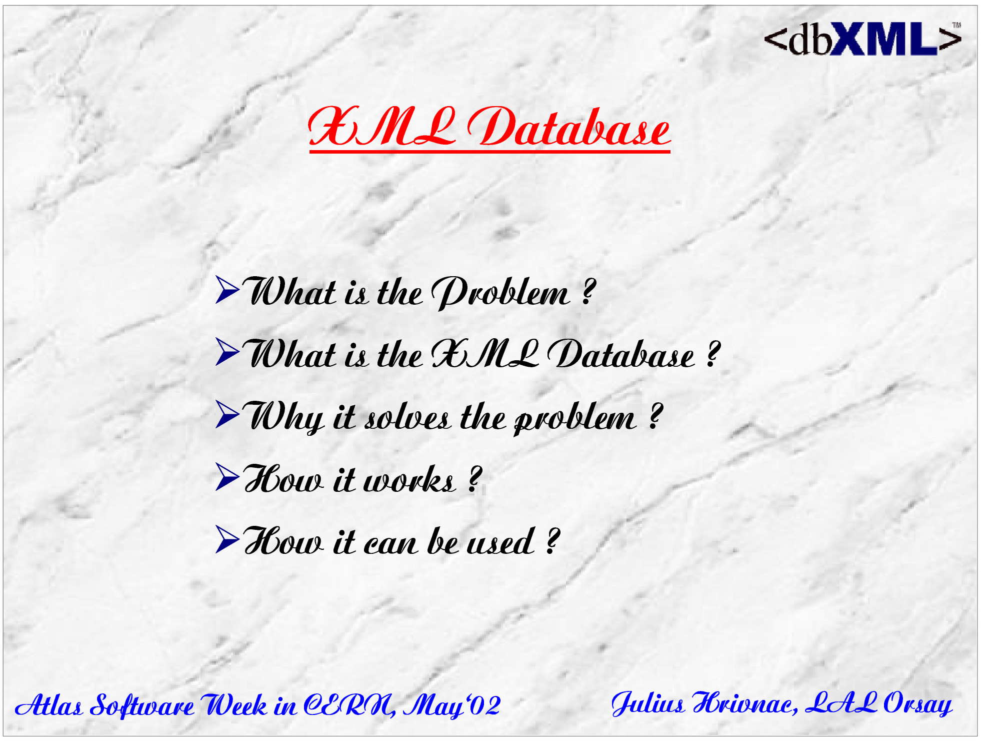


<dbXML>

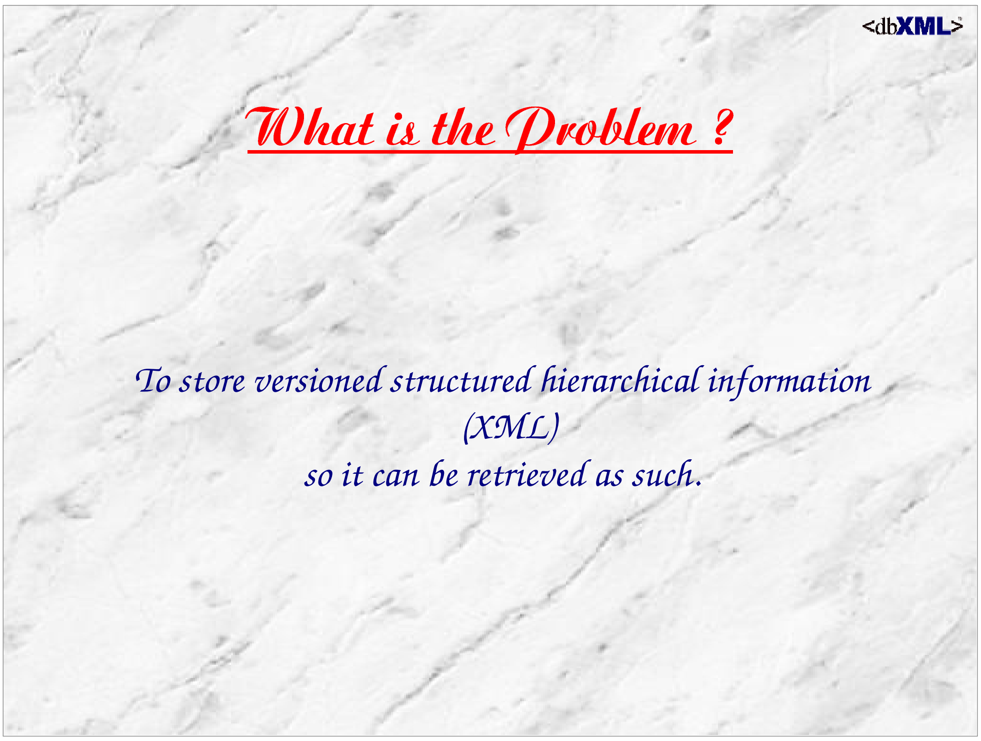
XML Database

- What is the Problem ?
- What is the XML Database ?
- Why it solves the problem ?
- How it works ?
- How it can be used ?

Atlas Software Week in CERN, May'02

Julius Hrivnac, LAL Orsay

- Persistency of AGDD has been discussed a lot, some ad-hoc solutions (RDBS, Strings,...) have been proposed.
- Currently, AGDD files are stored in random places inside CVS.
