“The Impossible Task of Comparing CALS Tables”

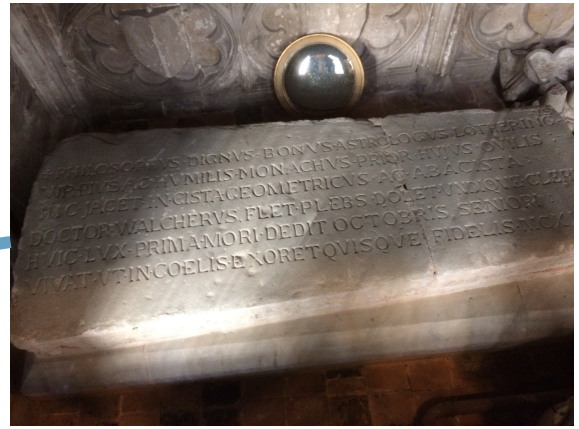
DeltaXML,

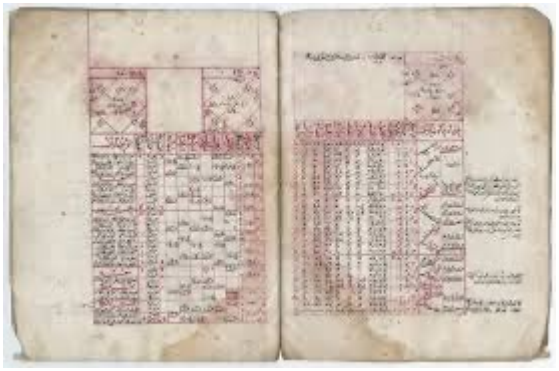
Robin La Fontaine

John Francis

DELTAXML

Tables go back a long way





"In this chest lies Doctor Walcher, a worthy philosopher, a good astronomer, a Lotharingian, a pious and humble man, a monk, the prior of his sheepfold, a geometer and abacist. The people mourn, the clergy grieve on all sides..." D.1125

DELTAXML

What will we cover today?

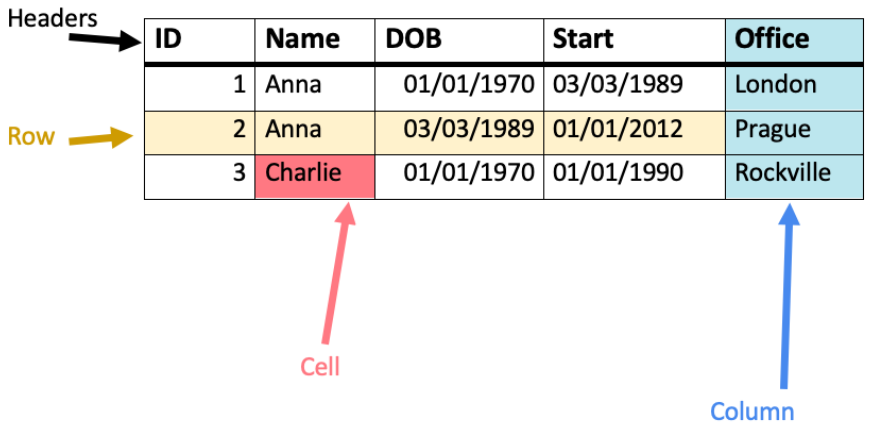
1. XML structure for CALS tables
2. Conventional approach: compare XML and manipulate delta file
3. Content based comparison: Make it simpler then complicate it later
4. The mixed blessing of user controls
5. Conclusions

CALS Tables – what does the XML look like?

```
<table frame="all">
  <title>A sample table</title>
  <tgroup cols="3">
    <thead>
      <row>
        <entry>Header 1</entry>
        <entry>Header 2</entry>
        <entry>Header 3</entry>
      </row>
    </thead>
    <tbody>
      <row>
        <entry>Row 1 Cell 1</entry>
        <entry>Row 1 Cell 2</entry>
        <entry>Row 1 Cell 3</entry>
      </row> <row>
        <entry>Row 2 Cell 1</entry>
        <entry>Row 2 Cell 2</entry>
        <entry>Row 2 Cell 3</entry>
      </row>
    </tbody>
  </tgroup>
</table>
```

Table 1. A sample table

Header 1	Header 2	Header 3
Row 1 Cell 1	Row 1 Cell 2	Row 1 Cell 3
Row 2 Cell 1	Row 2 Cell 2	Row 2 Cell 3



DELTAXML

CALS Tables – spans

```
<table frame="all">
  <title>A sample table</title>
  <tgroup cols="3">
    <colspec colname="c1"/>
    <colspec colname="c2"/>
    <colspec colname="c3"/>
    <thead>...</thead>
    <tbody>
      <row>
        <entry namest="c1" nameend="c2"
morerows="1"
          >A Span across 2 Columns and 2
Rows</entry>
        <entry colname="c3">Row 1 Cell
3</entry>
      </row>
      <row>
        <entry>Row 2 Cell 3</entry>
      </row>
    </tbody>
  </tgroup>
</table>
```

We can also use **colnames** on **entrys** to specify the absolute column an entry belongs to.

