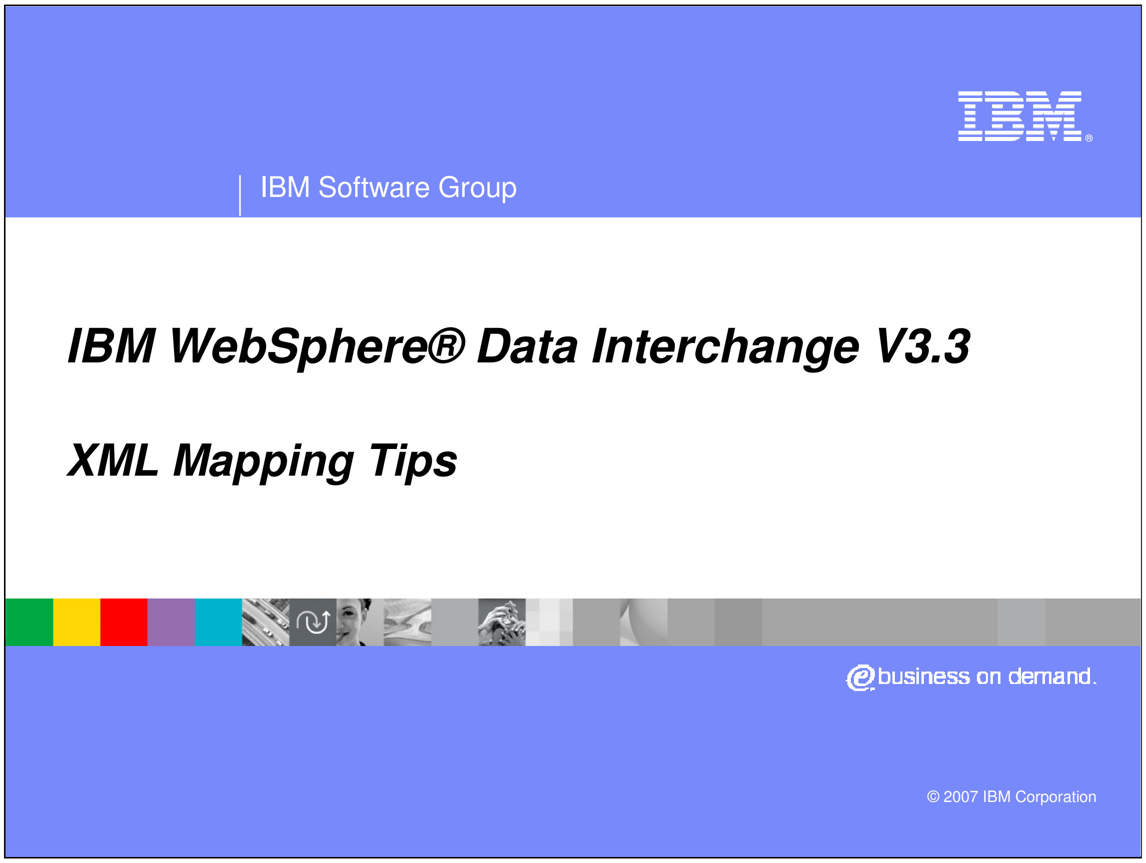


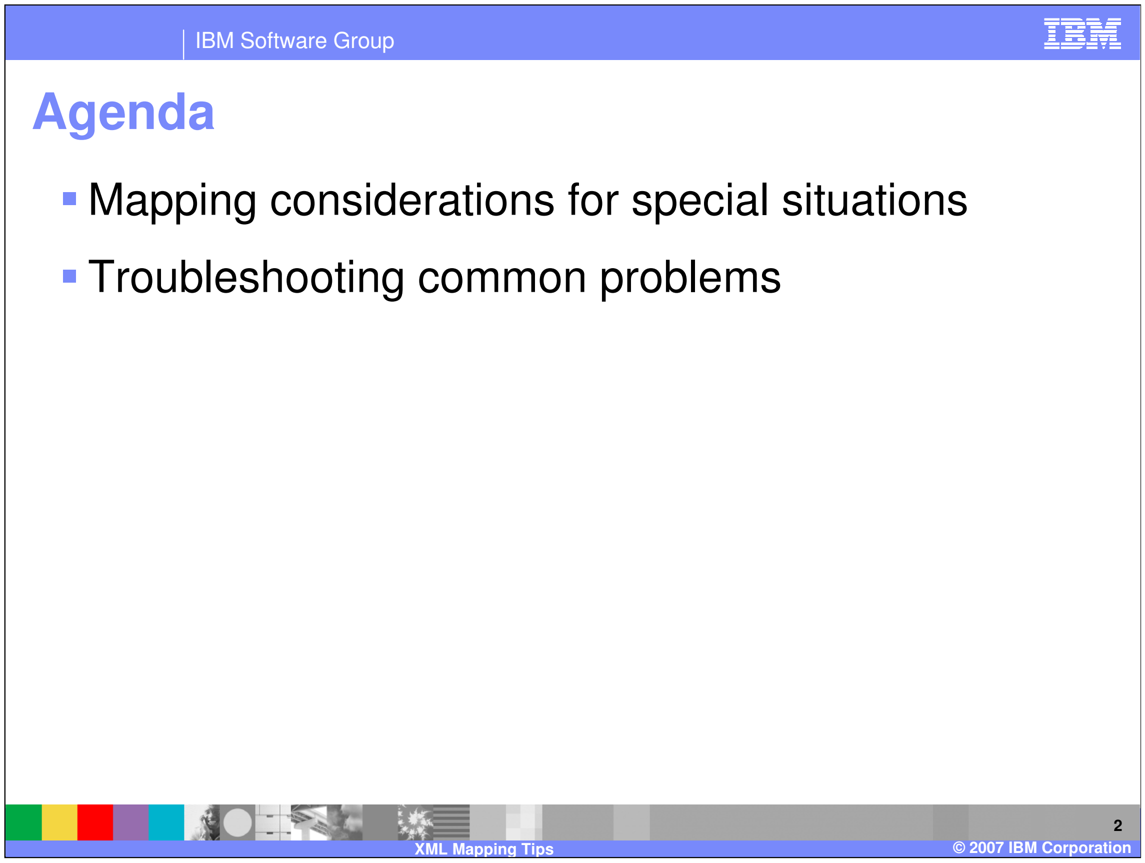
IBM Software Group

IBM WebSphere® Data Interchange V3.3

XML Mapping Tips

business on demand.

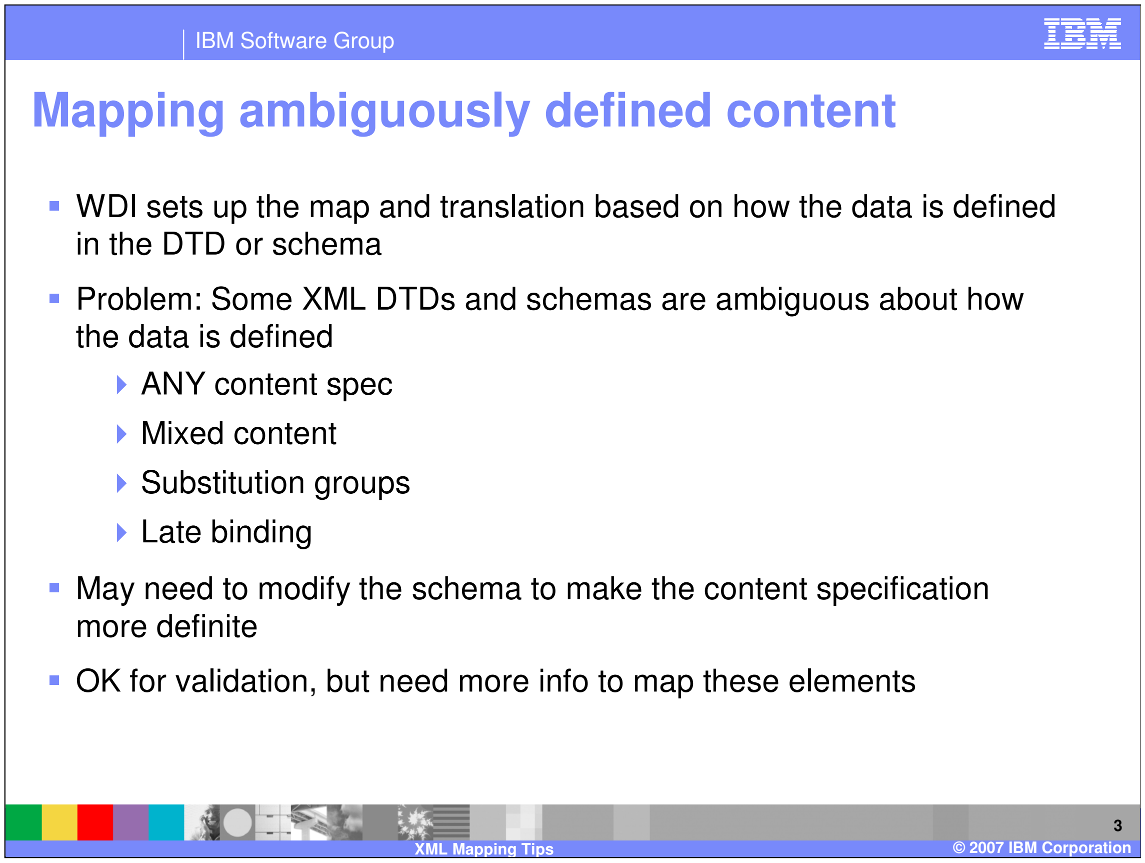
This presentation will review some XML Mapping tips.



Agenda

- Mapping considerations for special situations
- Troubleshooting common problems

The presentation shows hints and tips for mapping and transforming XML data. This includes some special XML DTD and schema constructs, and how to handle them. It also describes some common errors that are encountered, and provides typical solutions to resolve them.