- There is a natural solution for storage of XML files.

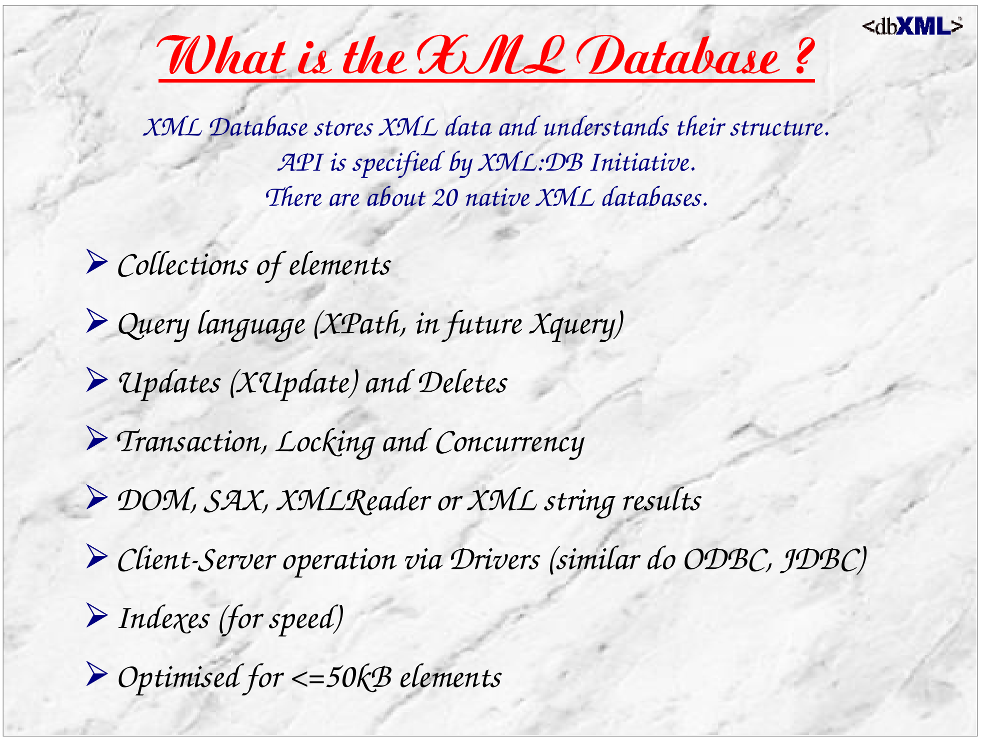


<dbXML>

What is the Problem ?

To store versioned structured hierarchical information (XML)
so it can be retrieved as such.

- Database should understand the structure of the XML elements (it should know what is element name, attribute name, attribute value,...).
- Database should support all usual DB functions (search, commit, update, delete, backup, remote access,...).



