Technical University of Braunschweig

Institute of Operating Systems and Networks

Student project work on

Distributed Calculation Object Models (COMs)

Syed Buturab Imran
Candidate for Master of Science in Computer Science

Supervised by

Martin Gutbrod

Prof. Dr.-Ing. Lars Wolf
Dec 2005, Braunschweig

ABSTRACT

Distributed Calculation Object Models (COM) is semantic description to calculate the costs of Calculation Objects (COs) in a distributive environment. COs are objects whose cost allocation can be defined as quantities, percent or allocation. Since the instances of COM exist in relational database management system (RDBMS), having the drawback that the whole responsibility for the system is attached to one person or institution and to calculate the cost of all COMs rights to their API and the databases backend are needed.

A much better scenario could be build on COs, which are stored on several hosts, e.g. in a department, institute, supplier, service provider or cost centre. To calculate a COM in a distributive environment, a client application loads the entire COM relevant (linked) COs and calculates their values.

Furthermore the data is stored in eXtensible Markup Language (XML) based databases rather than relational databases. As it will be easy for a client to communicate XML data over networks and provides good programming language support.

ACKNOWLEDGEMENT

I express my sincere thanks to my advisor Prof.Dr.-Ing. Lars Wolf for his continuous and encouraging support throughout the work. As always, his guidance is appreciated.

My special thanks go to my co-advisor Martin Gutbrod, for his kind guidance and support. He was always there to talk about my ideas, and to ask me good questions to help me proceed further in the project. I thank him for providing me all necessary materials to carry out this project.

I express my solemn gratitude to all my co-workers who shared their knowledge during the work. I would not have accomplished this task without the assistance from many people who gave their support in different ways. To them I would like to convey my heartfelt gratitude and sincere appreciation.

I thank my parents, brothers, sister and my wife for their support, encouragement, patience, and belief in me. They nourished my interests, even when the interests went beyond borders. I thank all my friends Ayyappan, Safdar, Sajjad and especially Gopi danda for supporting me in various aspects.

TABLE OF CONTENTS

TABLE OF FIGURES

1 INTRODUCTION

A Calculation Object Model (COM) based on the calculation objects. It can be created using different programming languages and in our case we have used Java on the Eclipse Java platform. These calculation objects resides at server side, which when initiated get the values from the clients and perform the required functions of calculation. The values (data) these COMs gets are stored basically in xml based database of the different Calculation Objects (COs) which act as a client when passing the values forward and as server when getting the values from other objects. In the case of databases, traditional database is always been in practice i.e. RDBMS but due to having a drawbacks xml database has been preferred.

COM is semantic description to calculate the costs of things in complex scenarios, where we have many Calculation Objects.

When we talk about databases, presently we use a relational database, where every thing looks like a table with rows and columns but in reality, however is more complicated than the data to be in table format only. Data that needs to be stored does not always exist in tabular format and required to be benefit from tools that are more closely fits its natural structure.

Generally the traditional databases and in particularly SQL databases have been so incredibly successful, that they have completely eliminated the competition. Infect, the relational databases fit a lot of problems very well but they don't really fit for eXtensible Markup Language (XML) document's data. Nowadays a great deal of data is being encoded in XML documents and more is being created everyday, thus felt the need for something better.

1 XML Database vs. RDBMS

Mapping XML documents into file systems causes difficulties in managing the structures formed by the document and also tends to lead to the loss of details i.e. element order, instruction for processing, comments, white space, and other elements those are important in many applications.

RDBMS suffered with the problem of scalability, as they degrade very quickly due to the storage of thousands of documents. Another problem RDBMS faces is the lack of structured queries since the full text search engines are not XML aware and the queries over metadata (which are extracted and stored in relational database) were limited to few fields. Synchronization between the databases is also a big problem.

1.1 XML Database vs. RDBMS [1]

Relational Database	XML database
➢ Contain Tables (columns & rows). ➢ Tables contain records with same schema. ➢ Records are unordered list of named values. ➢ Unordered set of records are retrieved by SQL query.	➢ Contains collections ➢ Collection contain xml documents with same schema ➢ Tree of nodes ➢ Ordered sequence of nodes

2.2 Need a XML Database

XML database treats XML documents and its elements as the fundamental structure rather than records or tables. Native databases can significantly outperform traditional RDBMS for the task that involves load of document processing, website management and also services like web services [1], because XML databases have a number of features that are useful for working with document-centric XML i.e. XML data model (flexible enough to model documents), XML-aware full-text searches and query languages.