For vertical spans we just use a **morerows** attribute on a starting cell.

Table 2. A sample table

Header 1	Header 2	Header 3
A Span across 2 Columns and 2 Rows		Row 1 Cell 3
		Row 2 Cell 3

Notice ‘overhangs’ from the preceding rows: there is only one cell specified in the second row of the tbody.

DELTAXML

CALS Tables – compare the XML

A version	B version	Delta: A and B showing changes
<row> <entry>Row 1 Cell 1</entry> <entry>Row 1 Cell 2</entry> <entry>Row 1 Cell 3</entry> </row>	<row> <entry>Row 1 Cell 1</entry> <entry>ROW 1 CELL 2</entry> </row>	<row delta="A!=B"> <entry delta="A=B">Row 1 Cell 1</entry> <entry delta="A!=B"> <dx:t delta="A">Row 1 Cell 2</dx:t> <dx:t delta="B">ROW 1 CELL 2</dx:t></entry> <entry delta="A">Row 1 Cell 3</entry> </row>

DELTAXML

CALS Tables – compare the XML

First Version

Sr	Name	Email	DOB
1	Joe	joe@gmail.com	01/01/1985
2	Anna	anna@gmail.com	02/02/1985
3	Chris	chris@gmail.com	03/03/1985
4	Dave	dave@gmail.com	04/04/1985

Second Version

Sr	Name	DOB	Email
1	Joe	01/01/1985	joe@gmail.com
2	Anna	02/02/1985	anna@gmail.com
3	Chris	03/03/1985	chris@gmail.com
4	Dave	04/04/1985	dave@gmail.com

Sr	Name	~~Email~~ *DOB*	~~DOB~~ *Email*
1	Joe	~~joe@gmail.com~~ *01/01/1985*	~~01/01/1985~~ *joe@gmail.com*
2	Anna	~~anna@gmail.com~~ *02/02/1985*	~~02/02/1985~~ *anna@gmail.com*
3	Chris	~~chris@gmail.com~~ *03/03/1985*	~~03/03/1985~~ *chris@gmail.com*
4	Dave	~~dave@gmail.com~~ *04/04/1985*	~~04/04/1985~~ *dave@gmail.com*

DELTAXML

How to approach an impossible problem?

Step 1: Ignore something to make it a simpler problem

Step 2: Solve the simpler problem

Step 3: Add back what has been ignored to get an improved result

Building a content-based approach

1. First, regularize the tables (this involves some complex processing!)

   Make into a rectangular grid

   Duplicate the span contents across cells but remember them for later

2. Next, align the columns based on content

   deltaxml:table-column-alignment="A|1=B|1, A|2=B|2, B|3, A|3=B|4, A|4”

3. Key the cells in each row based on column alignment

4. Compare the table rows using the column alignment

5. Restore the spans based on B priority

Content based comparison results: Spans

First Version 'A'

Region	Manager	Local Rep	Secretary
North	Adam	Callum	Dave
Southwest	Annie	Clive	Doreen
Southeast	Ant	Cecilia	Danny
East	Ash		
West			

Second Version 'B'

Region	Manager	Local Rep	Secretary
North	Adam	Callum	Dave
Southwest	Annie		Doreen
Southeast			Danny

Region	Manager	Local Rep	Secretary
North	Adam	Callum	Dave
Southwest	Annie ~~Clive Ant~~ ~~Cecilia~~		Doreen
Southeast			Danny
East	Ash		
West			

DELTAXML

Content based comparison results: Columns

First Version

ID	Name	Start	Email	Office
1	Anna	03/03/1989	anna.a@work.com	London
2	Anna	10/01/2012	annie.b@work.com	Rockville
3	Charlie	01/01/1990	charles.c@work.com	Prague

