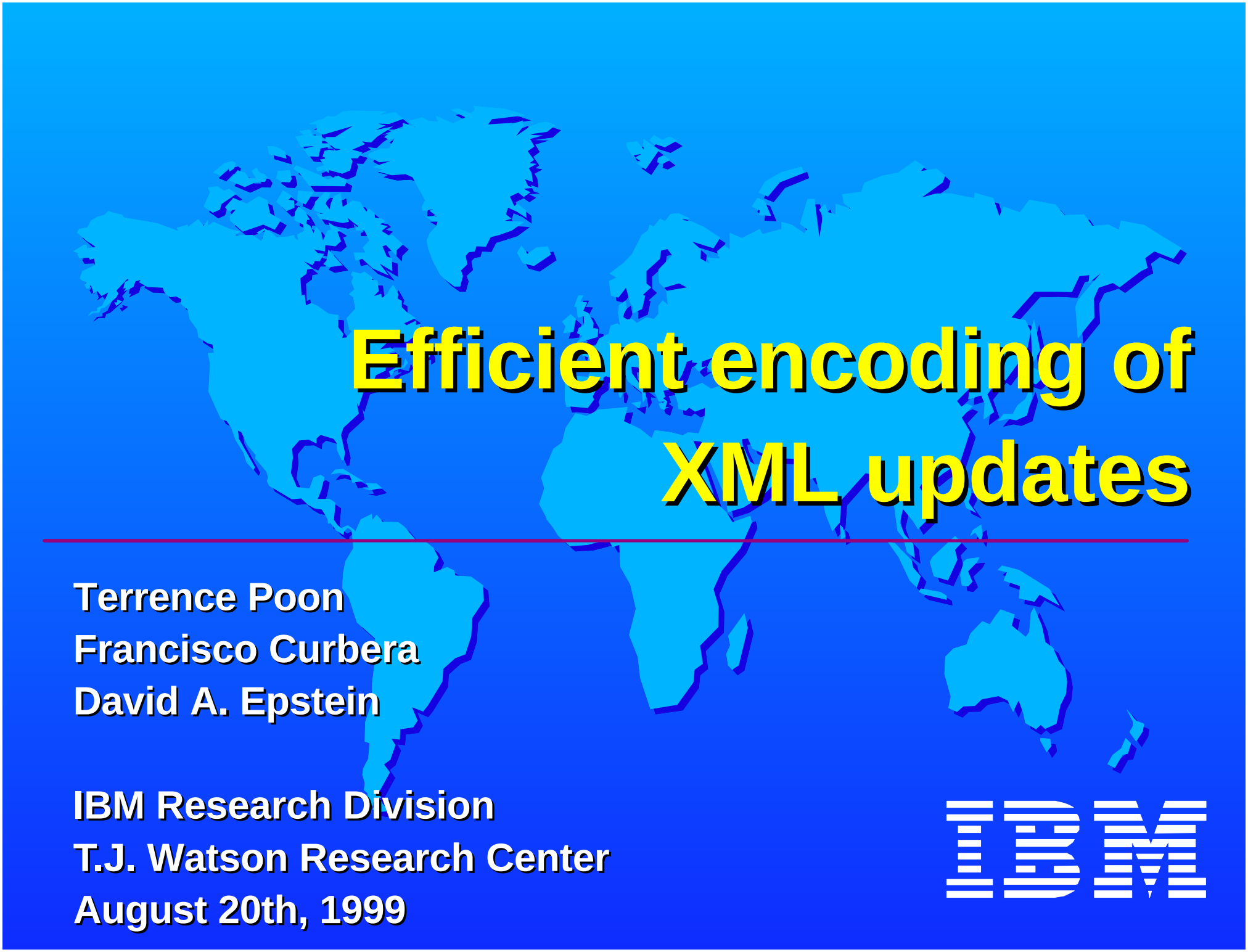


Efficient encoding of XML updates

Terrence Poon
Francisco Curbera
David A. Epstein

IBM Research Division
T.J. Watson Research Center
August 20th, 1999

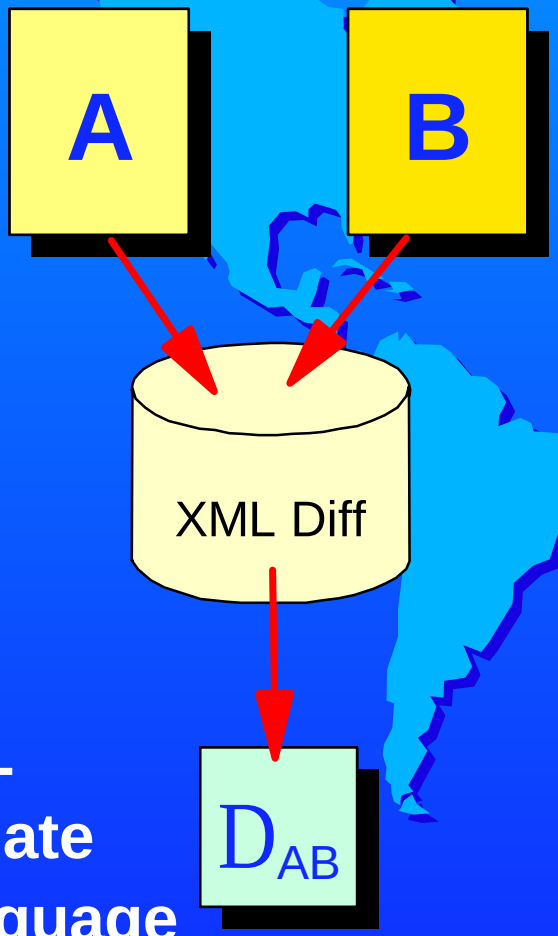
IBM

Overview

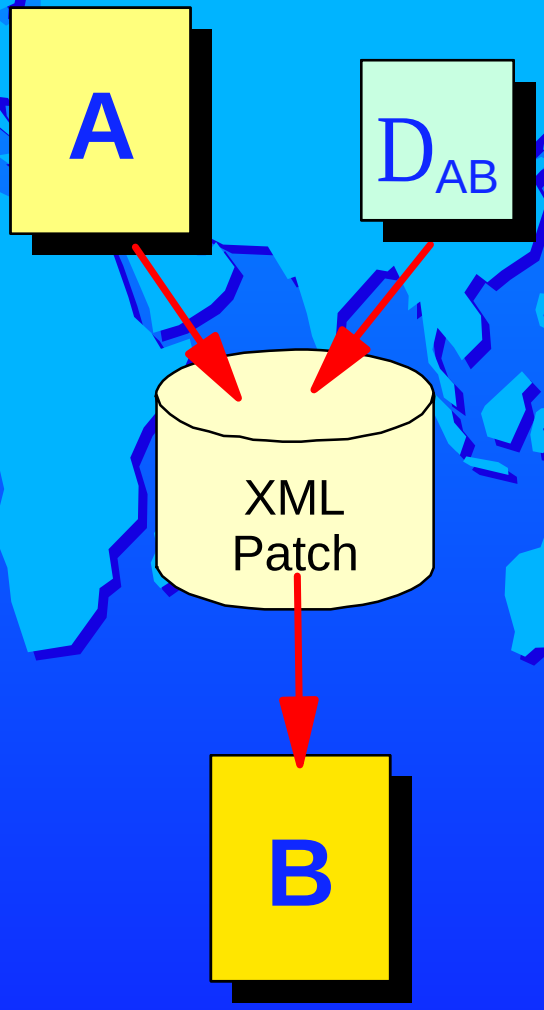
- Introduction
- Motivation
- XML Update Language (XUL)
- Patch algorithm
- Applications / demo
- Future work: adding semantics to XUL

