">
  <TransmissionData>
    <DocumentID>31819799f6c34658ab6c296ed80885a7</DocumentID>
    <CreatedDateTime>2019-05-01T20:14:54Z</CreatedDateTime>
    <DocumentTypeCode>RequestedRecord</DocumentTypeCode>
    <TransmissionType>Original</TransmissionType>
    <Source>
      <Organization>
        <ESIS>00066052</ESIS>
```

- Both?

XML

- Some validation tools/libraries cannot handle Batch and Documents using different Core and Sector libraries

XML

- Separate schemas for HS and PS
  - EDI had a single “template” for both

- Evolution of standards = multiple Core and Sector libraries (use all, or, edit xsd’s)

XML

- Schema validation:
  - on export?
  - on import?
  - at all? (e.g., impacts student in prod)
  - SIS capable, or, additional tools?

PESC

XML

- Enumerated lists
  - some code sets not “closed” (e.g., country, language)
  - ISO “snap shot” inside schema
  - historical codes may have been removed

XML

- Data lengths
  - e.g., AcademicAwardTitle (80 → 450 char)
- Data constructs
  - e.g., addresses, phone #'s
- Required fields
  - e.g., address country

PESC

XML

- Unicode!
  - Not handled consistently by all tools, databases, operating systems, IDE's, ERP's
  - Unexpected results exposing bad data (e.g., copy/paste from rich text)

XML

- Multiple academic records
- Infinite repeating tags (e.g., NoteMessage) may need some “practical” limit as they don’t always relate to rows in a database (e.g., might be a single field)

XML

- Course block
  - within a session
  - not tied to a session
- Slightly different in HS, PS and Application

XML

- User-Defined Extension (UDE)
  - not highly used
  - non-standard naming can add complexity
  - additional “bulk” in the payload

XML

- Version control

- Not all trading partners will be able to upgrade versions together … can impact Admissions when receiving a newer version

XML

- Automating GPA calculations
  - Session names are variable (text)
  - Session dates are optional
  - Course dates are optional

Gosh- isn’t XML just fine?

JSON is just another way to annotate data so that it can become information- just like XML and EDI.   So the business use cases are identical.   The differences are technical and implementation specific.

Both **XML and JSON** are in widespread use today. **EDI** is used in narrow verticals and losing support. All are **used as** data interchange formats and both have been adopted by applications as a way to store structured data. In addition, because **XML** is a document markup language, it is also **used by** document-based applications as a way to store documents.

Technical Description (1 of 3):

A JavaScript object is an associative array, or dictionary, of zero or more pairs of property names and associated JSON values. A JSON object is a JavaScript object literal. It is written as such a property list enclosed in braces ({, }), with name-value pairs separated by commas (,), and with the name and value of each pair separated by a colon (:). In JSON each property name and each string value must be enclosed in double quotation marks (").

Technical Description (2 of 3):

In JavaScript notation, a property name used in an object literal can be, but need not be, enclosed in double quotation marks. It can also be enclosed in single quotation marks ('). As a result of this difference, in practice, data that is represented using unquoted or single-quoted property names is sometimes referred to loosely as being represented in JSON, as the “lax syntax”

Technical Description (3 of 3):

A string in JSON is composed of Unicode characters, with backslash (\) escaping. A JSON number (numeral) is represented in decimal notation, possibly signed and possibly including a decimal exponent. An object property is often called a field. An object property name-value pair is often called an object member. Order is not significant among object members.

Advantages of JSON:

1. Light-weight Syntax (faster and easier processing)
2. Human-readable (Not as human readable as XML but more human readable than EDI or code)
3. Designed for light-weight applications on the web (“apps”)
4. Quicker to construct as there are fewer formal rules.

Disadvantages of JSON:

1. Missing some of the enterprise complexity of XML (attributes, name spacing, and metadata)
2. More immature tools and tools are more non-standardized
3. Schema models are just now converging. No W3C Standard for Schema.
4. Fewer formal rules.

JSON-LD makes up for a lot of that gap with context and Linked Open Data.