IBM Software Group

Mapping ambiguously defined content

- WDI sets up the map and translation based on how the data is defined in the DTD or schema
- Problem: Some XML DTDs and schemas are ambiguous about how the data is defined
  - ANY content spec
  - Mixed content
  - Substitution groups
  - Late binding
- May need to modify the schema to make the content specification more definite
- OK for validation, but need more info to map these elements

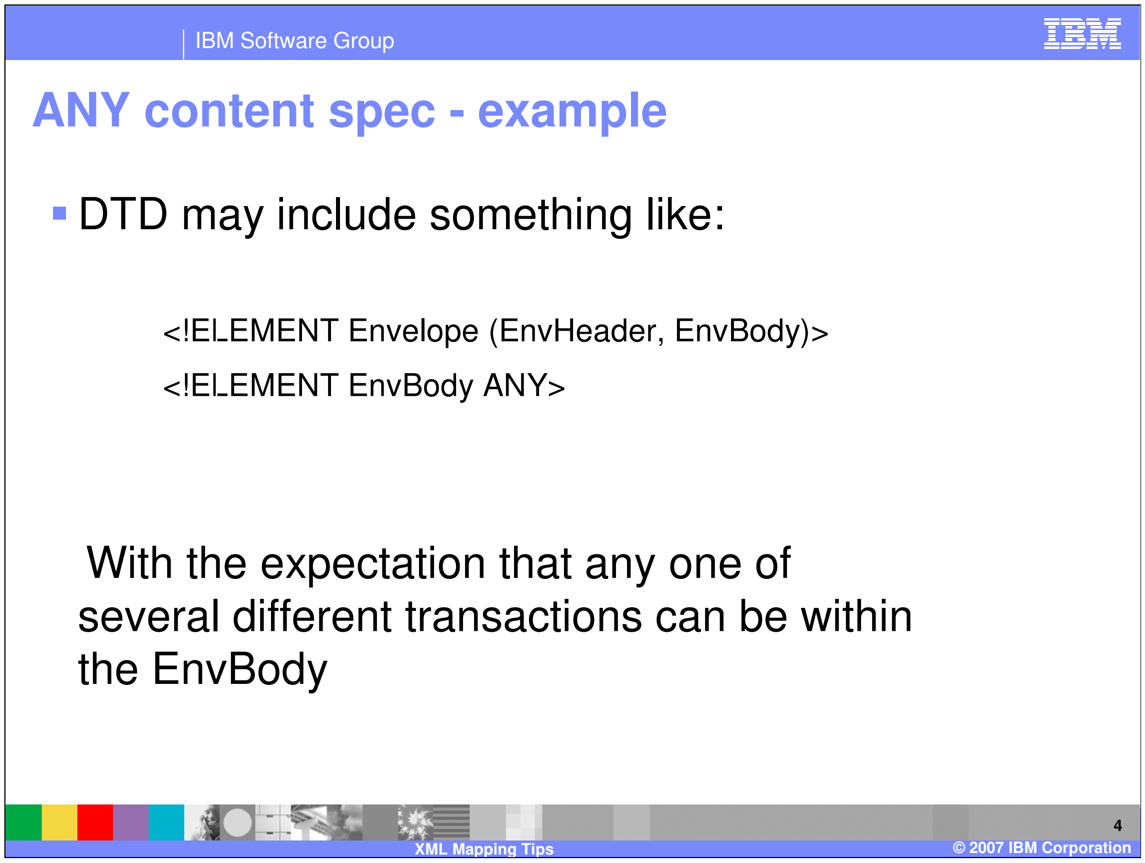
3

XML Mapping Tips

© 2007 IBM Corporation

Sometimes XML DTDs and schemas define the data very ambiguously. That is, they do not define the structure of the document very precisely, but allow a lot of freedom for the document structure. Some examples include the ANY content spec, mixed content, substitution groups, and late binding.

However, WebSphere Data Interchange (WDI) defines the map and translation based on the document definition provided by the DTD or the schema – not based on a particular instance of the document. If the DTD or schema does not provide enough information about the document structure, this can make it difficult or impossible to map these DTDs and schemas without modification. In these cases, the DTD or schema may be ok to use for validation, but may need to be modified to use it for mapping.



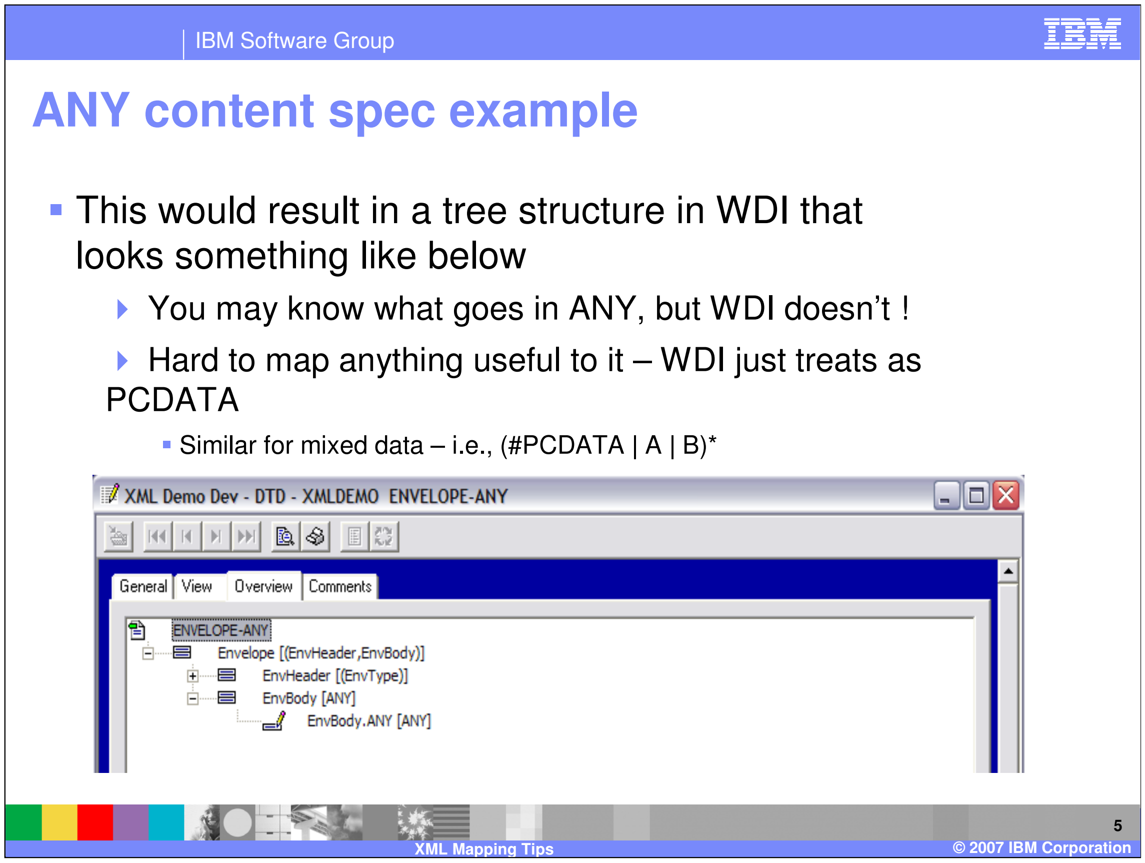
ANY content spec - example

- DTD may include something like:

```
<!ELEMENT Envelope (EnvHeader, EnvBody)>
<!ELEMENT EnvBody ANY>
```

With the expectation that any one of several different transactions can be within the EnvBody

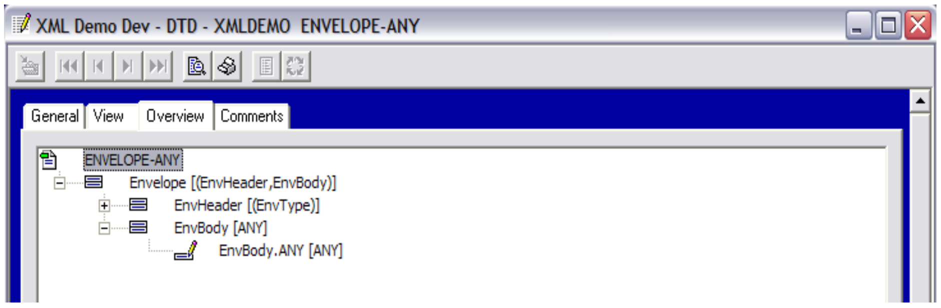
Here is an example of the ANY content specification. The EnvBody element is defined as ANY, with the expectation that any type of transaction could be contained within the EnvBody.



IBM Software Group

ANY content spec example

- This would result in a tree structure in WDI that looks something like below
  - You may know what goes in ANY, but WDI doesn't !
  - Hard to map anything useful to it – WDI just treats as PCDATA
    - Similar for mixed data – i.e., (#PCDATA | A | B)*



XML Demo Dev - DTD - XMLDEMO ENVELOPE-ANY

```
ENVELOPE-ANY
  Envelope [(EnvHeader,EnvBody)]
    EnvHeader [(EnvType)]
    EnvBody [ANY]
      EnvBody.ANY [ANY]
```