Introduction

Differentiation



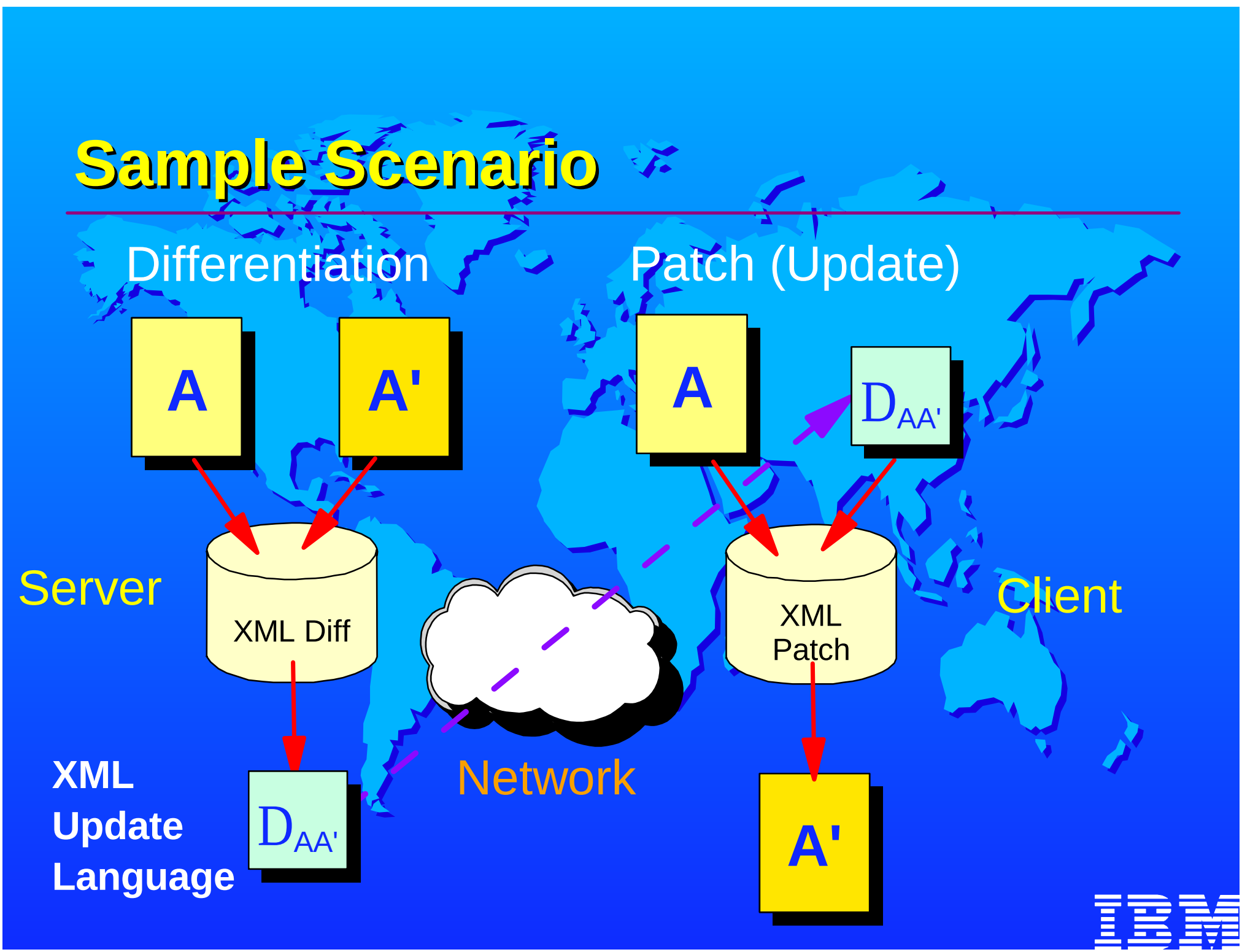
Patch (Update)



XML Update Language

IBM

Sample Scenario



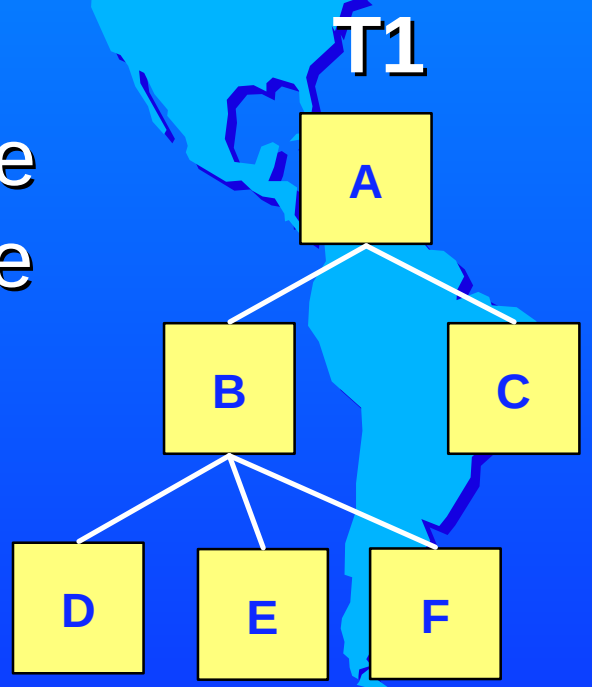
Motivation

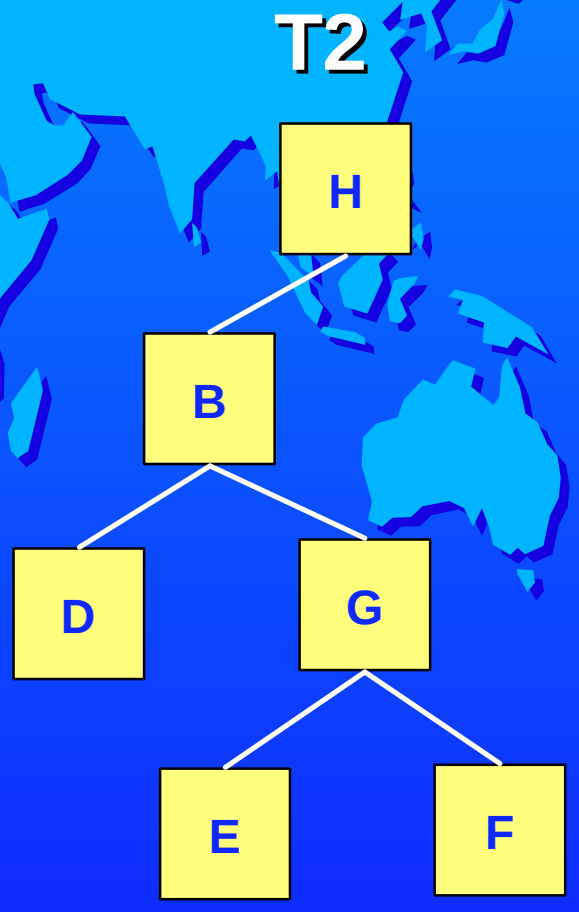
- Small update vs. large document
- Advantages over "flat" diff and patch
  - Understands the tree structure of an XML document
  - Avoids problems from multiple surface string representations
  - Operates directly on DOM - no re-parsing needed