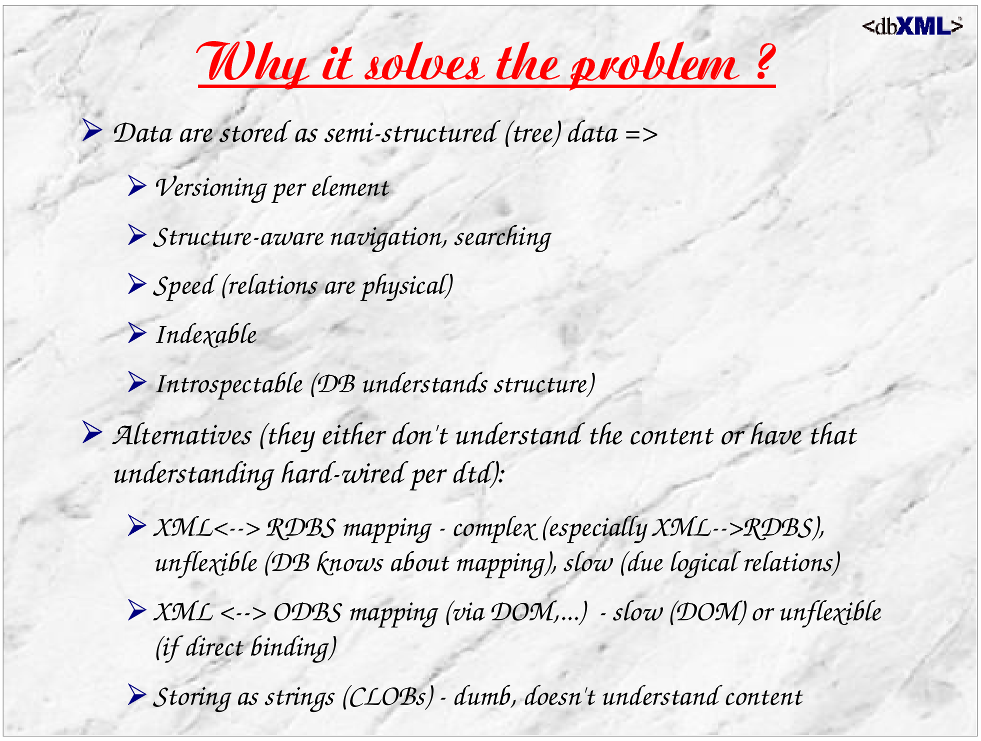
<dbXML>

What is the XML Database ?

XML Database stores XML data and understands their structure.
API is specified by XML:DB Initiative.
There are about 20 native XML databases.

- Collections of elements
- Query language (XPath, in future Xquery)
- Updates (XUpdate) and Deletes
- Transaction, Locking and Concurrency
- DOM, SAX, XMLReader or XML string results
- Client-Server operation via Drivers (similar do ODBC, JDBC)
- Indexes (for speed)
- Optimised for <=50kB elements

- Some XML DBs free. Most DBs in Java.
- Slight differences in API, but big overlap (= API defined by XML:DB).
- Query elements or attributes.