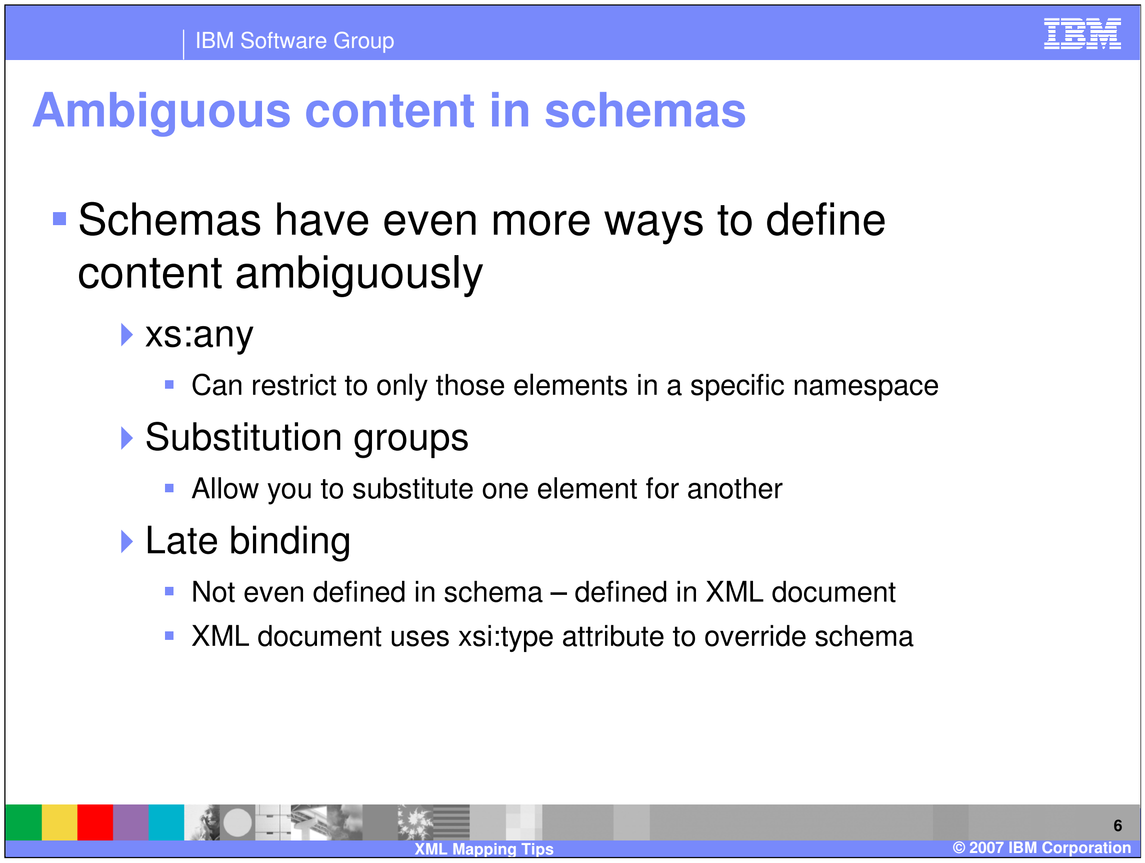
5

XML Mapping Tips

© 2007 IBM Corporation

This does not provide WDI enough information to create a useful mapping tree. You may know what goes there, but WDI does not!

Since WDI does not have enough information to map it, it just treats EnvBody like a PCDATA element – allowing you to map text data within it, but not elements. WDI handles mixed data the same way.



Ambiguous content in schemas

- Schemas have even more ways to define content ambiguously
  - xs:any
    - Can restrict to only those elements in a specific namespace
  - Substitution groups
    - Allow you to substitute one element for another
  - Late binding
    - Not even defined in schema – defined in XML document
    - XML document uses xsi:type attribute to override schema

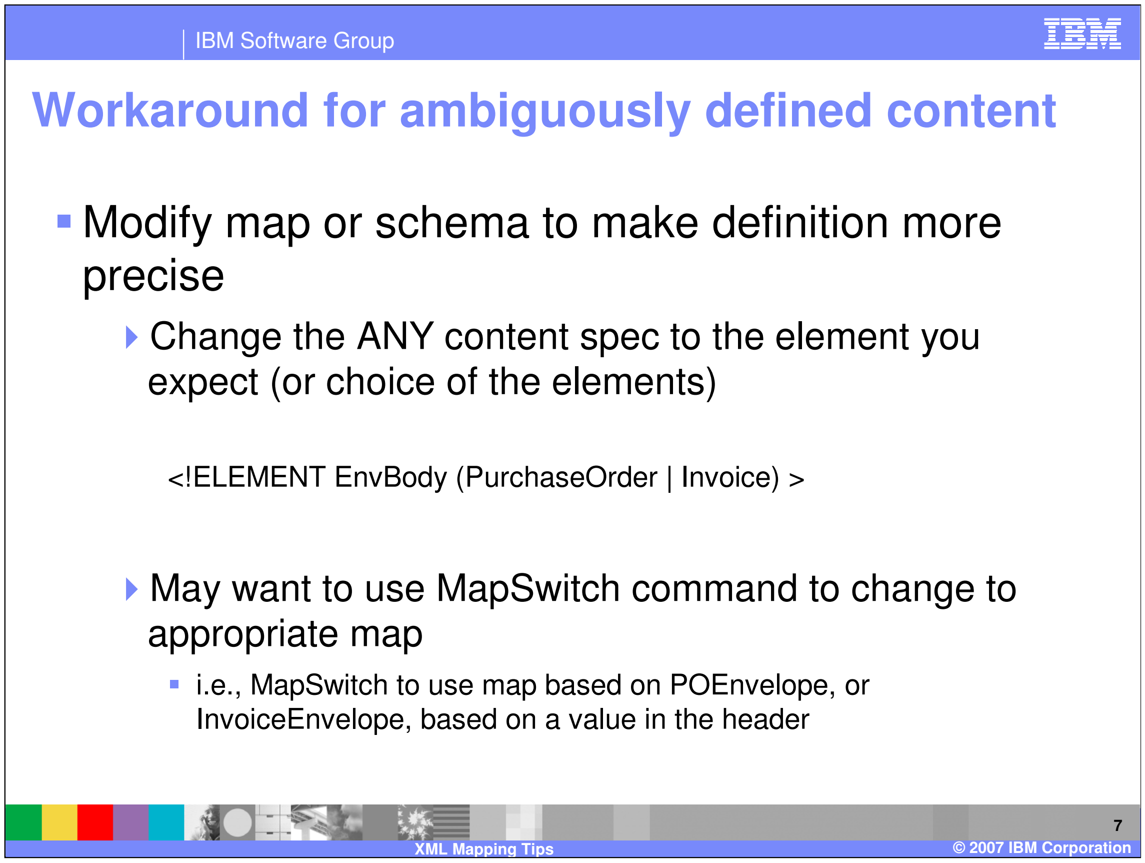
Schemas allow even more ways to make the content ambiguous.

•xs:any is similar to the ANY definition in DTDs, although schemas allow you to restrict the elements to a particular namespace

•Substitution groups allow you to substitute one element for another

•Late binding provides a way to define the element or override the schema definition in the XML document

These do not provide enough information to WDI to map the element, and may require modifications to the schema if you want to map these elements.



IBM Software Group

Workaround for ambiguously defined content

- Modify map or schema to make definition more precise
  - Change the ANY content spec to the element you expect (or choice of the elements)

    <!ELEMENT EnvBody (PurchaseOrder | Invoice) >

  - May want to use MapSwitch command to change to appropriate map
    - i.e., MapSwitch to use map based on POEnvelope, or InvoiceEnvelope, based on a value in the header

7

XML Mapping Tips

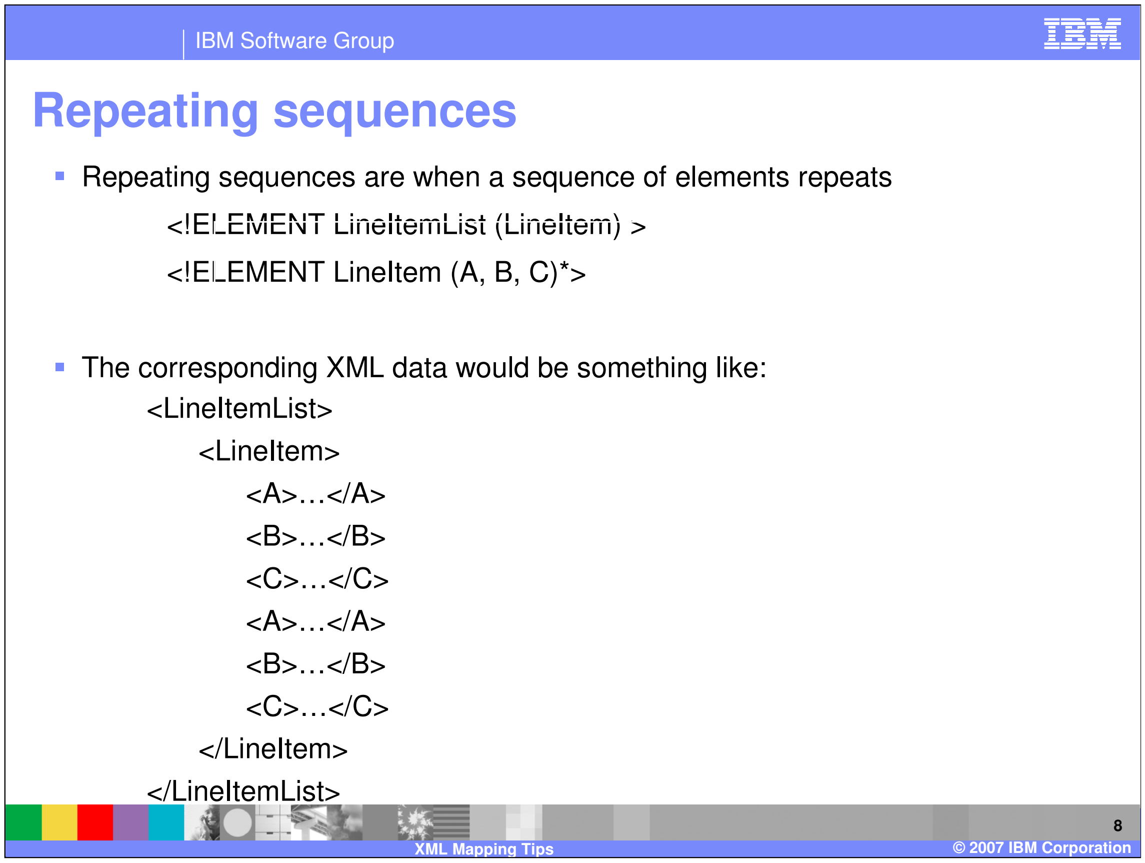
© 2007 IBM Corporation

You can modify the DTD or schema to make the definition more precise, then create the map using the modified version.

In this example, the EnvBody element was changed so instead of ANY, it contains either a PurchaseOrder or Invoice element. Then the PurchaseOrder and Invoice elements should have enough information that they can be mapped.