Tree-to-tree correction problem

- Find the least expensive sequenece of operations to transform T1 to T2

add
remove
replace



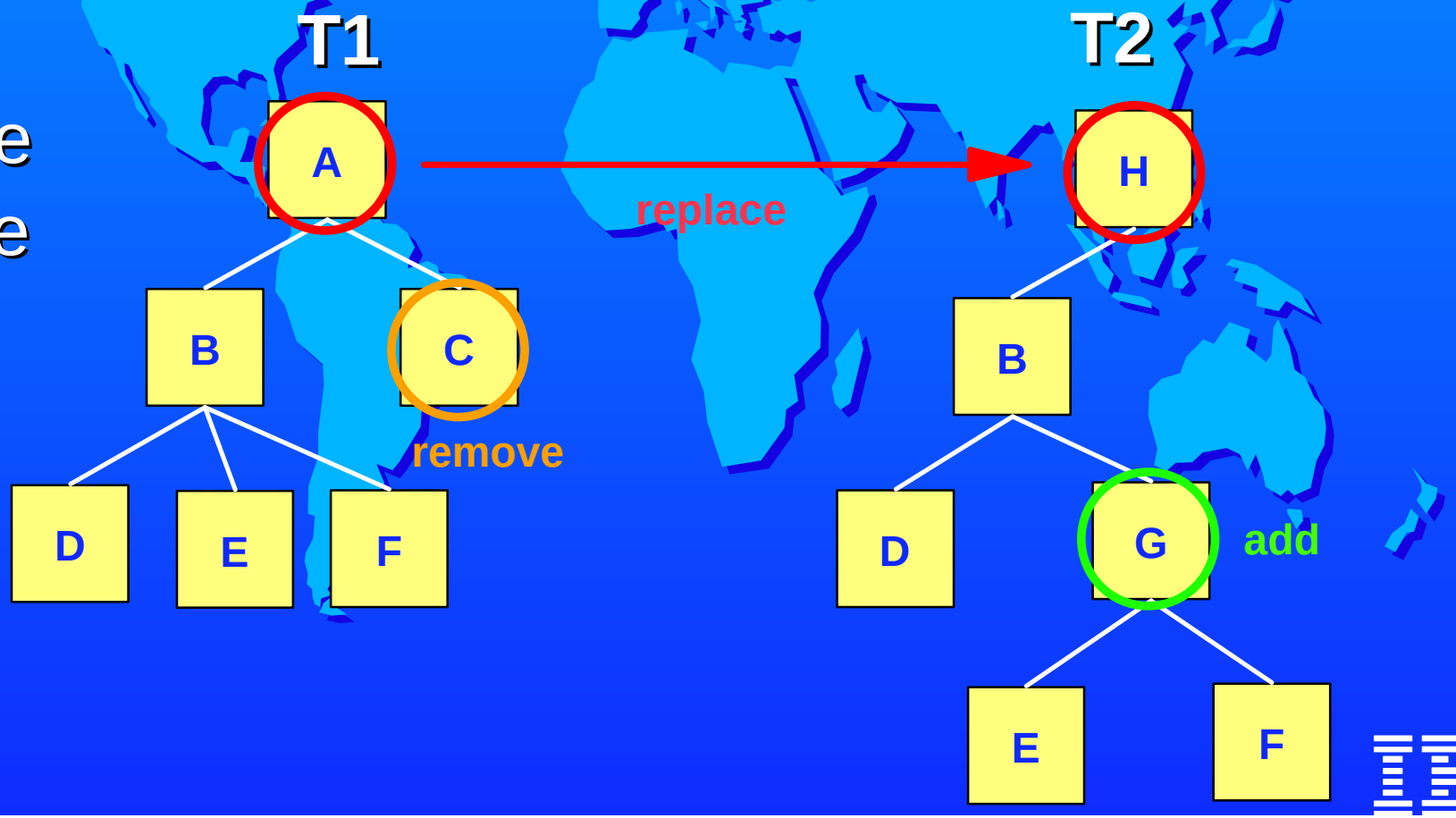


IBM

Tree-to-tree correction problem

- Find the least expensive sequenece of operations to transform T1 to T2

add
remove
replace

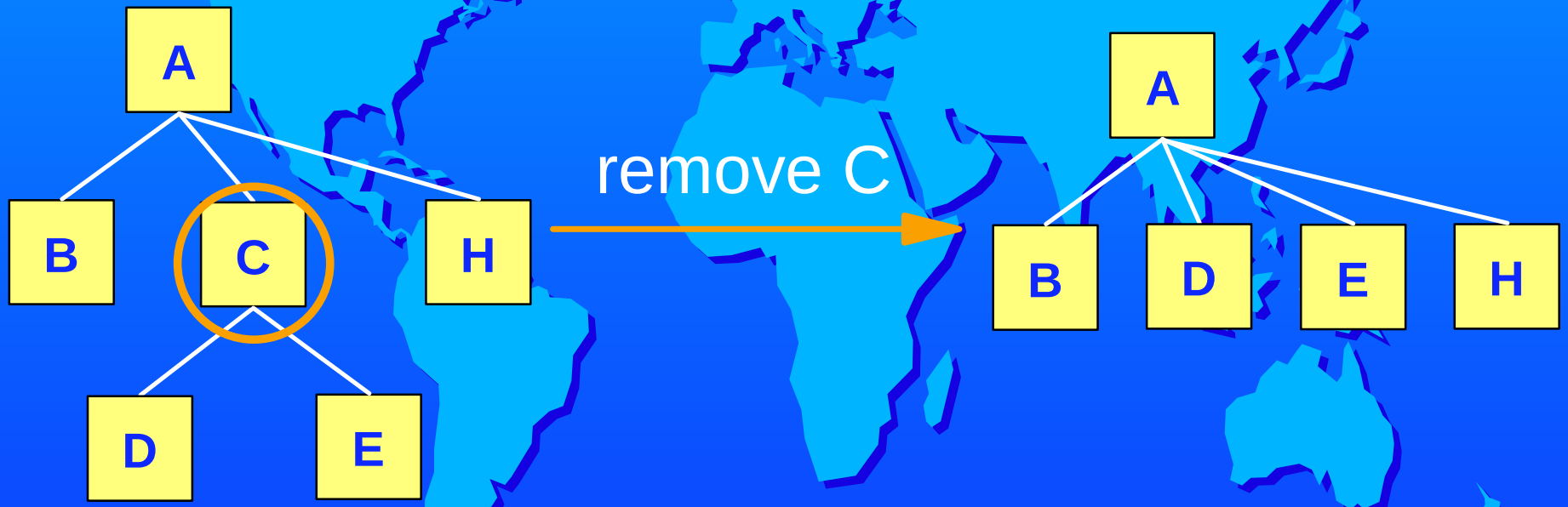


Basic operations

- Replace

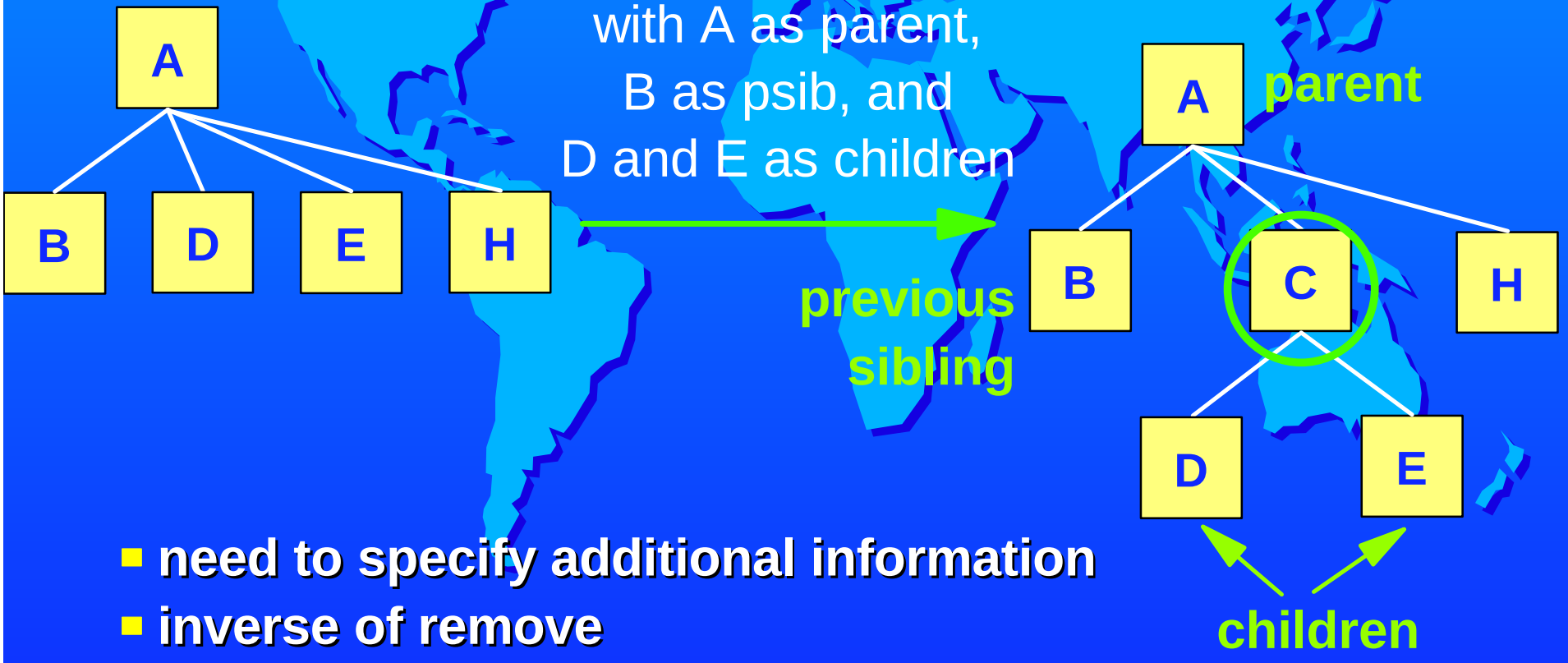
Basic operations

- Remove



Basic operations

- **Add**



- **need to specify additional information**
- **inverse of remove**

IBM

Encoding document updates

- Language requirements:
  - XML-based
  - Basic operations: add, remove, replace
  - Aggregate operations: prune, graft, move
  - Expresses dependencies between operations
  - Refers only to initial and final states of document
- Naive approach (FUL)
- XML Update Language (XUL)