Second Version

ID	Name	Email	Start	Office
1	Anna	anna.a@work.com	03/03/1989	London
2	Anna	annie.b@dev.work.com	10/01/2012	Rockville
3	Charlie	charles.c@work.com	01/01/1990	Prague

Column order change shown

ID	Name	Email	Start	Email	Office
1	Anna	anna.a@work.com	03/03/1989	anna.a@work.com	London
2	Anna	annie.b@dev.work.com	10/01/2012	annie.b@work.com	Rockville
3	Charlie	charles.c@work.com	01/01/1990	charles.c@work.com	Prague

Column order ignored (orderless)

ID	Name	Email	Start	Office
1	Anna	anna.a@work.com	03/03/1989	London
2	Anna	annie.b@*dev*.work.com	10/01/2012	Rockville
3	Charlie	charles.c@work.com	01/01/1990	Prague

DELTAXML

Content based comparison results: Column and span

Version 'A'

Example 18: Column inserted, Row deleted in middle of Span

Region	Manager	Local Rep	Secretary
North	Adam	Callum	Diane
South West	Annie		Dodie
South East Upper			Dylan
South East Lower			Dahlia
East	Ash	Cecilia	Dave
West	Art	Colin	Danny

Version 'B'

Example 18: Column inserted, Row deleted in middle of Span

Region	Manager	Coordinator	Local Rep	Secretary
North	Adam	Beth	Callum	Diane
South West	Annie			Dodie
South East				Dahlia
East	Ash	Bill	Cecilia	Dave
West	Art	Bruce	Colin	Danny

Old

Example 18: Column inserted, Row deleted in middle of Span

Region	Manager	Local Rep	Secretary
North	Adam	Callum	Diane
South West	Annie		Dodie
South East Upper			Dylan
South East Lower			Dahlia
East	Ash	Cecilia	Dave
West	Art	Colin	Danny

Region	Manager	Coordinator	Local Rep	Secretary
North	Adam	Beth	Callum	Diane
South West	Annie			Dodie
South East				Dahlia
East	Ash	Bill	Cecilia	Dave
West	Art	Bruce	Colin	Danny

New

Example 18: Column inserted, Row deleted in middle of Span

Region	Manager	Coordinator	Local Rep	Secretary
North	Adam	Beth	Callum	Diane
South West	Annie			Dodie
South East Upper				Dylan
South East Lower				Dahlia
East	Ash	Bill	Cecilia	Dave
West	Art	Bruce	Colin	Danny

DELTAXML

Content based comparison difficulties and TODO

A

Problems with short wide tables with many hidden columns

I/O Type 1		I/O Type 2					
1	Dis-abled	3.2	4	3	2	2	2

```
<table frame="all" rowsep="1" colsep="1" id="table_pj4_tfl_1tb">
    <title>Problems with short wide tables with many hidden columns</title>
    <tgroup cols="20">
        <colspec colname="c1" colnum="1" colwidth="5*"/>
        <colspec colname="c2" colnum="2" colwidth="5*"/>
        <colspec colname="c3" colnum="3" colwidth="5*"/>
        <colspec colname="c4" colnum="4" colwidth="5*"/>
        <colspec colname="c5" colnum="5" colwidth="5*"/>
        <colspec colname="c6" colnum="6" colwidth="5*"/>
        <colspec colname="c7" colnum="7" colwidth="5*"/>
        <colspec colname="c8" colnum="8" colwidth="5*"/>
        <colspec colname="c9" colnum="9" colwidth="5*"/>
        <colspec colname="c10" colnum="10" colwidth="5*"/>
        <colspec colname="c11" colnum="11" colwidth="5*"/>
        <colspec colname="c12" colnum="12" colwidth="5*"/>
        <colspec colname="c13" colnum="13" colwidth="5*"/>
        <colspec colname="c14" colnum="14" colwidth="5*"/>
        <colspec colname="c15" colnum="15" colwidth="5*"/>
        <colspec colname="c16" colnum="16" colwidth="5*"/>
        <colspec colname="c17" colnum="17" colwidth="5*"/>
        <colspec colname="c18" colnum="18" colwidth="5*"/>
        <colspec colname="c19" colnum="19" colwidth="5*"/>
        <colspec colname="c20" colnum="20" colwidth="5*"/>
        <thead>
            <row>
                <entry namest="c1" nameend="c10">I/O Type 1 </entry>
                <entry namest="c11" nameend="c20">I/O Type 2</entry>
            </row>
        </thead>
        <tbody>
            <row>
                <entry>1</entry>
                <entry namest="c2" nameend="c10">Dis-abled</entry>
                <entry namest="c11" nameend="c15">3.2</entry>
                <entry>4</entry>
                <entry>3</entry>
                <entry>2</entry>
                <entry>2</entry>
                <entry>2</entry>
            </row>
        </tbody>
    </tgroup>
</table>
```