Having these features allows documents to be stored in and queried in a single location rather than multiple locations connected by means of relations. Also these features allow us to have node-level updates which reduce the cost of updating.

One of the most interesting factors is that that the XML model stores and makes available information about the structure of the data - as well as just the data - through the same interface. XML database simply keeps all its data content in one easily searched and manageable place providing the capability of having everything in one place. The data in XML database can be accessed and edited by many different users with varying levels of privileges across heterogeneous systems. Well-implemented XML database are simply faster than queries over documents stored in a file systems by assigning sequence numbers to each node so that it knows the position of each node and can compare the document order of two nodes in constant time. The performance of XML databases are improved as of the reason that the database has essentially pre-parsed each document when storing it, thus does create the need to check the document that query has accessed. XML database provide Enterprise Information Integration(EII), that is a unique approach to

information integration that provides users with a real-time, a single view of information from disparate data sources across and beyond enterprise, where data from a variety of internal and external sources and in different formats can be taken and treated as a single data source.

3 XML DATABASE

XML has been emerged as standard for data representation, exchange and storage. It has tags on data elements identifying the meaning of the data, these tags specify how the data should be formatted, provided the relationship between the data items. Applications encoding there data in XML can quickly make their information available in a simple and usable format, which can further be interoperable.

The parent element tag of an entire relationship tree in XML is the root element and its child elements. All the information is stored between the starting tag and ending tag and the data between the tags can consists of several kind of content. Given in figure 1 is the example written in XML in my project.

```
<?xml version="1.0" encoding="UTF-8"?>
<DCOM xmlns="http://www.lk2.ifalt.com">
     <!-- Travel.xm -->
     <CalculationObject ID="3">
          <Name> Tickets </Name>
          <Description> Comfortable, Extra space for language
</Description>
          <Number>88</Number>
          <Cost>48</Cost>
          <DriverID>3</DriverID>
     </CalculationObject>
.
.
.
</DCOM>
```

Fig 1: XML Syntax

XML document written by an editor is stored further in XML database and when for the data retrieval XML parsers are used, which perform the process of reading

the document. XML parser performs the extremely important task of taking a raw XML data as input and making sense of it. The output of XML document being parsed can be easily be manipulated and handled using XML tools or java API in our case. Java APIs allows XML the use of XML data directly within Java code.

For application point of view the document stored in XML database is parsed and later manipulated as shown in figure 2.

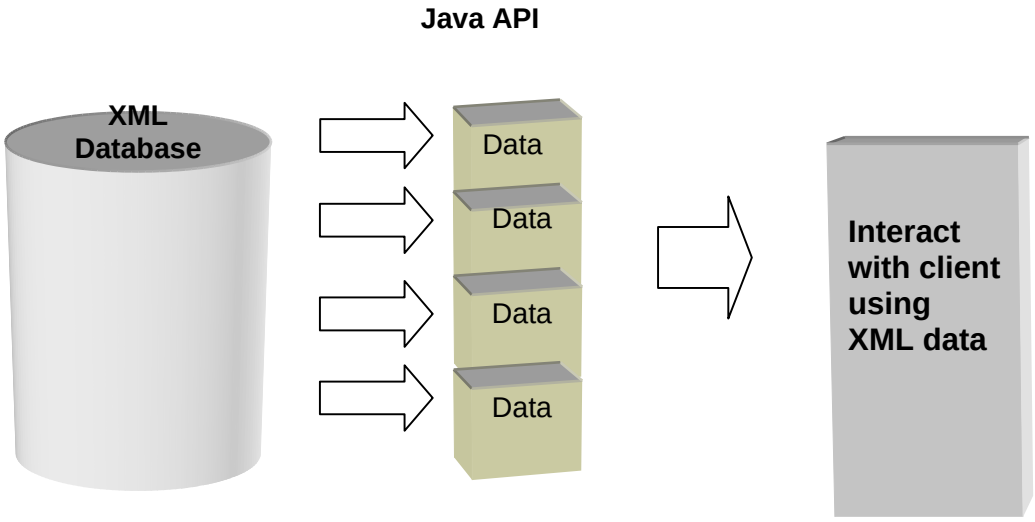

Fig 2: Retrieving data from XML database

Parsing XML documents into "information sets" or document trees and then scanning those trees to extract data. The data retrieval access can either be complete or can be selective. After retrieving the data from database, the parser passes the data to the browser to display or some other application to manipulate it.