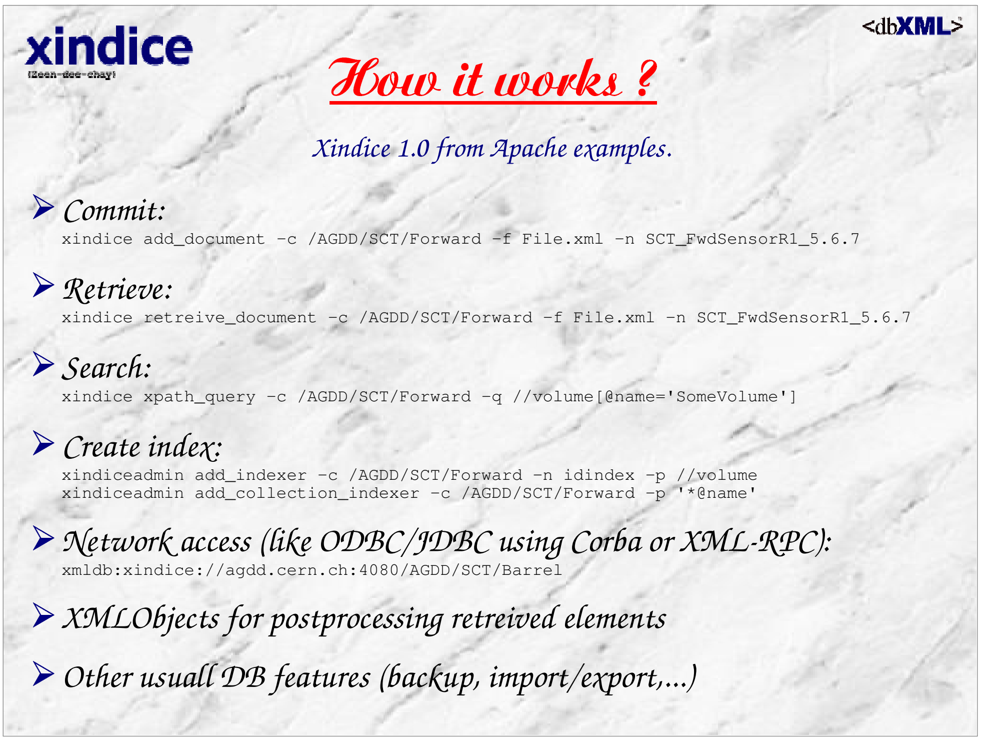


<dbXML>

Why it solves the problem ?

- Data are stored as semi-structured (tree) data =>
  - Versioning per element
  - Structure-aware navigation, searching
  - Speed (relations are physical)
  - Indexable
  - Introspectable (DB understands structure)
- Alternatives (they either don't understand the content or have that understanding hard-wired per dtd):
  - XML<--> RDBS mapping - complex (especially XML-->RDBS), unflexible (DB knows about mapping), slow (due logical relations)
  - XML <--> ODBS mapping (via DOM,...) - slow (DOM) or unflexible (if direct binding)
  - Storing as strings (CLOBs) - dumb, doesn't understand content

- Main advantages are:
-- Flexibility (one doesn't have to change Schema when XML DTD changes) - unlike RDBS/ODBS mapping
-- Functionality (structure is known to the DB so it can act on it) - inlike storage of Strings
- Mapping RDBS --> XML easy and everywhere supported; mapping XML --> RDBS difficult



xindice
(Zeen-dee-chay)

<dbXML>

How it works ?

Xindice 1.0 from Apache examples.

- Commit:
```
xindice add_document -c /AGDD/SCT/Forward -f File.xml -n SCT_FwdSensorR1_5.6.7
```
- Retrieve:
```
xindice retreive_document -c /AGDD/SCT/Forward -f File.xml -n SCT_FwdSensorR1_5.6.7
```
- Search:
```
xindice xpath_query -c /AGDD/SCT/Forward -q //volume[@name='SomeVolume']
```
- Create index:
```
xindiceadmin add_indexer -c /AGDD/SCT/Forward -n idindex -p //volume
xindiceadmin add_collection_indexer -c /AGDD/SCT/Forward -p '*@name'
```
- Network access (like ODBC/JDBC using Corba or XML-RPC):
```
xmldb:xindice://agdd.cern.ch:4080/AGDD/SCT/Barrel
```
- XMLObjects for postprocessing retreived elements
- Other usuall DB features (backup, import/export,...)

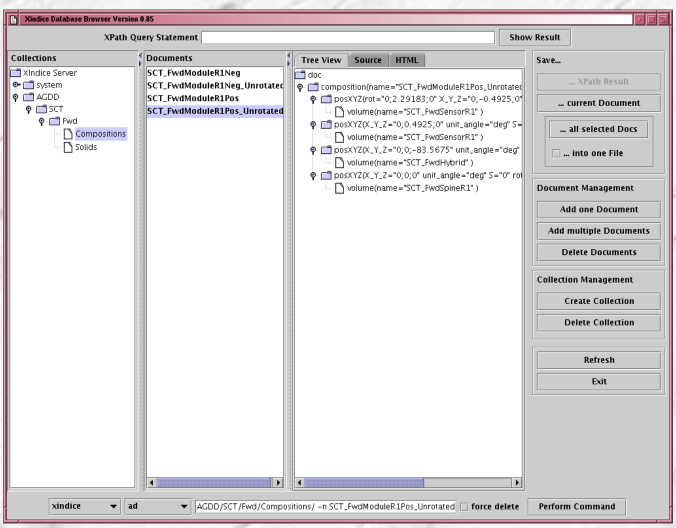
- XMLObjects is an extension to XML:DB.
- Xpath is used in searching.

xindice
(Zeen-dee-chay)

<dbXML>

How it works ?