Another option is to create two separate DTDs or schemas – one where the PurchaseOrder is the only element in EnvBody, and another where the Invoice is the only element in EnvBody. Then a separate map could be created for each DTD or schema, and you could use a “switching map” that uses the MapSwitch command to determine which map should do the transformation. This has the advantage of isolating the mapping logic for the PurchaseOrder from the mapping logic for the Invoice.

Of course either of these techniques can be extended to more than two elements.



Repeating sequences

- Repeating sequences are when a sequence of elements repeats

```
<!ELEMENT LineItemList (LineItem) >
<!ELEMENT LineItem (A, B, C)*>
```

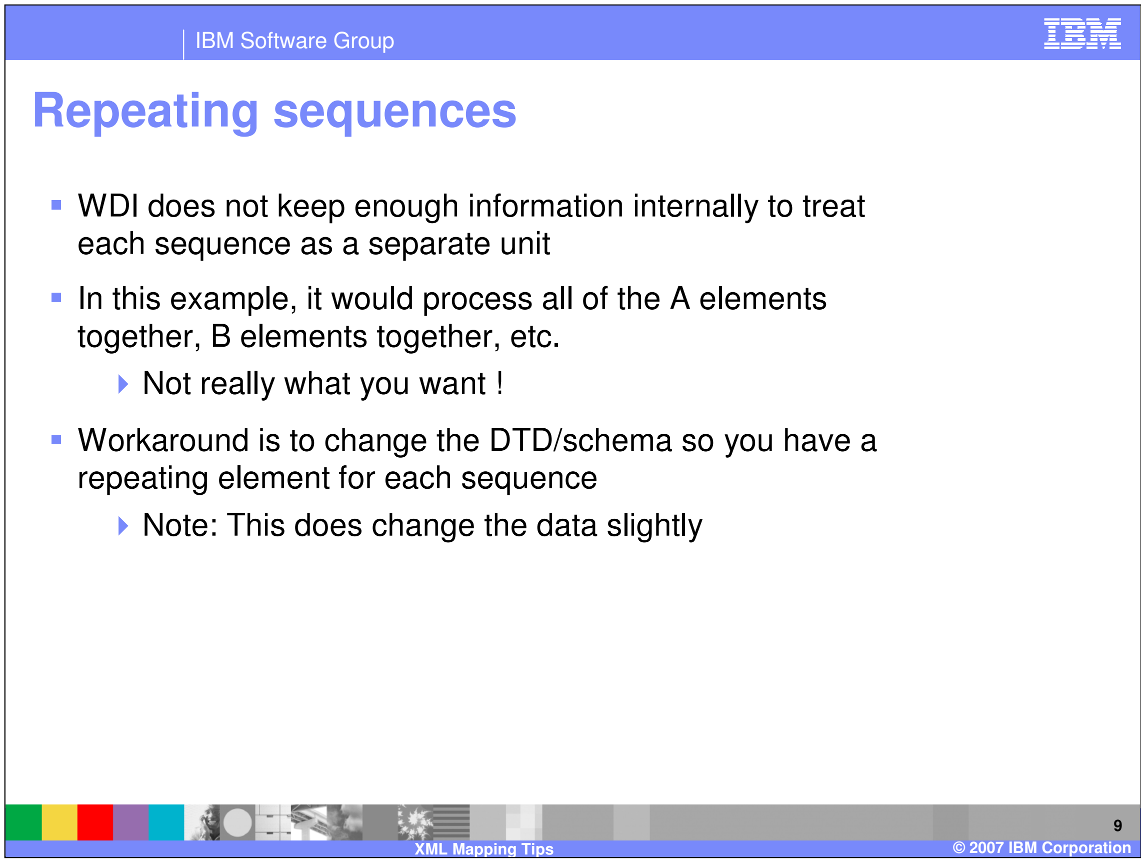
- The corresponding XML data would be something like:

```
<LineItemList>
    <LineItem>
        <A>…</A>
        <B>…</B>
        <C>…</C>
        <A>…</A>
        <B>…</B>
        <C>…</C>
    </LineItem>
</LineItemList>
```

Repeating sequences are another case where some DTD or schema modifications may be required.

Repeating sequences are when you have a sequence of elements that repeat, but each sequence is not contained within a higher-level element.

For example, A, B, C, A, B, C, A, B, C…. (and so on).


IBM Software Group

Repeating sequences

- WDI does not keep enough information internally to treat each sequence as a separate unit
- In this example, it would process all of the A elements together, B elements together, etc.
  - Not really what you want !
- Workaround is to change the DTD/schema so you have a repeating element for each sequence
  - Note: This does change the data slightly

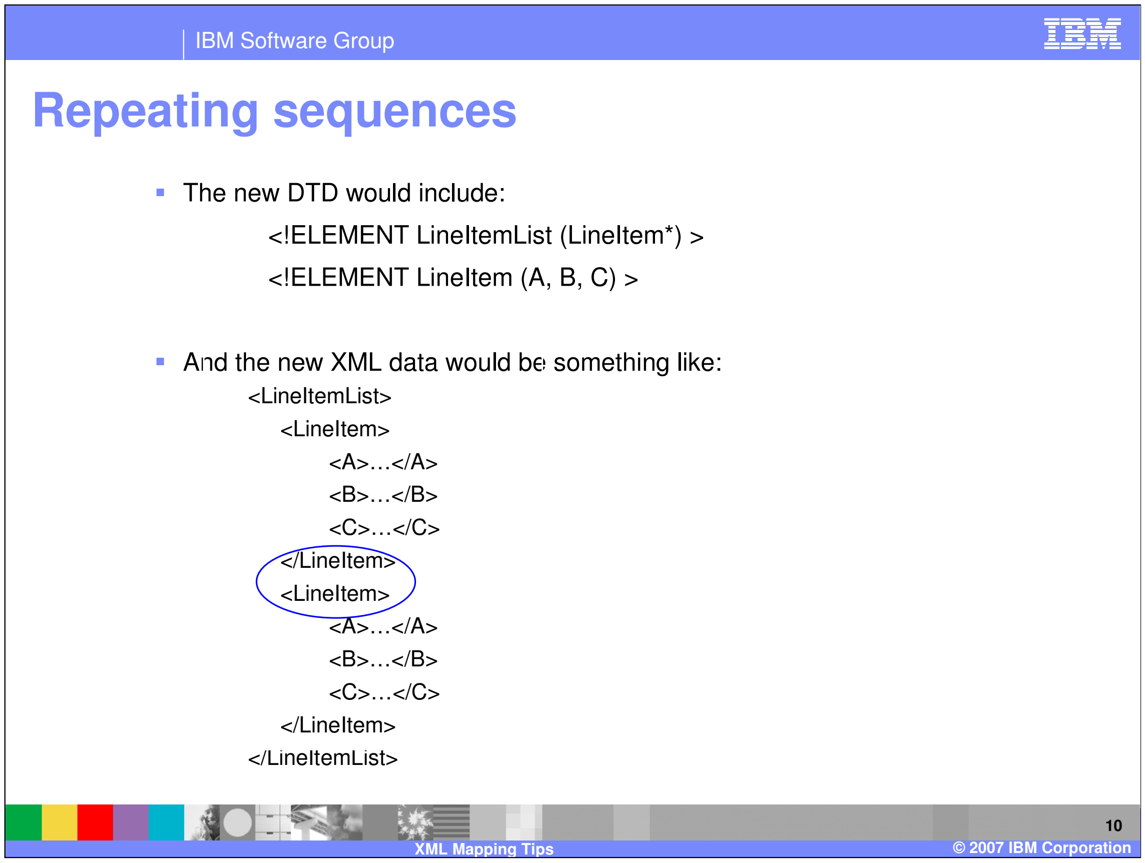
9

XML Mapping Tips

© 2007 IBM Corporation

Repeating sequences are not supported by WDI, since WDI does not keep enough information internally to keep each sequence as a separate unit. In this example, it would process all of the A elements together, all the B elements together, etc. That is probably not what you really want.

The workaround is to change the DTD or schema so you have a repeating element for each sequence. Note that this does change the data slightly.



IBM Software Group

Repeating sequences

- The new DTD would include:

```
<!ELEMENT LineItemList (LineItem*) >
<!ELEMENT LineItem (A, B, C) >
```

- And the new XML data would be something like:

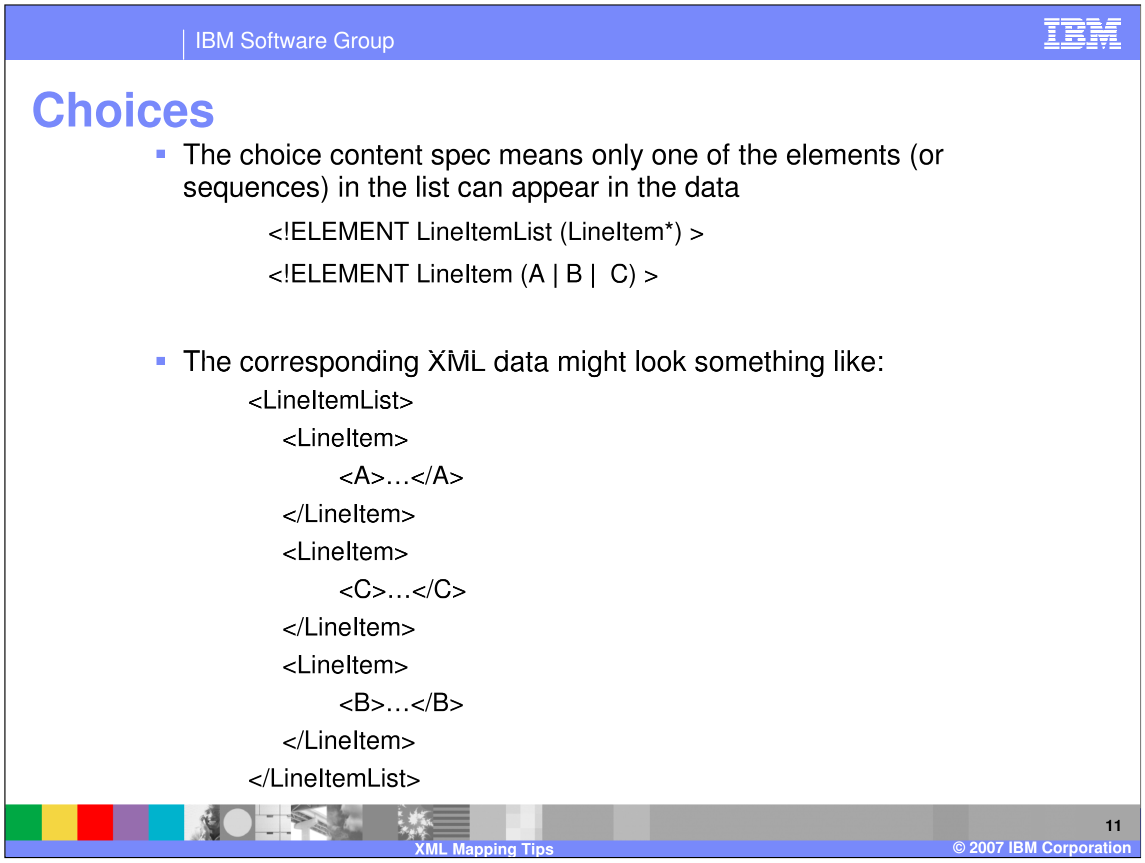
```
<LineItemList>
    <LineItem>
        <A>…</A>
        <B>…</B>
        <C>…</C>
    </LineItem>
    <LineItem>
        <A>…</A>
        <B>…</B>
        <C>…</C>
    </LineItem>
</LineItemList>
```