3.1 XML Parsers

The parser is the most basic yet important XML based tool, almost every XML

application includes parser. It is a software component which sits between the application and XML files. The Parser is a module that reads and interprets the source code.

The Java API for the use of processing xml document makes it much easier to process the data in XML using application written in Java programming language [2] to read and manipulate XML we need a XML parser. The two most widely used APIs are SAX (Simple API for XML Parsing) and DOM (Document Object Model). Data can be even parsed as a stream of events or to build a tree-structured representation of it. The Java API for XML processing (JAXP) also now support the XSLT (XML Stylesheet Language Transformation) standard which gives control over the presentation of data and enabled to convert the data to other XML documents or to other formats as well as provide the namespace support, which are a convention for associating a URI with elements and attributes, creating an extensible naming framework in place of the original flat namespace.

Further the XML document is read by the parser and populates the tree in the memory. The tree build by the Parser is exact match of the tree in the XML document. [5], Thus the application manipulates it as if it was the XML document and infect for the application it is the XML document. In our project we use DOM as parser.

4 CONVERGENCE OF JAVA AND XML

XML when combined with Java provides a powerful tool for integrating data from diverse applications. Combination of Java and XML data can be sent across the network and between users regardless of platform or operating system. As java is portable code and XML is portable data. Match of XML and Java fills in the gaps in the application development picture. Convergence of Java and XML provides the ability to represent input and output to the application with a system independency, where the data is portable. Both of these languages are designed to be used in distributed systems with Java increasing the application portability and XML enhancing data portability across platforms and user access method. They both create standalone programs, which are Self-contained and usually independently operating.

Abilities of parties to communicate with each other even if they are using different information systems and different data formats is essential as Web services are dependent on it. Since the XML is a key technology addressing the ability of data portability, as a result XML is increasingly being used for enterprise integration applications. Furthermore, The Web services also depend on the ability of using different computing platforms to communicate with each other, thus by using java platform which makes the code portable has become the natural choice. It has become more interesting as the new Java APIs for XML is available, giving the convenience of data portability and code portability.

The use of XML and Java platform has become an ideal combination and plays a central role on web based services

JAVA + XML = Portable Language + Portable Data

Convergence of Java and XML is essential for business to business information interchange with synchronous data messaging and for the content generation in the context of Java application and web services development. That's why, because of the tremendous capabilities of this combination offer, many companies are looking to XML and Java as their stepping stone to an e-business data integration solution [12]. Furthermore, Java provides the most robust set of APIs, parsers, processors, publishing frameworks, and tools for XML use of any programming language.

4.1 DOM Parser

DOM is standardized and defined by the W3C Working group. It is a set of interfaces for building an object representation in the form of a tree of a parsed XML document. The tree data structure resulting by the DOM parsing can be manipulated with DOM methods. DOM parser even allow random access to particular pieces of data in a XML document and can build the object representation of the document and manipulate it in memory by adding new elements or deleting an existing one. Although by using the DOM Parser, the entire XML document has been read into memory by storing all the data in nodes, in this way the entire document is very fast to access as it is all in memory for the length of its existence in the DOM tree, where as each node represents a piece of the data pulled from the original document. Having the whole document object model in memory makes DOM parser generally less efficient for searches involving just few items especially when the documents are large which is the main drawback of using DOM and

degrades the performance for large and complex documents but making fast search in memory.

DOM parser uses the `javax.xml.parsers.DocumentBuilderFactory` class as in fig 3 to get a `documentBuilder` instance and then by using that instance to produce a Document object that conforms to the DOM specification.