XindiceBrowser 0.85.



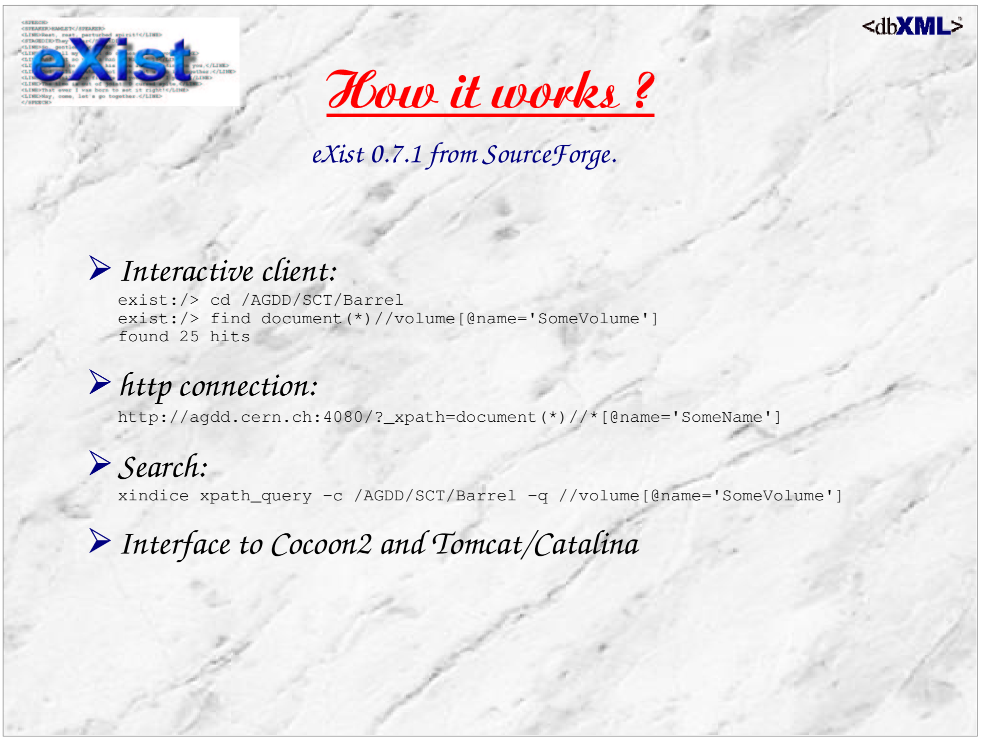
```
XIndice Server
  system
  AGDD
    SCT
      Fwd
        Compositions
        Solids
```

```
SCT_FwdModuleR1Neg
SCT_FwdModuleR1Neg_Unrotated
SCT_FwdModuleR1Pos
SCT_FwdModuleR1Pos_Unrotated
```

```
doc
  composition(name="SCT_FwdModuleR1Pos_Unrotated
    posXYZ(rot="0;2.29183;0" X_Y_Z="0;-0.4925;0"
      volume(name="SCT_FwdSensorR1" )
    posXYZ(X_Y_Z="0;0.4925;0" unit_angle="deg" S=
      volume(name="SCT_FwdSensorR1" )
    posXYZ(X_Y_Z="0;0;-83.5675" unit_angle="deg"
      volume(name="SCT_FwdHybrid" )
    posXYZ(X_Y_Z="0;0;0" unit_angle="deg" S="0" rot
      volume(name="SCT_FwdSpineR1" )
```

xindice | ad | AGDD/SCT/Fwd/Compositions/ -n SCT_FwdModuleR1Pos_Unrotated | □ force delete

- Not very usefull, just allows to understand DB' design and content.



How it works ?

eXist 0.7.1 from SourceForge.