10

XML Mapping Tips

© 2007 IBM Corporation

In this example, each A, B, C sequence is enclosed in a repeating LineItem element. This allows WDI to keep each A, B, C sequence together within the occurrence of the LineItem.


IBM Software Group

Choices

- The choice content spec means only one of the elements (or sequences) in the list can appear in the data

```
<!ELEMENT LineItemList (LineItem*) >
<!ELEMENT LineItem (A | B |  C) >
```

- The corresponding XML data might look something like:

```
<LineItemList>
    <LineItem>
        <A>…</A>
    </LineItem>
    <LineItem>
        <C>…</C>
    </LineItem>
    <LineItem>
        <B>…</B>
    </LineItem>
</LineItemList>
```

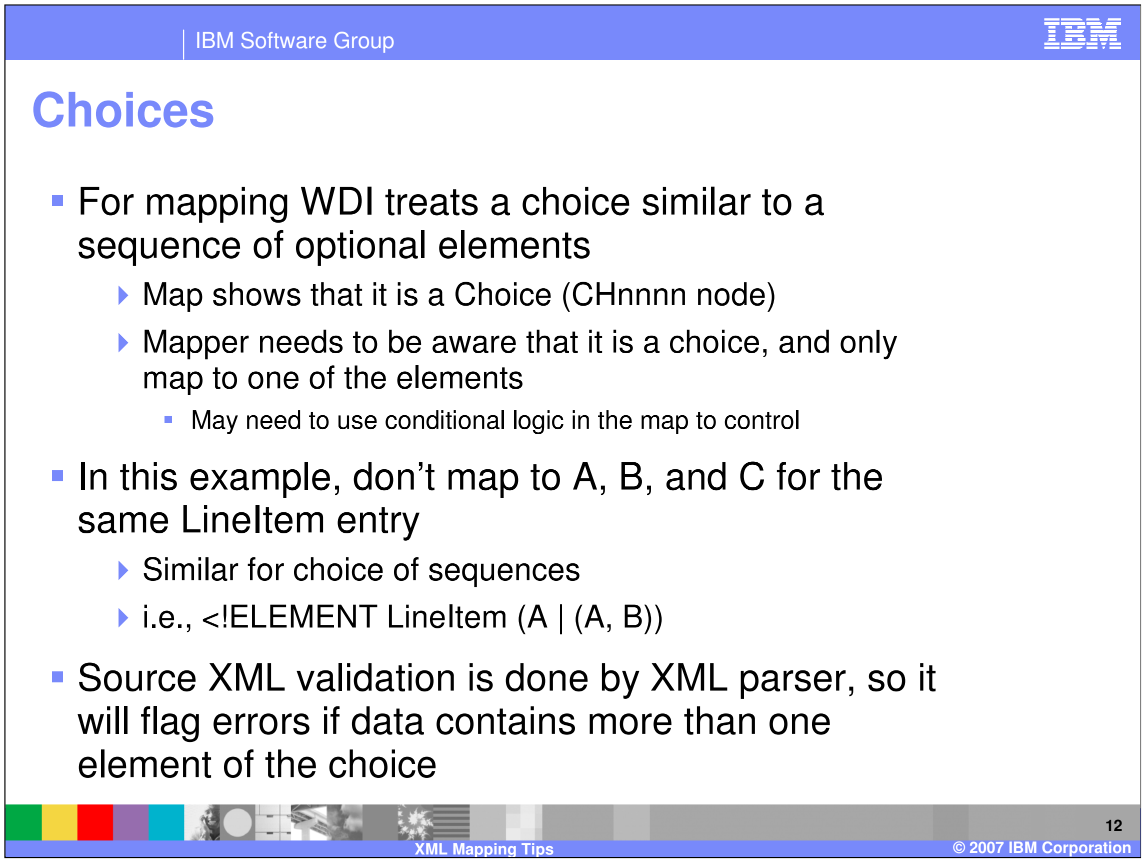
11

XML Mapping Tips

© 2007 IBM Corporation

The choice content spec means that only one of the elements (or possibly sequences) in the list can appear in the data.

In this example, each LineItem contains A, B, or C, but does not contain more than one of them.


IBM Software Group

Choices

- For mapping WDI treats a choice similar to a sequence of optional elements
  - Map shows that it is a Choice (CHnnnn node)
  - Mapper needs to be aware that it is a choice, and only map to one of the elements
    - May need to use conditional logic in the map to control
- In this example, don't map to A, B, and C for the same LineItem entry
  - Similar for choice of sequences
  - i.e., <!ELEMENT LineItem (A | (A, B))
- Source XML validation is done by XML parser, so it will flag errors if data contains more than one element of the choice

12

XML Mapping Tips

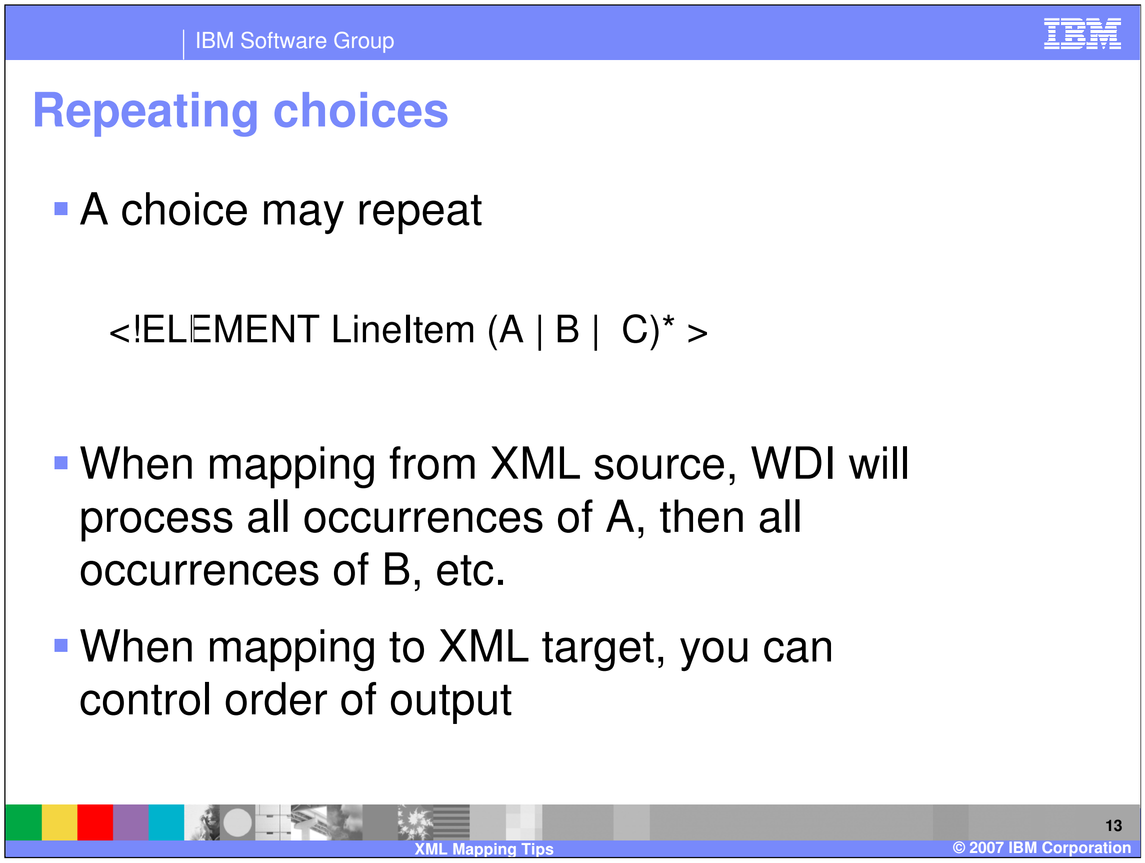
© 2007 IBM Corporation

For mapping, WDI treats a choice similar to a sequence of optional elements. The map shows that it is a choice by putting each element inside a CH(number) node in the map tree, but it does not prevent the user from mapping to more than one element.

In this example, the map logic should ensure that you only map to A, B, or C for each LineItem, but not to all three.

Similar considerations apply if a sequence is defined as one of the components in the choice.

When you are validating input XML data, the parser uses the schema or DTD information directly (not the mapping tree), so WDI will flag the data as an error if it contains more than one element of the choice.