B

Problems with short wide tables with many hidden columns

I/O Type 1		I/O Type 2					
1	Disabled	3.2	4	3	2	2	2

```
<table frame="all" rowsep="1" colsep="1" id="table_pj4_tfl_1tb">
    <title>Problems with short wide tables with many hidden columns</ti
    <tgroup cols="20">
        <colspec colname="c1" colnum="1" colwidth="5*"/>
        <colspec colname="c2" colnum="2" colwidth="5*"/>
        <colspec colname="c3" colnum="3" colwidth="5*"/>
        <colspec colname="c4" colnum="4" colwidth="5*"/>
        <colspec colname="c5" colnum="5" colwidth="5*"/>
        <colspec colname="c6" colnum="6" colwidth="5*"/>
        <colspec colname="c7" colnum="7" colwidth="5*"/>
        <colspec colname="c8" colnum="8" colwidth="5*"/>
        <colspec colname="c9" colnum="9" colwidth="5*"/>
        <colspec colname="c10" colnum="10" colwidth="5*"/>
        <colspec colname="c11" colnum="11" colwidth="5*"/>
        <colspec colname="c12" colnum="12" colwidth="5*"/>
        <colspec colname="c13" colnum="13" colwidth="5*"/>
        <colspec colname="c14" colnum="14" colwidth="5*"/>
        <colspec colname="c15" colnum="15" colwidth="5*"/>
        <colspec colname="c16" colnum="16" colwidth="5*"/>
        <colspec colname="c17" colnum="17" colwidth="5*"/>
        <colspec colname="c18" colnum="18" colwidth="5*"/>
        <colspec colname="c19" colnum="19" colwidth="5*"/>
        <colspec colname="c20" colnum="20" colwidth="5*"/>
        <thead>
            <row>
                <entry namest="c1" nameend="c10">I/O Type 1 </entry>
                <entry namest="c11" nameend="c20">I/O Type 2</entry>
            </row>
        </thead>
        <tbody>
            <row>
                <entry>1</entry>
                <entry namest="c2" nameend="c10">Disabled</entry>
                <entry namest="c11" nameend="c15">3.2</entry>
                <entry>4</entry>
                <entry>3</entry>
                <entry>2</entry>
                <entry>2</entry>
                <entry>2</entry>
            </row>
        </tbody>
    </tgroup>
</table>
```

Problems with short wide tables with many hidden columns

I/O Type 1		I/O Type 1		I/O Type 2						I/O Type 2		
1	Disabled	1	Dis-abled	3.2	4	3	2	2	2	2	2	2

Problems with short wide tables with many hidden columns

I/O Type 1		I/O Type 2					
1	Dis-abledDisabled	3.2	4	3	2	2	2

DELTAXML

What does a user really want?

1. To see changes to the values of cells wherever possible.
2. The result should contain valid table markup which can be rendered.
3. Not to have to spend lots of effort saying what type of table they are using, at least to begin with.
4. To have the difference be more robust than focusing just on the structure of the underlying markup.
5. Don’t lie!

User controls: mixed blessing

What controls should we include for users?

- Lots of control leads to complexity in code, documentation and testing
- Good default choices are ideal: “It should just work the way I expect it to”

Necessary controls include:

- Controlling whether or not columns are to be treated as ordered
- Adding keys to columns to control alignment
- Adding Processing Instructions (PI) to control alignment by colspec/colnum

Standards, committees and camels

We learn a few lessons about what makes a ‘good’ standard

- Preferably only **one** way to represent something
  - Tends not to happen when there is a committee involved!
  - However, committees are very good at making standards robust and well-defined
- Avoid shorthands
  - A computer can easily generate the full version and it is always easier to read the full version back in

… and thanks to Norm Tovey-Walsh for help with the CALS standard

In conclusion…

- Comparison of CALS tables has provided us with a challenge
- Comparing the native XML has two surprises:
  - How well it works much of the time
  - How easily it is tripped up
- Content based approach seems to be a significant impovement
  - Some user controls are needed but can be kept to a minimum
- If you are involved in standards, please avoid shorthands and multiple ways to represent the same concept

DELTAXML