Issues with JSON:

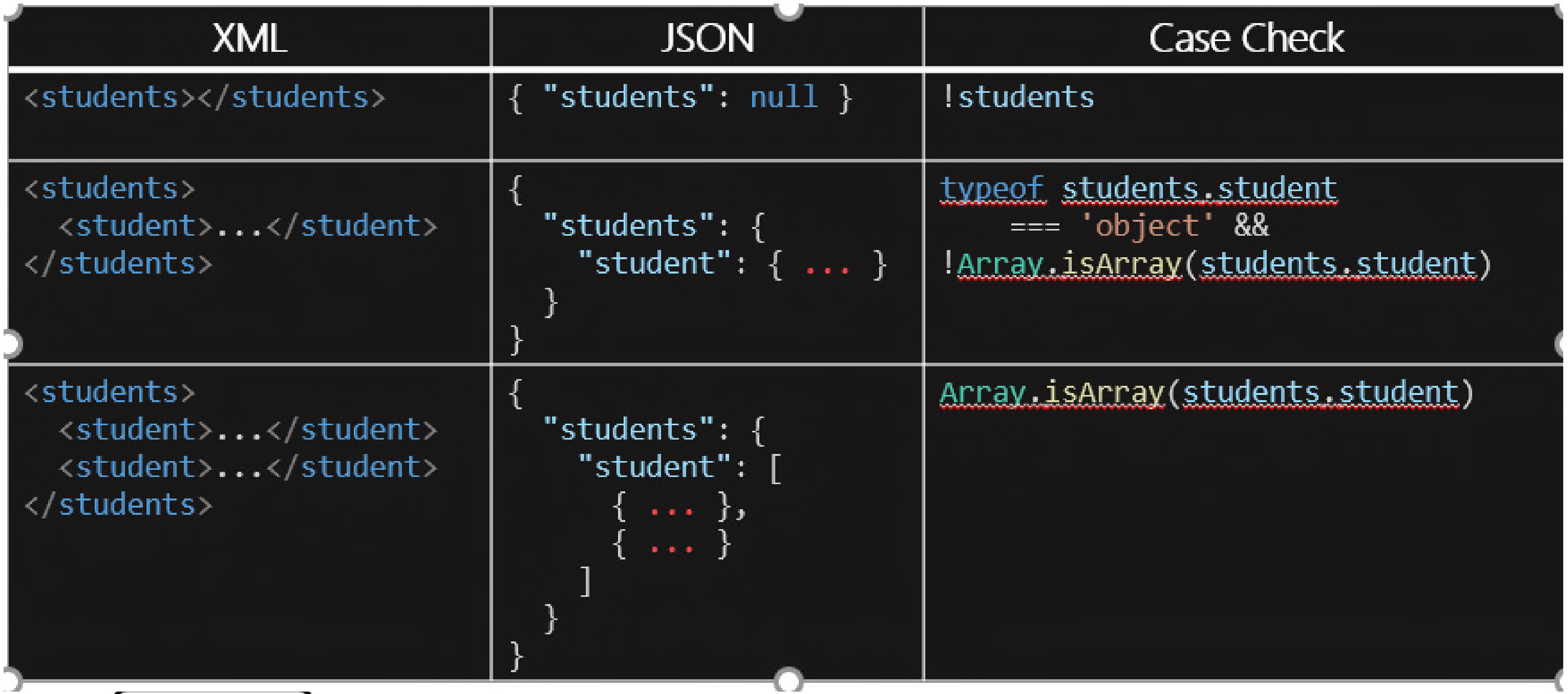
1. Mapping difficulties: Multiple solutions

```
<e attr="value"/>
<e>text</e>
<e attr="value">text</e>
<e><b>text</b></e>
<e><b>text</b><b>text</b></e>
```

2. Security Standards are not defined (https only)

Issues with JSON:

3. Heterogeneous lists
    - I.e. lists with more than one type of content
    - Define rule for Data Models to exclude this structure
4. Mixed / arbitrary content
    - i.e. HTML formatted text
    - Content would have to be escaped to a single string
5. Order of content
    - JSON properties are unordered, XML elements are ordered (i.e. sequence)



XML	JSON	Case Check
`<students></students>`	`{ "students": null }`	`!students`

`<students>`
`  <student>...</student>`
`</students>`
	`{`
	`  "students": {`
	`    "student": { ... }`
	`  }`
	`}`
		`typeof students.student`
		`    === 'object' &&`
		`!Array.isArray(students.student)`

`<students>`
`  <student>...</student>`
`  <student>...</student>`
`</students>`
	`{`
	`  "students": {`
	`    "student": [`
	`      { ... },`
	`      { ... }`
	`    ]`
	`  }`
	`}`
		`Array.isArray(students.student)`

THANK YOU!

QUESTIONS & ANSWERS