Repeating choices

- A choice may repeat

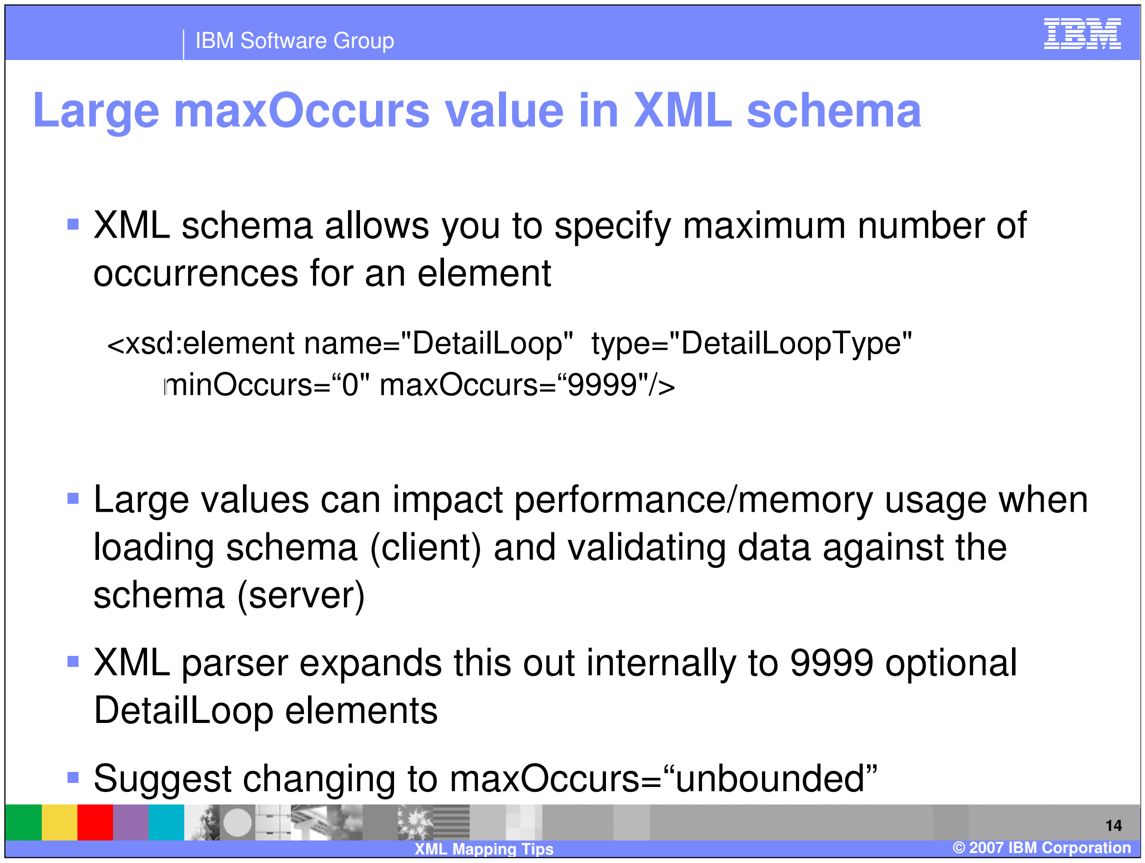
  <!ELEMENT LineItem (A | B | C)* >

- When mapping from XML source, WDI will process all occurrences of A, then all occurrences of B, etc.
- When mapping to XML target, you can control order of output

It is also possible for a choice to repeat. If you are mapping from an XML source document, WDI will process all occurrences of each element in the order they are defined, not the order they appear in the data.

So in this example, WDI would process all occurrences of A, then all occurrences of B, then all occurrences of C, even if the order is mixed in the XML input document.

When mapping to an XML target document, the A, B, and C elements would appear in the output in the order they were created by the map.



Large maxOccurs value in XML schema

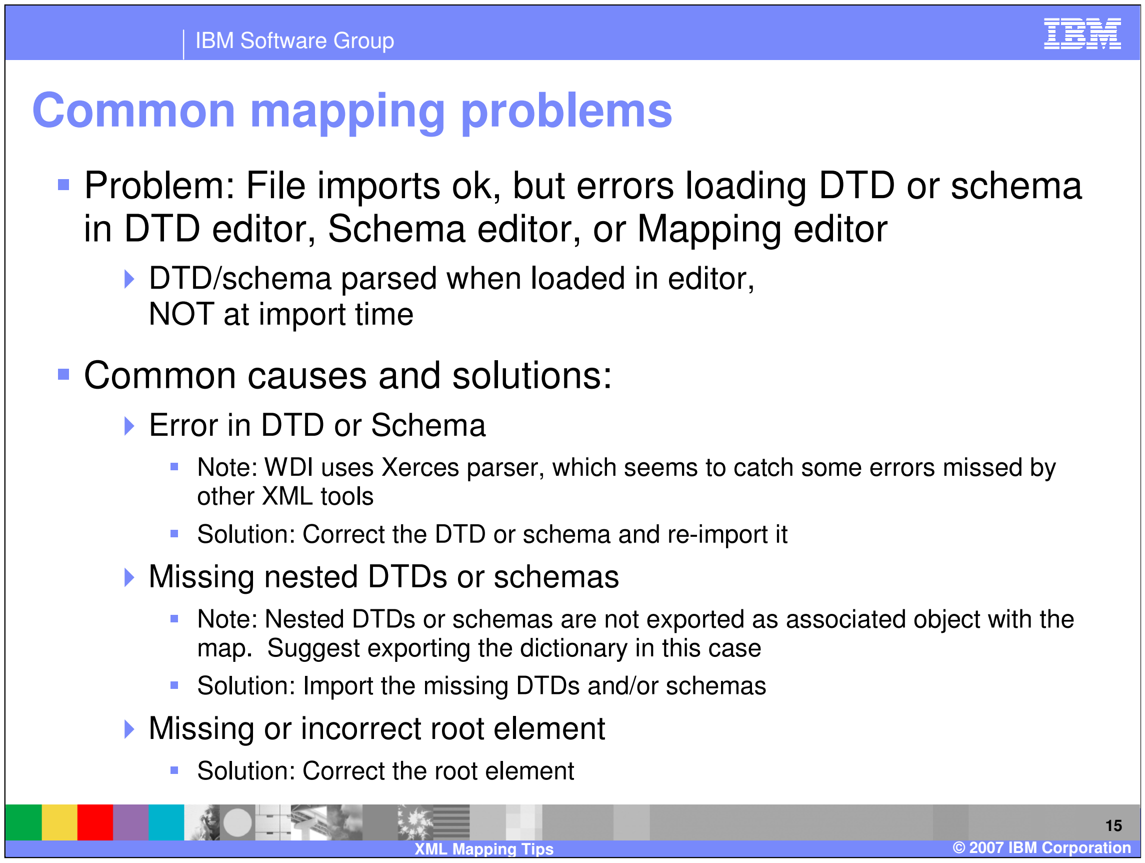
- XML schema allows you to specify maximum number of occurrences for an element

```
<xsd:element name="DetailLoop"  type="DetailLoopType"
      minOccurs="0" maxOccurs="9999"/>
```

- Large values can impact performance/memory usage when loading schema (client) and validating data against the schema (server)
- XML parser expands this out internally to 9999 optional DetailLoop elements
- Suggest changing to maxOccurs="unbounded"

XML schemas allow you to specify a maxOccurs value to specify the maximum number of times that an element can repeat. If these values are very large, this can impact performance and memory usage when you load the schema on the client, and when you validate against the schema on the server. This is because internally, the XML parser expands this out as a number of optional elements. In this example, the DetailLoop would be represented internally as 9999 optional elements.

To improve performance for these cases, you may want to modify the schema to show maxOccurs="unbounded", which allows the XML parser to handle it more efficiently.



IBM Software Group

Common mapping problems

- Problem: File imports ok, but errors loading DTD or schema in DTD editor, Schema editor, or Mapping editor
  - DTD/schema parsed when loaded in editor, NOT at import time
- Common causes and solutions:
  - Error in DTD or Schema
    - Note: WDI uses Xerces parser, which seems to catch some errors missed by other XML tools
    - Solution: Correct the DTD or schema and re-import it
  - Missing nested DTDs or schemas
    - Note: Nested DTDs or schemas are not exported as associated object with the map. Suggest exporting the dictionary in this case
    - Solution: Import the missing DTDs and/or schemas
  - Missing or incorrect root element
    - Solution: Correct the root element