IBM

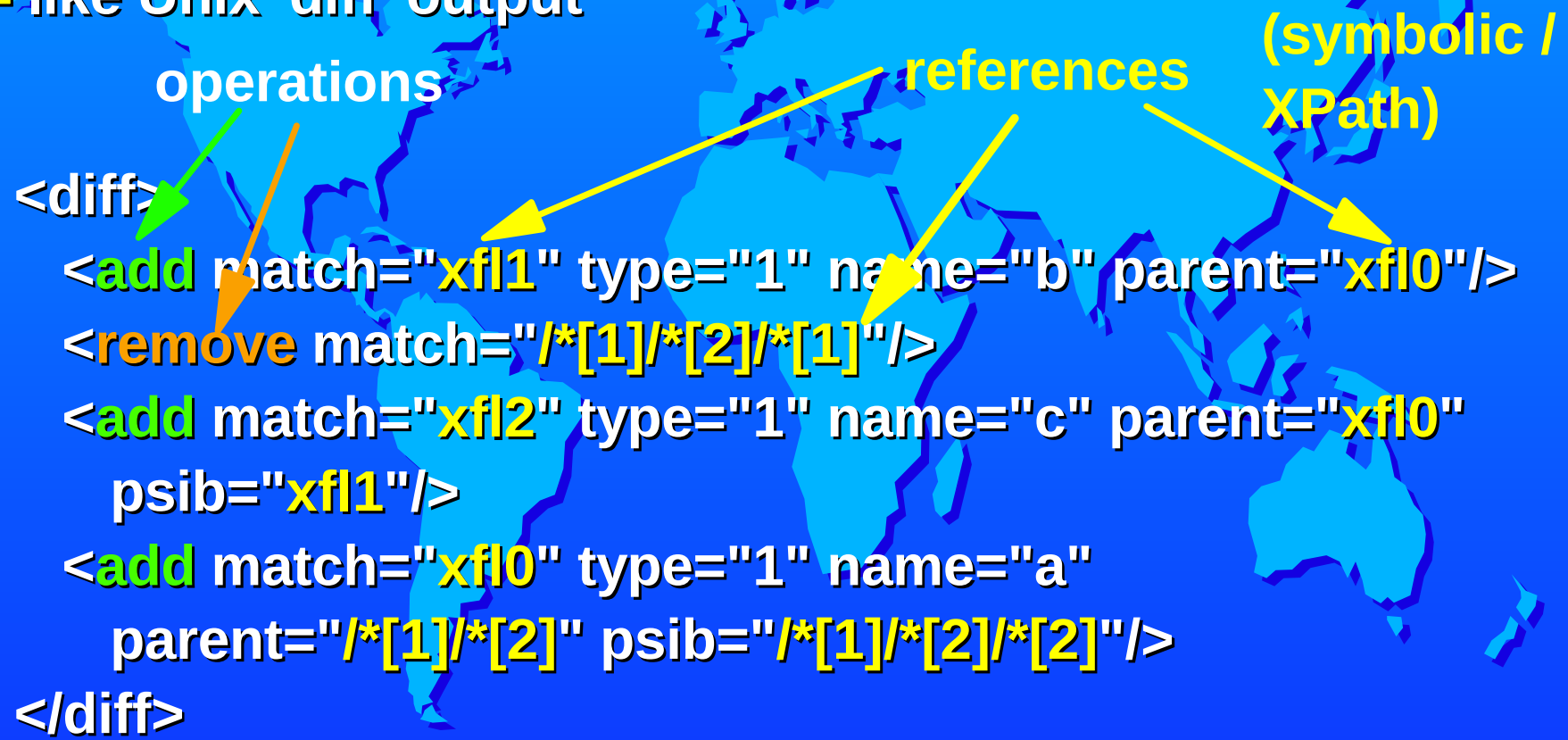
Naive approach (FUL)

- Flat list of operations
- like Unix 'diff' output

```
<diff>
  <add match="xfl1" type="1" name="b" parent="xfl0"/>
  <remove match="/*[1]/*[2]/*[1]"/>
  <add match="xfl2" type="1" name="c" parent="xfl0"
    psib="xfl1"/>
  <add match="xfl0" type="1" name="a"
    parent="/*[1]/*[2]" psib="/*[1]/*[2]/*[2]"/>
</diff>
```

Naive approach (FUL)

- Flat list of operations
- like Unix 'diff' output

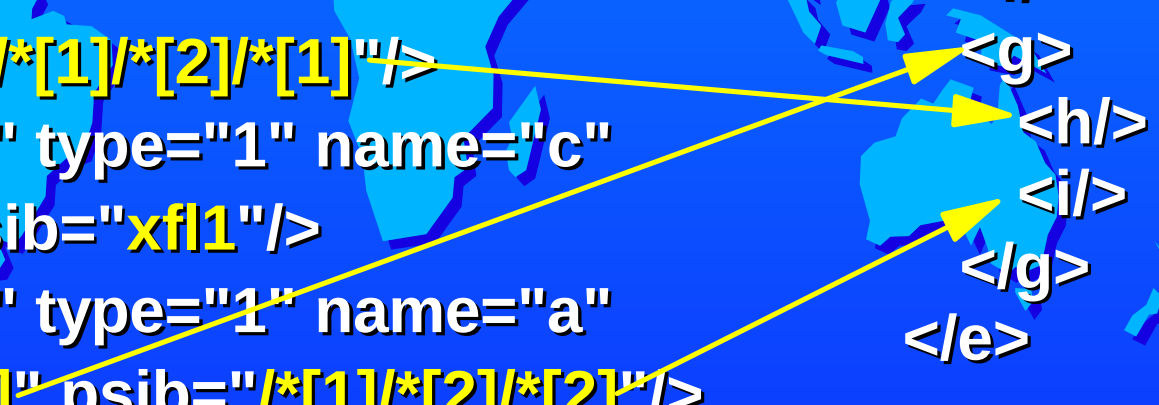


Naive approach (FUL)

- Flat list of operations
- like Unix 'diff' output

```
<diff>
  <add match="xfl1" type="1" name="b"
     parent="xfl0"/>
  <remove match="/*[1]/*[2]/*[1]"/>
  <add match="xfl2" type="1" name="c"
    parent="xfl0" psib="xfl1"/>
  <add match="xfl0" type="1" name="a"
    parent="/*[1]/*[2]" psib="/*[1]/*[2]/*[2]"/>
</diff>
```