- *Interactive client:*

```
exist:/> cd /AGDD/SCT/Barrel
exist:/> find document(*)//volume[@name='SomeVolume']
found 25 hits
```

- *http connection:*

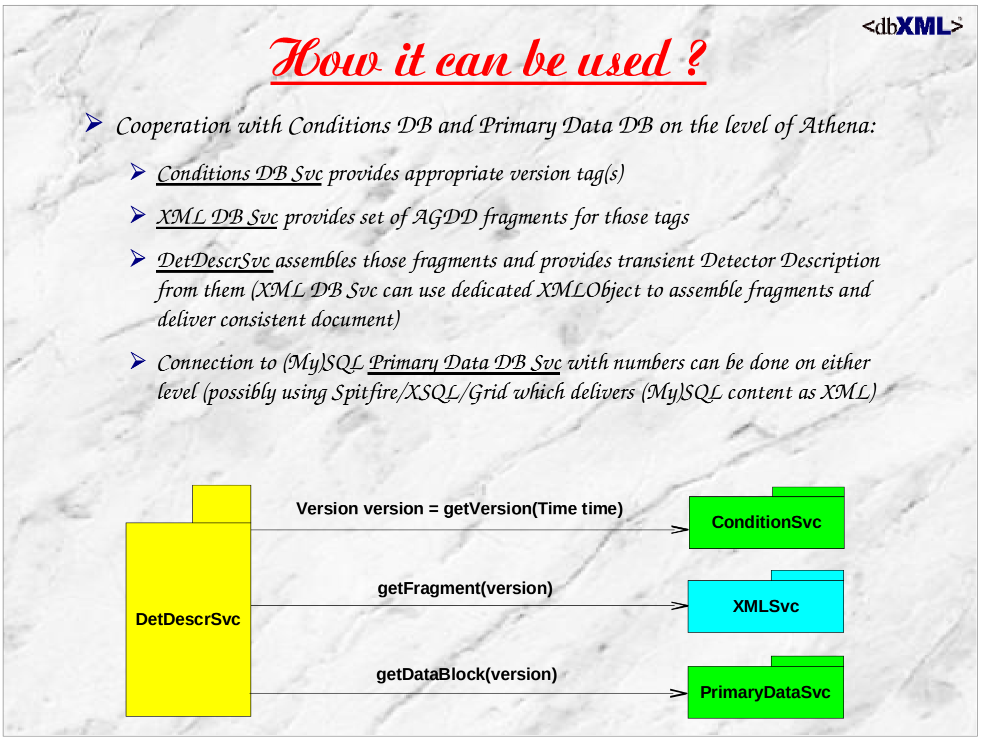
```
http://agdd.cern.ch:4080/?_xpath=document(*)//*[@name='SomeName']
```

- *Search:*

```
xindice xpath_query -c /AGDD/SCT/Barrel -q //volume[@name='SomeVolume']
```

- *Interface to Cocoon2 and Tomcat/Catalina*

- Using extended XPath.
- Very good integration with Tomcat Web Service tool.
- Othewise similar to Xindice.
- Both Native XML and RDBS backends exist. RDBS backend may disappear as it is slower and less functional than native XML one.



<dbXML>

How it can be used ?

- *Cooperation with Conditions DB and Primary Data DB on the level of Athena:*
  - *Conditions DB Svc provides appropriate version tag(s)*
  - *XML DB Svc provides set of AGDD fragments for those tags*
  - *DetDescrSvc assembles those fragments and provides transient Detector Description from them (XML DB Svc can use dedicated XMLObject to assemble fragments and deliver consistent document)*
  - *Connection to (My)SQL Primary Data DB Svc with numbers can be done on either level (possibly using Spitfire/XSQL/Grid which delivers (My)SQL content as XML)*

- Conditional DB stores conditions, XML DB stores DetDescr structures (topology), (My)SQL DB stores concrete numbers (dimensions).
- Fragments from XML DB should be processed:
-- Assembled
-- Filled with data from (My)SQL
-- Expanded with respect to Arithmetics
-- Expanded with respect to Compact elements
All those steps can be done in various places, often by XSLT stylesheets (already existing).
- Green: (My)SQL, Blue: dbXML

How it can be used ?

- DbXML runs as a server either serving a distributed (Grid) environment or just private notebook.
- Server installation and management (start/stop) is trivial.
- Data export/import/backup is trivial. Standard data can be part of the Release.
- Embeded (integrated) operation is possible too.
- The storage hierarchy and granularity should be agreed on

- Server installation: just copy.
- Each Linux Notebook runs already tens of Servers, this is just another one.
- Versions are part of fragment' name (like SCT_Wafer_R5_1.2.3).