15

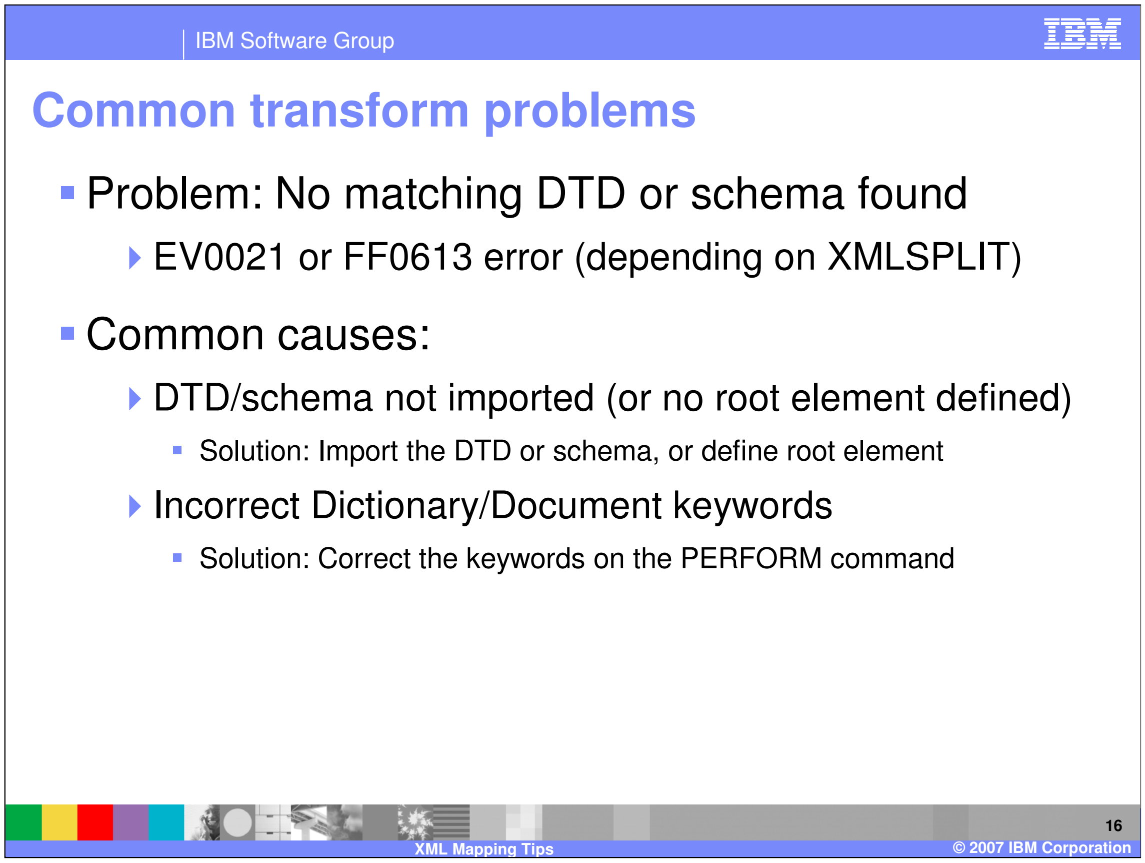
XML Mapping Tips

© 2007 IBM Corporation

One common problem mapping with XML data is that the file imports ok, but then there are errors when you try to load the DTD or schema in one of the editors. The schema or DTD does not get parsed when you import it – only when it is actually loaded in one of the editors. When you see the message that it was imported successfully, this just means that it was loaded into the database – not checked to see if it is correct.

Some common causes and solutions include:

1. The DTD or schema is in fact incorrect. This may be a syntax error, or some other type of error. WDI uses the Xerces parser, which seems to catch some errors that are overlooked by other XML tools. The solution for this is to correct the DTD or schema and re-import it.
2. The DTD or schema contains references to other DTDs or schemas, but the other DTDs or schemas are missing from the XML Dictionary. This may happen because you forgot to import all of the DTDs or schemas, or it may occur because nested DTDs and schemas are not exported as associated objects with the map. If you are exporting a map that uses nested schemas or DTDs, it is usually a good idea to export the entire XML Dictionary.
   If you are missing DTDs or schemas, the solution of course is to import them into the XML Dictionary.
3. The root element is missing or incorrect. If this happens, WDI does not know which element it should use to for the root of the document tree structure. If this happens, you need to correct the root element. Note that even if the schema is using namespaces, you do not include the prefix on the root element.
   Some schemas or DTDs are only nested within other schemas or DTDs – the elements they define are never used as the root element for a document. In this case, you do not need to specify a root element.


IBM Software Group

Common transform problems

- Problem: No matching DTD or schema found
  - EV0021 or FF0613 error (depending on XMLSPLIT)
- Common causes:
  - DTD/schema not imported (or no root element defined)
    - Solution: Import the DTD or schema, or define root element
  - Incorrect Dictionary/Document keywords
    - Solution: Correct the keywords on the PERFORM command

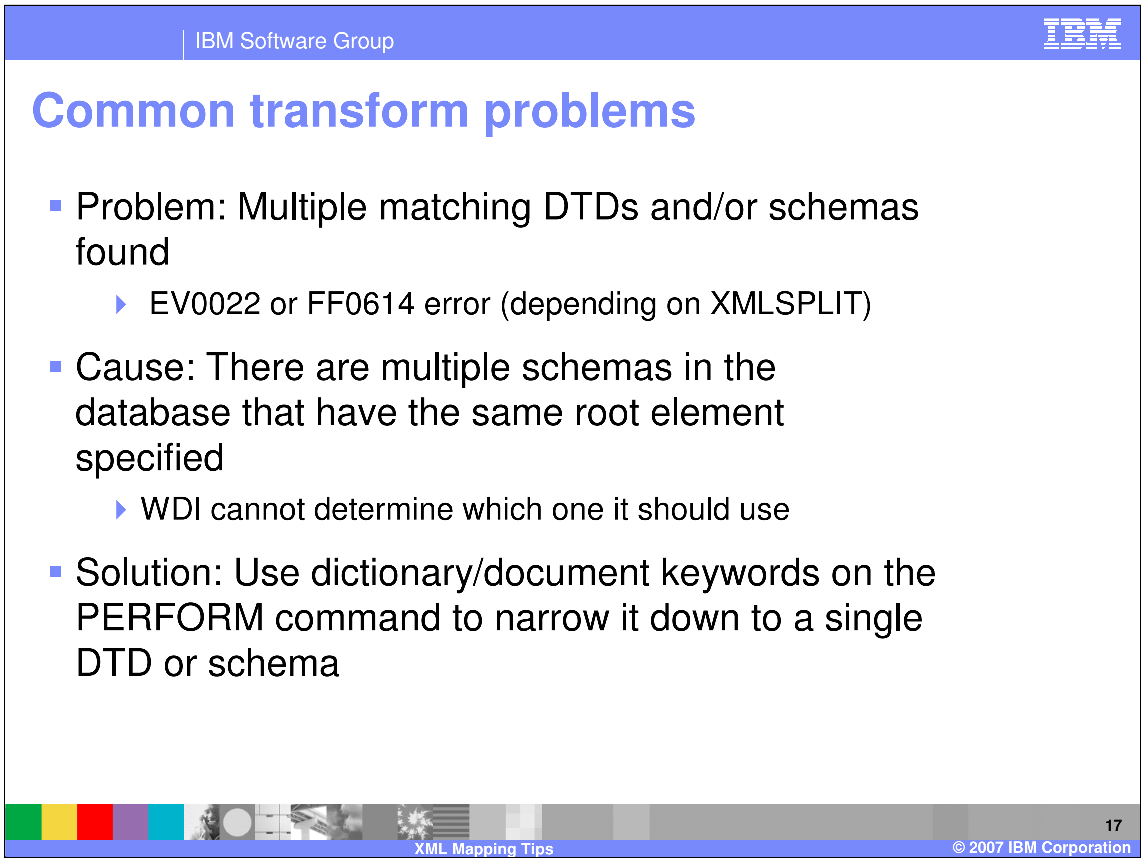
16

XML Mapping Tips © 2007 IBM Corporation

Sometimes during the TRANSFORM you may get an error that says "No matching DTD or schema was found" or something similar. Depending on your XMLSPLIT keyword, this may be either an EV0021 or FF0613 error. This is because WDI finds the root element in the XML document, then uses the root element to look up the DTD or schema the database and determine various processing information. If the root element is not found, then one of these errors is returned.

One common cause for this is that the DTD or schema is not imported or no root element is defined for it. This might happen if the map was imported from another system, but the XML DTD or schema was not. If this is the case, you need to import the DTD or schema and define the root element.

Another common cause is that the DICTIONARY and/or DOCUMENT keywords on the PERFORM TRANSFORM command are incorrect. You can use these keywords to filter DTDs and schemas that you DO NOT want to be used for this TRANSFORM command. This is useful in cases where you have multiple DTDs or schemas that use the same root element. However, if the DICTIONARY and/or DOCUMENT are incorrect, you might filter out the DTDs or schema that you DO want to use. If this occurs, correct the keywords on the PERFORM TRANSFORM command.

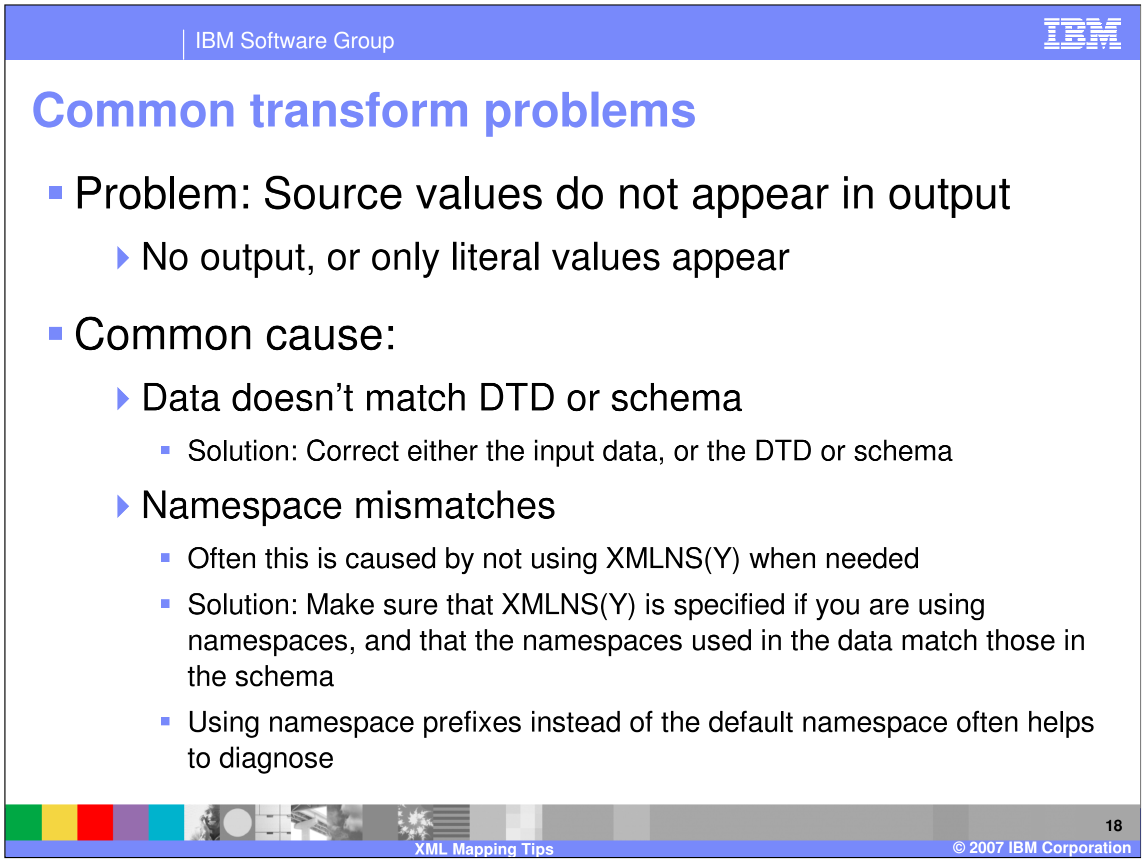


Common transform problems

- Problem: Multiple matching DTDs and/or schemas found
  - EV0022 or FF0614 error (depending on XMLSPLIT)
- Cause: There are multiple schemas in the database that have the same root element specified
  - WDI cannot determine which one it should use
- Solution: Use dictionary/document keywords on the PERFORM command to narrow it down to a single DTD or schema

Sometimes WDI might find multiple DTDs and/or schemas that all have the same root element as the XML document. This is similar to the previous problem, since WDI cannot determine which DTD or schema in the database it should use for the processing information.