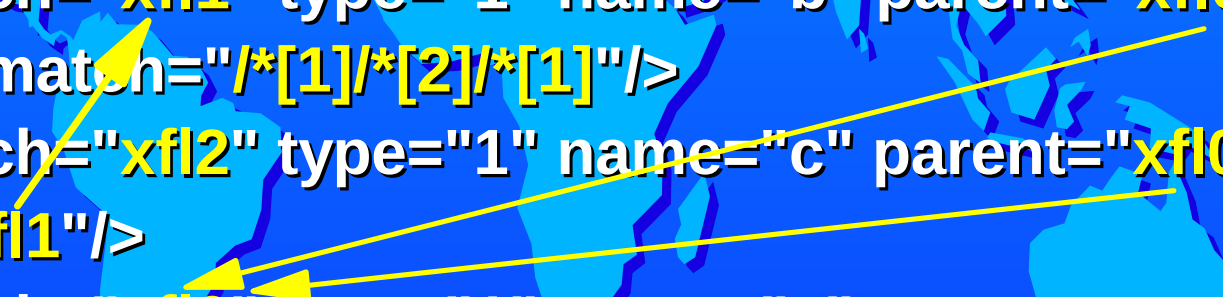
```
<e>
  <f/>
  <g>
    <h/>
    <i/>
  </g>
</e>
```



Problems: naive approach (FUL)

- Abundance of symbolic references
  - ex. xfl0, xfl1, etc.
  - due to structured nature of XML

```
<diff>
  <add match="xfl1" type="1" name="b" parent="xfl0"/>
  <remove match="/*[1]/*[2]/*[1]"/>
  <add match="xfl2" type="1" name="c" parent="xfl0"
    psib="xfl1"/>
  <add match="xfl0" type="1" name="a"
    parent="/*[1]/*[2]" psib="/*[1]/*[2]/*[2]"/>
</diff>
```



IBM

Problems: naive approach (FUL)

- Implicit dependencies
  - need topological sorting

```
<diff>
  <add match="xfl1" type="1" name="b" parent="xfl0"/>
  <remove match="/*[1]/*[2]/*[1]"/>
  <add match="xfl2" type="1" name="c" parent="xfl0"
    psib="xfl1"/>
  <add match="xfl0" type="1" name="a"
    parent="/*[1]/*[2]" psib="/*[1]/*[2]/*[2]"/>
</diff>
```

IBM

Problems: naive approach (FUL)

- Implicit dependencies
  - need topological sorting
  - difficult to parallelize (need partitioning)

```
<diff>
  <remove match="/*[1]/*[2]/*[1]"/>
  <add match="xfl0" type="1" name="a"
    parent="/*[1]/*[2]" psib="/*[1]/*[2]/*[2]"/>
  <add match="xfl1" type="1" name="b" parent="xfl0"/>
  <add match="xfl2" type="1" name="c" parent="xfl0"
    psib="xfl1"/>
</diff>
```

IBM

Problems: naive approach (FUL)

- Hard to interpet
  - unstructured
  - absolute paths - no context

```
<diff>
  <add match="xfl1" type="1" name="b" parent="xfl0"/>
  <remove match="/*[1]/*[2]/*[1]"/>
  <add match="xfl2" type="1" name="c" parent="xfl0"
    psib="xfl1"/>
  <add match="xfl0" type="1" name="a"
    parent="/*[1]/*[2]" psib="/*[1]/*[2]/*[2]"/>
</diff>
```

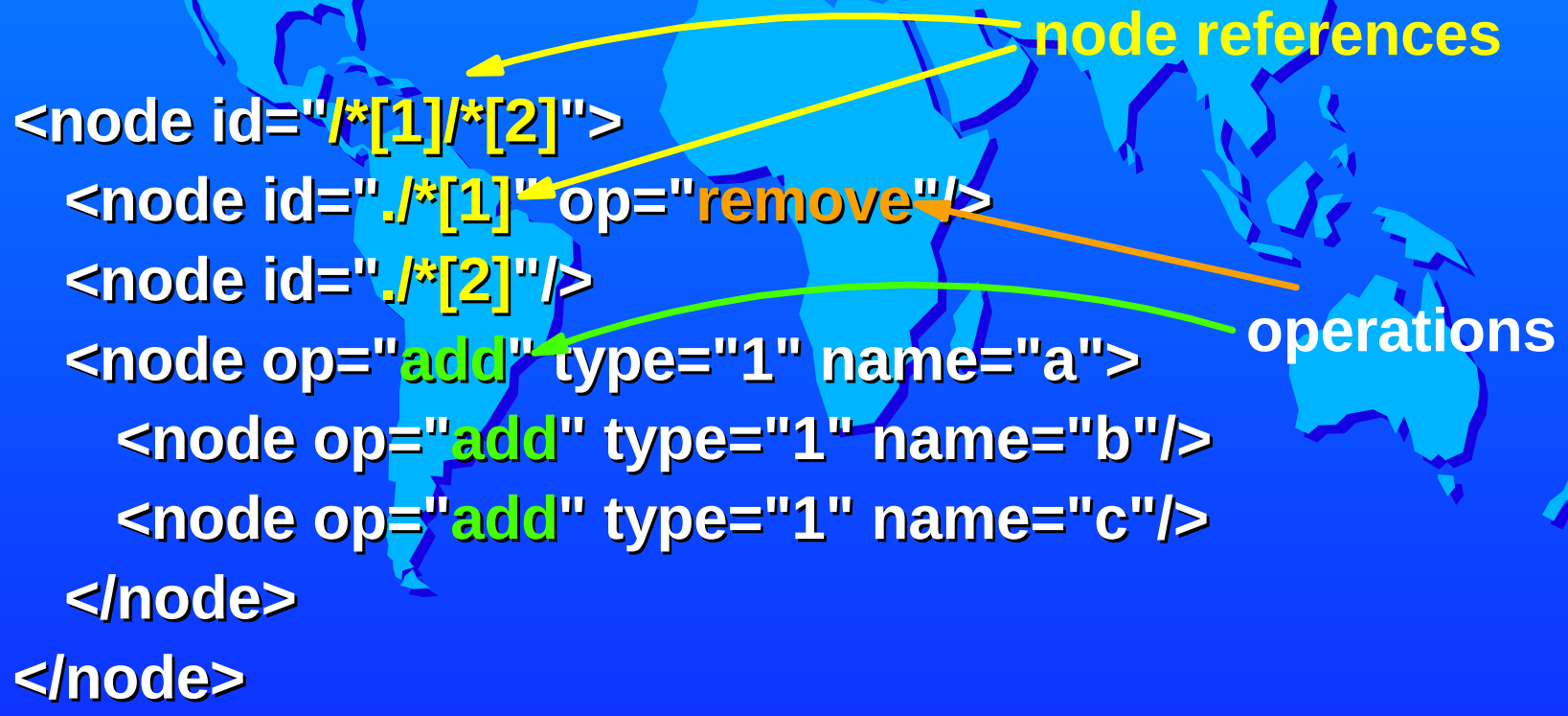
XML Update Language (XUL)

- structured, contextual language

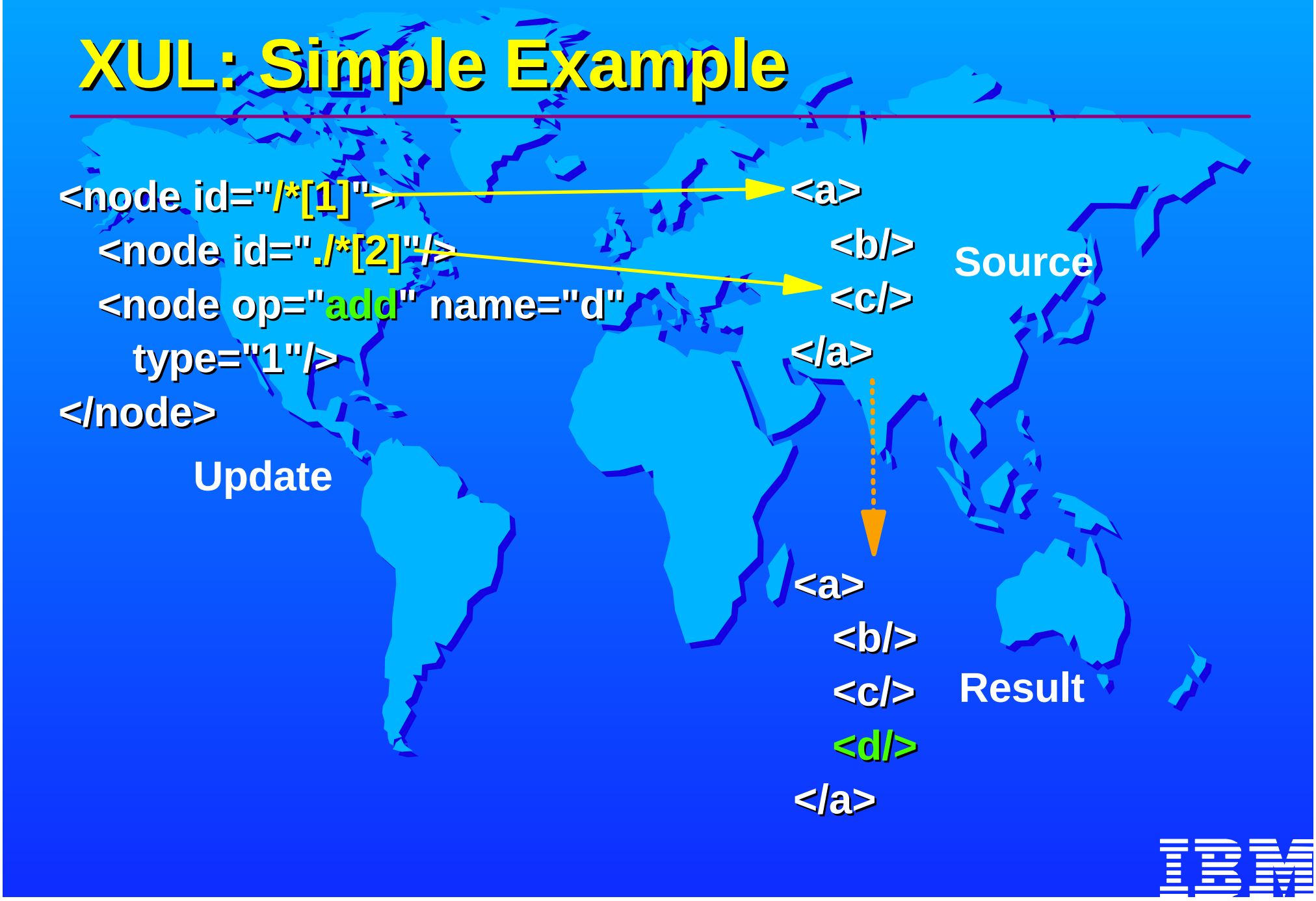
```
<node id="/*[1]/*[2]">
  <node id="./*[1]" op="remove"/>
  <node id="./*[2]"/>
  <node op="add" type="1" name="a">
    <node op="add" type="1" name="b"/>
    <node op="add" type="1" name="c"/>
  </node>
</node>
```

XML Update Language (XUL)

- structured, contextual language



XUL: Simple Example



XML Update Language

- structured, contextual language
  - contextual references

```
<node id="/*[1]/*[2]">
  <node id="./*[1]" op="remove"/>
  <node id="./*[2]"/>
  <node op="add" type="1" name="a">
    <node op="add" type="1" name="b"/>
    <node op="add" type="1" name="c"/>
  </node>
</node>
```

XML Update Language

- structured, contextual language
  - explicit dependencies
    - easily parallelizable

```
<node id="/*[1]/*[2]">
  <node id="./*[1]" op="remove"/>
  <node id="./*[2]"/>
  <node op="add" type="1" name="a">
    <node op="add" type="1" name="b"/>
    <node op="add" type="1" name="c"/>
  </node>
</node>
```

IBM

XML Update Language

- structured, contextual language
  - easier to interpet
    - reproduces meaningful parts of the source
    - uses relative paths

```
<node id="/*[1]/*[2]">
  <node id="./*[1]" op="remove"/>
  <node id="./*[2]"/>
  <node op="add" type="1" name="a">
    <node op="add" type="1" name="b"/>
    <node op="add" type="1" name="c"/>
  </node>
</node>
```

XML Update Language

- faster to patch than FUL (~15%)
  - no need for topological sorting
  - uses relative paths

```
<node id="/*[1]/*[2]">
  <node id="./*[1]" op="remove"/>
  <node id="./*[2]"/>
  <node op="add" type="1" name="a">
    <node op="add" type="1" name="b"/>
    <node op="add" type="1" name="c"/>
  </node>
</node>
```

Patch algorithm

- Input: source document and update document
- Output: updated version of the document
- Two pass algorithm
  1. Resolve the node references.
  2. Perform the operations using a depth-first traversal of the XUL.
     - Possibly inconsistent or invalid intermediate states

Resolving node references (XPath)

```
<n id="/*[1]/*[2]">
  <n id="./*[1]" op="r"/>
  <n id="./*[2]"/>
  <n op="a" type="1" name="a">
    <n op="a" type="1" name="b"/>
    <n op="a" type="1" name="c"/>
  </n>
</n>
```

```
<?xml version="1.0"?>
<e>
  <f/>
  <g>
    <h/>
    <i/>
  </g>
</e>
```

Update

Source

Performing the operations (remove)

```
<n id="/*[1]/*[2]">
  <n id="./*[1]" op="r"/>
  <n id="./*[2]"/>
  <n op="a" type="1" name="a">
    <n op="a" type="1" name="b"/>
    <n op="a" type="1" name="c"/>
  </n>
</n>
```

```
<?xml version="1.0"?>
<e>
  <f/>
  <g>
    <h/>
    <i/>
  </g>
</e>
```

Update

Source

Perform the operations (add)

```
<n id="/*[1]/*[2]">
  <n id="./*[1]" op="r"/>
  <n id="./*[2]"/>
  <n op="a" type="1" name="a">
    <n op="a" type="1" name="b"/>
    <n op="a" type="1" name="c"/>
  </n>
</n>
```

```
<?xml version="1.0"?>
<e>
  <f/>
  <g>
    <i/>
  </g>
</e>
```

Update

Intermediate state

IBM

Performing the operations (add)

```
<n id="/*[1]/*[2]">
  <n id="./*[1]" op="r"/>
  <n id="./*[2]"/>
  <n op="a" type="1" name="a">
    <n op="a" type="1" name="b"/>
    <n op="a" type="1" name="c"/>
  </n>
</n>
```

```
<?xml version="1.0"?>
<e>
  <f/>
  <g>
    <i/>
    <a/>
  </g>
</e>
```

Update

Intermediate state

Performing the operations (add)

```
<n id="/*[1]/*[2]">
  <n id="./*[1]" op="r"/>
  <n id="./*[2]"/>
  <n op="a" type="1" name="a">
    <n op="a" type="1" name="b"/>
    <n op="a" type="1" name="c"/>
  </n>
</n>
```

```
<?xml version="1.0"?>
<e>
  <f/>
  <g>
    <i/>
    <a>
      <b/>
    </a>
  </g>
</e>
```

Update

Intermediate state

Performing the operations (add)

```
<n id="/*[1]/*[2]">
  <n id="./*[1]" op="r"/>
  <n id="./*[2]"/>
  <n op="a" type="1" name="a">
    <n op="a" type="1" name="b"/>
    <n op="a" type="1" name="c"/>
  </n>
</n>
```

```
<?xml version="1.0"?>
<e>
  <f/>
  <g>
    <i/>
    <a>
      <b/>
      <c/>
    </a>
  </g>
</e>
```

Update

Updated document

DOM-related issues: Inconsistent intermediate states

- The document root can only have one element child.
  - Intermediate state: the document root may have multiple children

Source

```
<a>
  <b/>
  <c/>
</a>
```

Update (XUL)

```
<node id="/">
  <node id="./*[1]" op="remove">
    <node id="./*[1]" op="remove"/>
  </node>
</node>
```