In this case, you may need to ADD the DICTIONARY and/or DOCUMENT keywords in order to filter out the additional DTDs or schemas, so WDI will only find a single match.


IBM Software Group

Common transform problems

- Problem: Source values do not appear in output
  - No output, or only literal values appear
- Common cause:
  - Data doesn't match DTD or schema
    - Solution: Correct either the input data, or the DTD or schema
  - Namespace mismatches
    - Often this is caused by not using XMLNS(Y) when needed
    - Solution: Make sure that XMLNS(Y) is specified if you are using namespaces, and that the namespaces used in the data match those in the schema
    - Using namespace prefixes instead of the default namespace often helps to diagnose

18

XML Mapping Tips

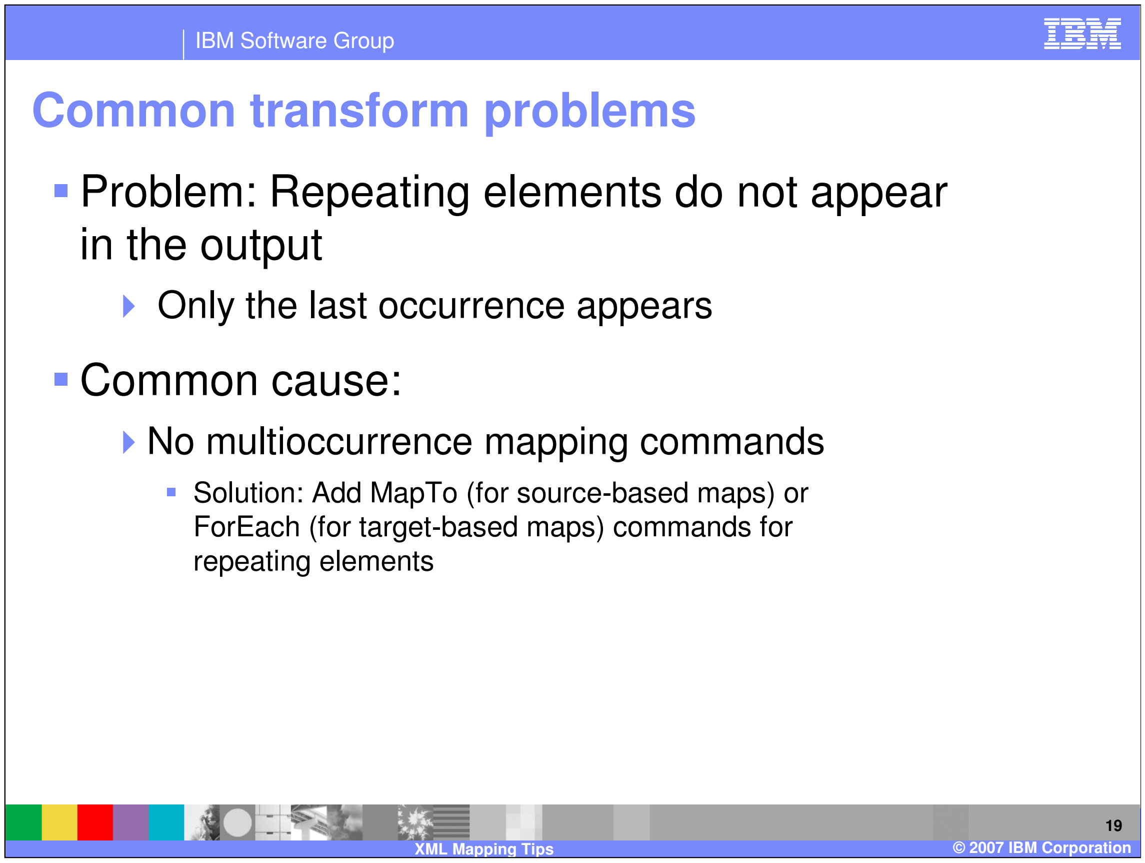
© 2007 IBM Corporation

You may have cases where none of the source values are mapped to the target document. Either no output is created, or the output may only contain literal values from the map.

This is typically caused by having XML input data that does not match the DTD or schema definition correctly. As a result, WDI does not find any of the elements that it is trying to map. For example, the DTD may have an element named “Header” with a capital “H”, but the XML document might have it with a lowercase “h”. WDI would be looking for it with a capital “H” but not find it. WDI would not recognize the header element with the lowercase “h”, so would not map any of the elements within it. “Close” doesn't count with XML.

Sometimes the input data does not match because of namespace issues. Either the XMLNS keyword was omitted when it should have been used, or the namespace URI in the data does not match the namespace URI in the schema. Sometimes using namespace prefixes instead of the default namespace makes it easier to diagnose namespace issues.

For cases where you suspect some type of mismatch between the XML data and the DTD or schema, it may be helpful to either have WDI validate the input data using the XMLVALIDATE or XMLSCHEMAVAL keywords, or to validate the data with some other XML tool. These may point out errors that would be difficult to find if you are using complex DTDs or schemas.

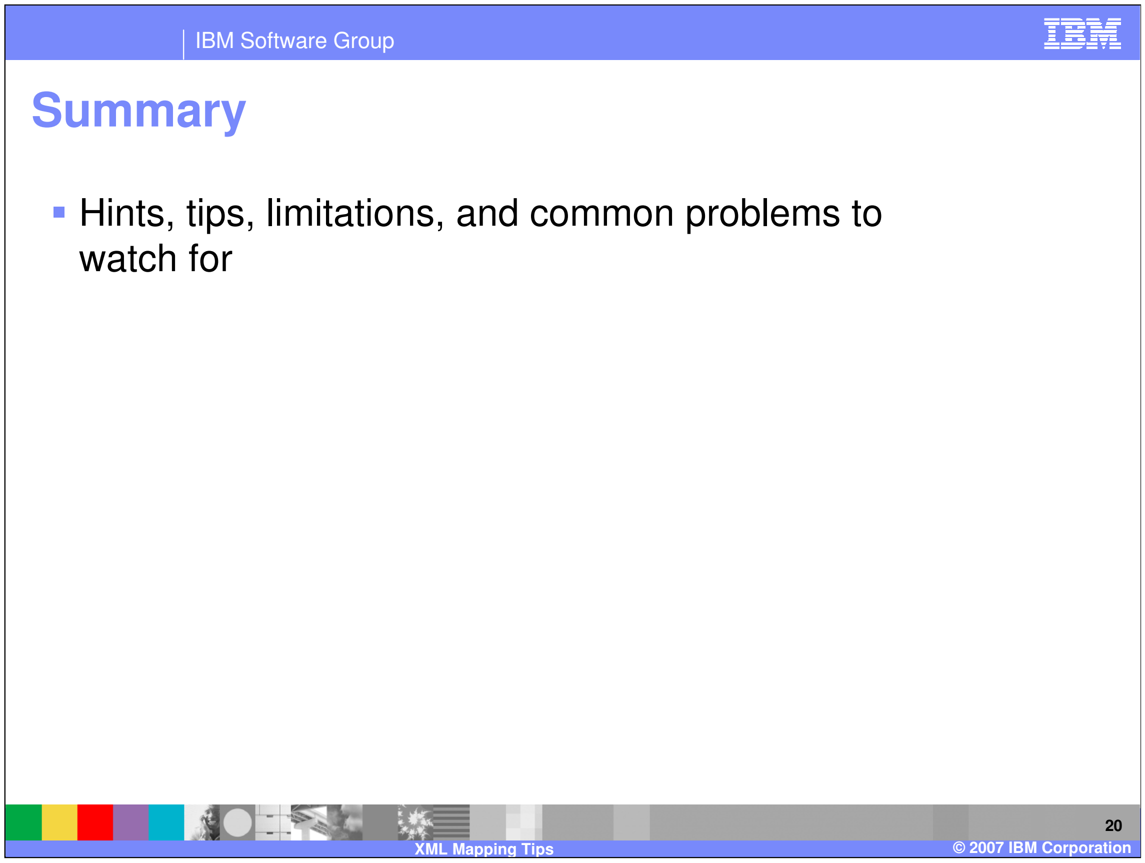


Common transform problems

- Problem: Repeating elements do not appear in the output
  - Only the last occurrence appears
- Common cause:
  - No multioccurrence mapping commands
    - Solution: Add MapTo (for source-based maps) or ForEach (for target-based maps) commands for repeating elements

Another common concern is that sometimes repeating elements do not repeat in the output – only the last occurrence is there. This is not limited to XML data, but can occur with EDI and Data Formats too.

This is typically caused because the map did not correctly use multioccurrence mapping. You use the MapTo command for source-based maps, or ForEach command for target-based maps, to tell WDI that you want to generate a new occurrence of the target element for each occurrence of the source element. If you do not use multioccurrence mapping, then the value of each source element just overwrites the existing target value.



Summary

- Hints, tips, limitations, and common problems to watch for

In summary, we described various hints and tips for handling special situations. We also identified some common problems, and described typical solutions for them.

Most users will not need ALL of these hints and tips. But for those that run into one or more of these situations, these suggestions may save you having to call for support.

Template Revision: 04/25/2006 11:09 AM

Trademarks, copyrights, and disclaimers