DOM-related issues: Inconsistent intermediate states

- The document root can only have one element child.
  - Intermediate state: the document root may have multiple children

Source

```
<b/>
<c/>
```

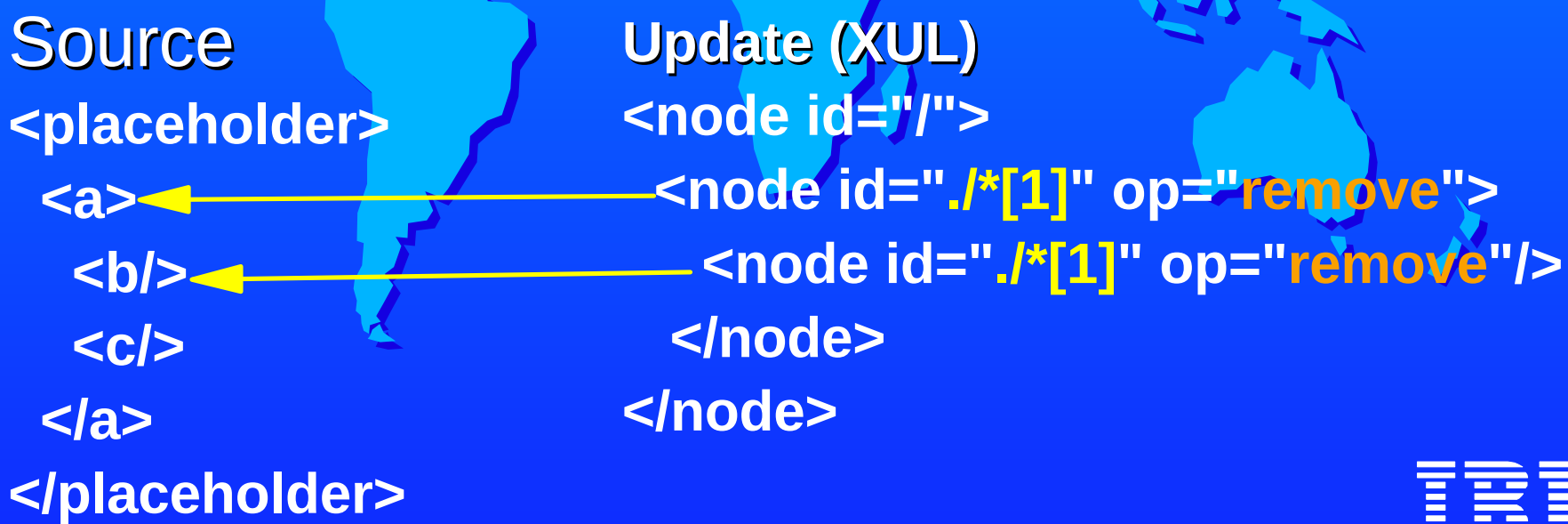
Illegal!

Update (XUL)

```
<node id="/">
  <node id="./*[1]" op="remove">
    <node id="./*[1]" op="remove"/>
  </node>
</node>
```

DOM-related issues: Inconsistent intermediate states

- The document root can only have one element child.
  - Intermediate state: the document root may have multiple children
  - Solution: insert a placeholder root

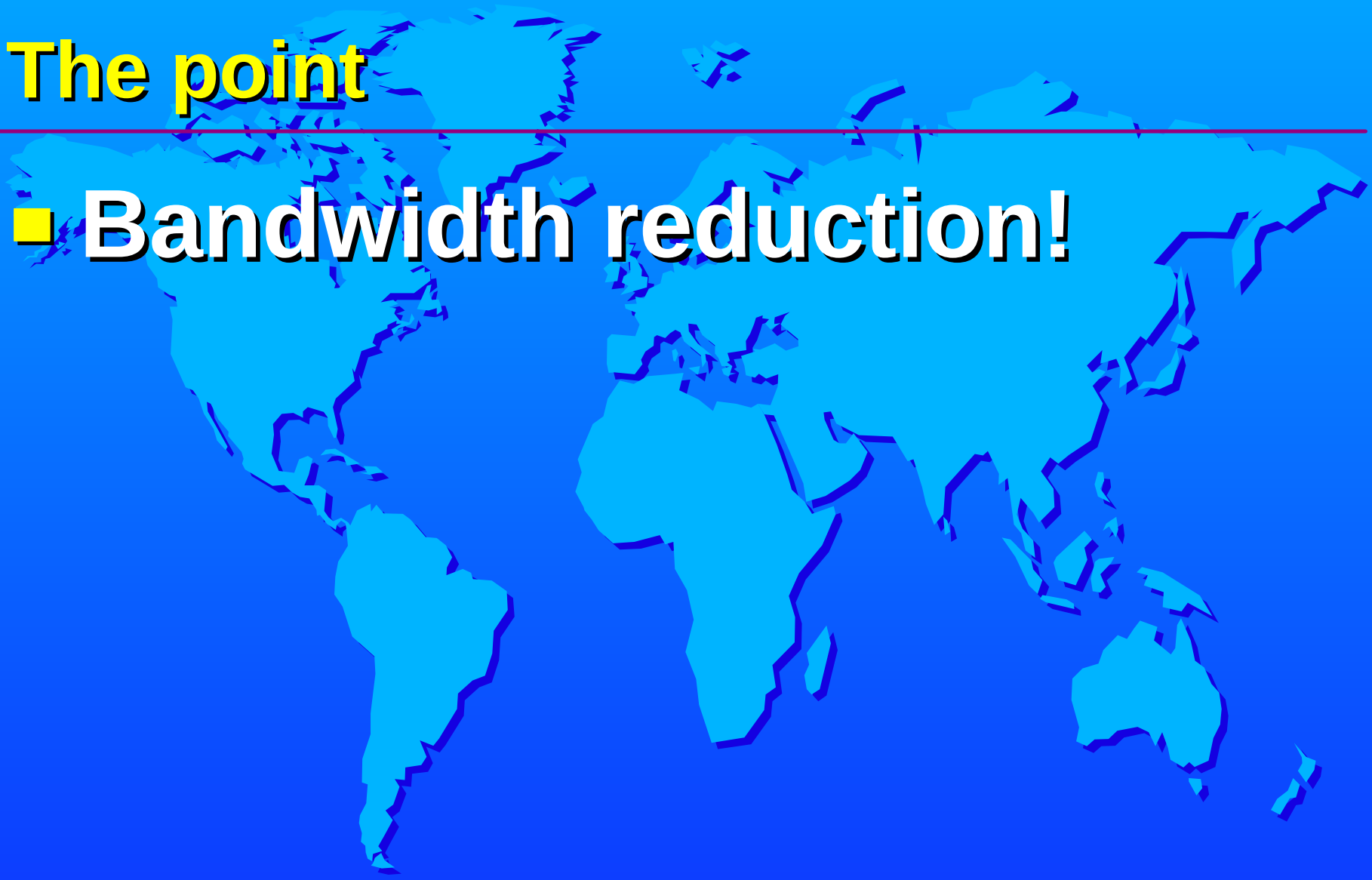


Applications

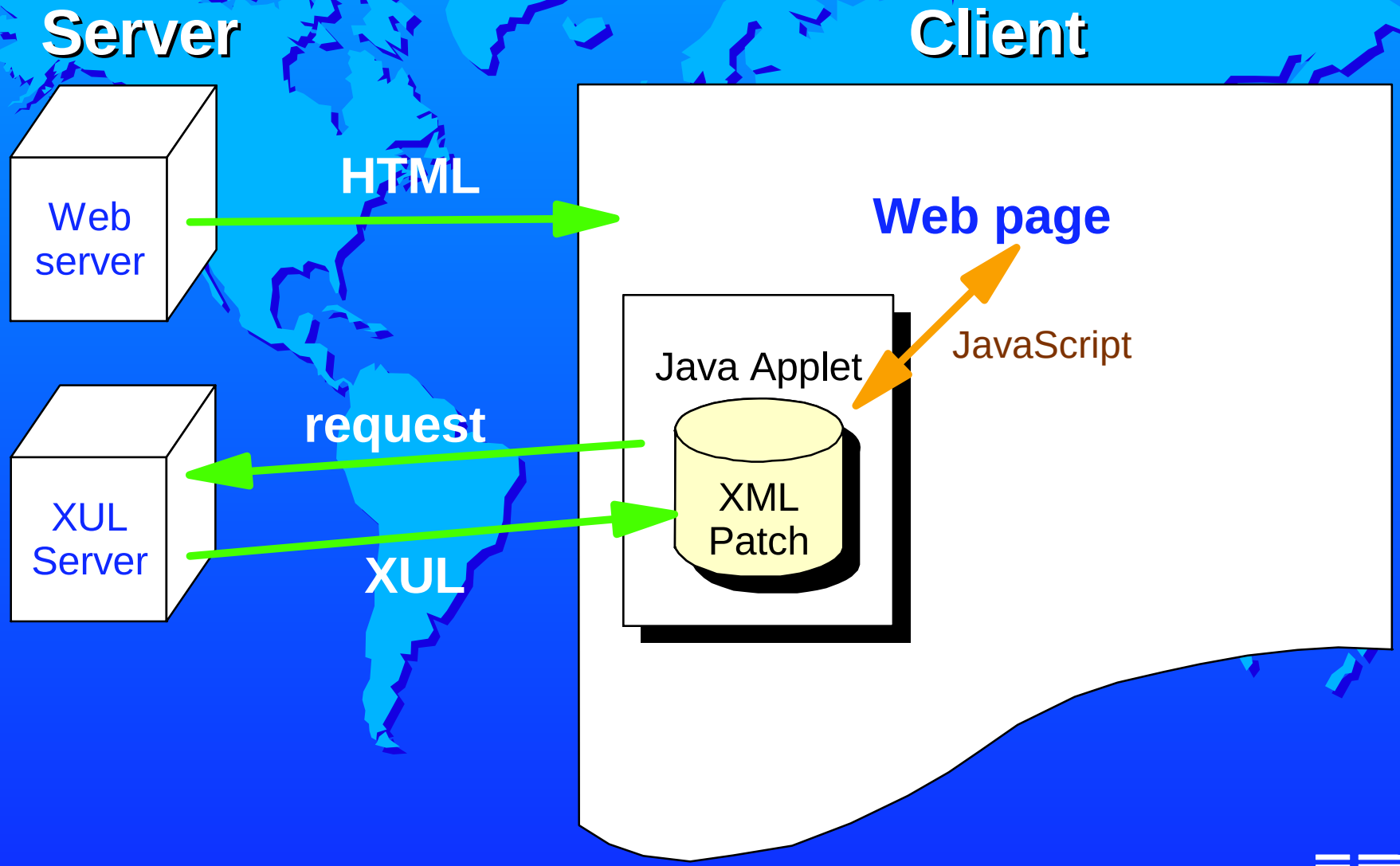
- Update from a central server
  - i.e. web pages
- Version control (like RCS / CVS)
- Collaborative editing of a document
- Selective update using an interactive editor
- Mirroring of large archives

The point

- **Bandwidth reduction!**

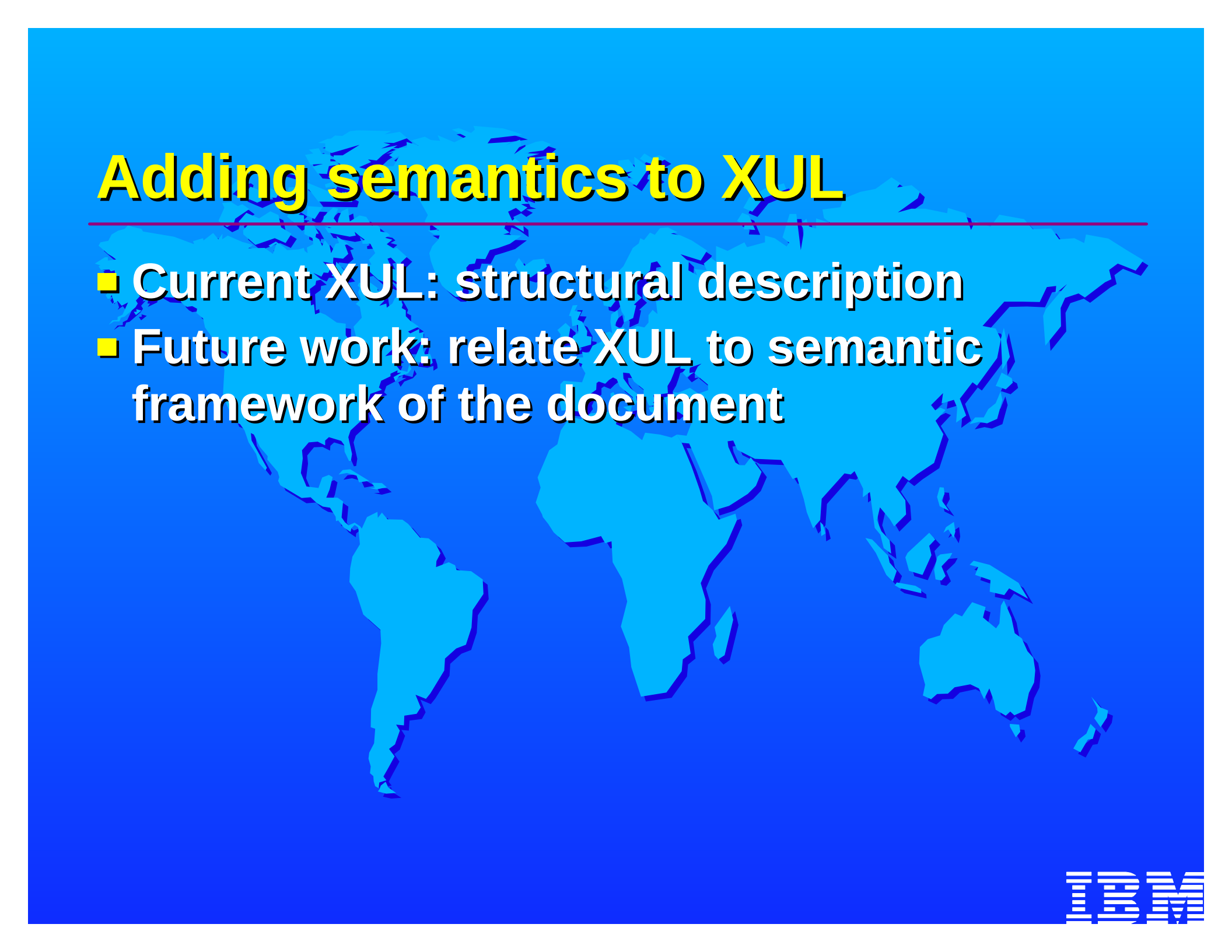


Demo (IBM WebSphere FAQ)



Adding semantics to XUL

- Current XUL: structural description
- Future work: relate XUL to semantic framework of the document



References

- H. Maruyama, K. Tamura, and R. Uramoto, "Digest Values for DOM (DOMHash) Proposal," http://w3e.trl.ibm.com/projects/xml/domhash.htm, June 1998.
- World Wide Web Consortium, "Document Object Model (DOM) Level 1 Specification," http://www.w3.org/TR/REC-DOM-Level-1/, October 1998.
- World Wide Web Consortium, "XML Path Language (XPath)," http://www.w3.org/TR/xpath, July 1999.
- K. Zhang and D. Sasha, "Simple Fast Algorithm for the Editing Distance between Trees and Related Problems," SIAM J. Comput., 18, (6), 1245-1262, (1989).

IBM

XML TreeDiff

- **IBM alphaWorks**
  http://www.alphaworks.ibm.com/formula/xmltreediff
- **Contacts:**
  - **David Epstein**
    epstein@us.ibm.com
  - **Francisco Curbera**
    curbera@us.ibm.com