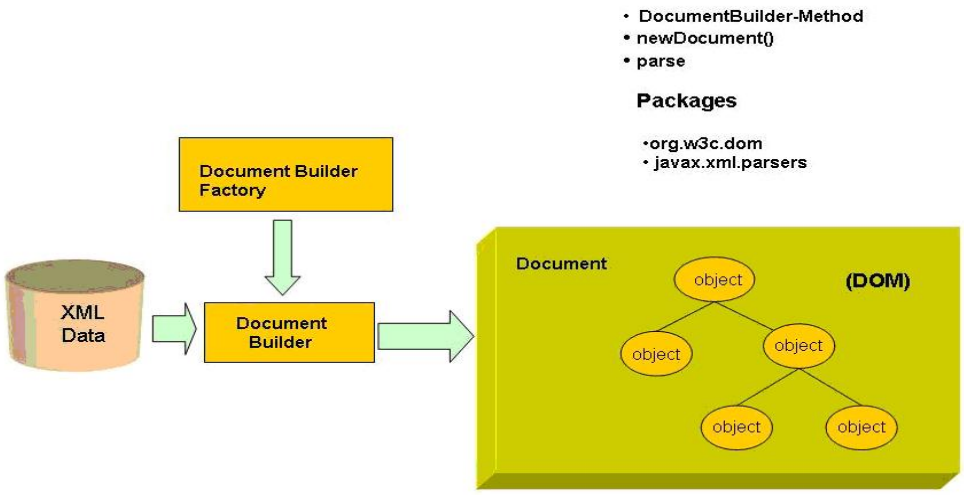

Fig 3 : DOM Parser

In fact, the builder is determined by the system property `java.xml.parser. DocumentBuilderFactory`, that selects the factory implementation used to produce the builder. Also by using `DocumentBuilder newDocument()` method to create an empty Document that implements the `org.w3c.dom.Document` interface. Thus one of the builder's parse methods to create a document from existing XML data can be used [11].

DOM parser is defined in terms of interfaces rather than classes. There is no need to install DOM interfaces on their own as they come bundled with parser distribution that provide the detailed implementation classes [9].

As DOM Parser treats XML document as a tree made up of nodes of several types, where the tree has single root node and all the nodes in the tree except the for the root have the single parent node. Nodes have the list of their child nodes and in the case the list of the children is empty the node is considered as leaf node.
In the case of an Attribute node they are not considered as a part of tree structure though they belongs to one element node but they are not to be considered as child node. That is why the whole DOM document is composed of a tree of nodes.
Attribute *nodeName*, *nodeValue* and *attributes* are included as for mechanisms to get the information of the node without casting down to specific derived interfaces [10].

There are twelve types of DOM nodes, seven of which can potentially be part of DOM tree:

- **ATTRIBUTE_NODE**
  The node is an Attribute.

- **CDATA_SECTION_NODE**
  The node is a CDATASection.

- **COMMENT_NODE**
  The node is a Comment.

- **DOCUMENT_FRAGMENT_NODE**
  The node is a DocumentFragment.

- **DOCUMENT_NODE**
  The node is a Document.

- **DOCUMENT_TYPE_NODE**
  The node is a DocumentType.

- **ELEMENT_NODE**
  The node is an Element.

- **ENTITY_NODE**
  The node is an Entity.

- **ENTITY_REFERENCE_NODE**
  The node is an EntityReference.

- **NOTATION_NODE**
  The node is a Notation.

- **PROCESSING_INSTRUCTION_NODE**
  The node is a ProcessingInstruction.

- **TEXT_NODE**
  The node is a Text node.

The code for the DOM Parser as been used in the project to parsed the XML data file is as given in figure 4.

```
DocumentBuilderFactor docBuilderFactory =
DocumentBuilderFactory.newInstance();
DocumentBuilder docBuilder =
docBuilderFactory.newDocumentBuilder();
doc = docBuilder.parse(new File("Travel.xml"));

...

Node rootNode = doc.getDocumentElement();
NodeList list = doc.getElementsByTagName("cost");
```

Fig 4 : DOM Parser code

When parsing the XML documents it is important to remember that the XML parsers are strict as they complain for errors. Thus to minimize the risk of errors in XML documents, the XML document should be validated using XML editors.

5 CALCULATION OBJECTS

Dimensional Calculation Objects (COs) deals with special mathematical modeling where individual cost methods can be assigned freely which are interlinked with other objects makeup the whole scenario of cost calculations at different levels and depth.COM facilitates us with transparency of overhead costs resides in distributed resources in heterogeneous platform. COs quits the traditional way of thinking of hierarchical cost structures but provides the level interlaced structure.

COs are reside at server side, which when getting the data values from the client perform the required function of calculations of the cost in the form of percent, amount and capacity. These calculation objects at different levels do the calculation and then finally they are interconnected to the other Cos that sums the project or scenario in which we working at.

One of the main advantages of the calculation object is to get the required allocation thus satisfying the requirement of the cost allotted, percentage needed or amount with the cost framework.

The most important advantages of COM are as follows [4]

- More transparent figure of material connections
- The use of interlaced objects in the place of hierarchical cost assignments
- Provide simple acquisition and usability
- Provide high flexible and variable cost assignments
- Provide a flexible structure of the costing objects compared with rigid cost centers
- Provide simple reference modeling of standard scenarios

In fig 5 is given a well descriptive diagram of CO and its possible methods like calculating the percentage, cost, amount, or some user defined method.

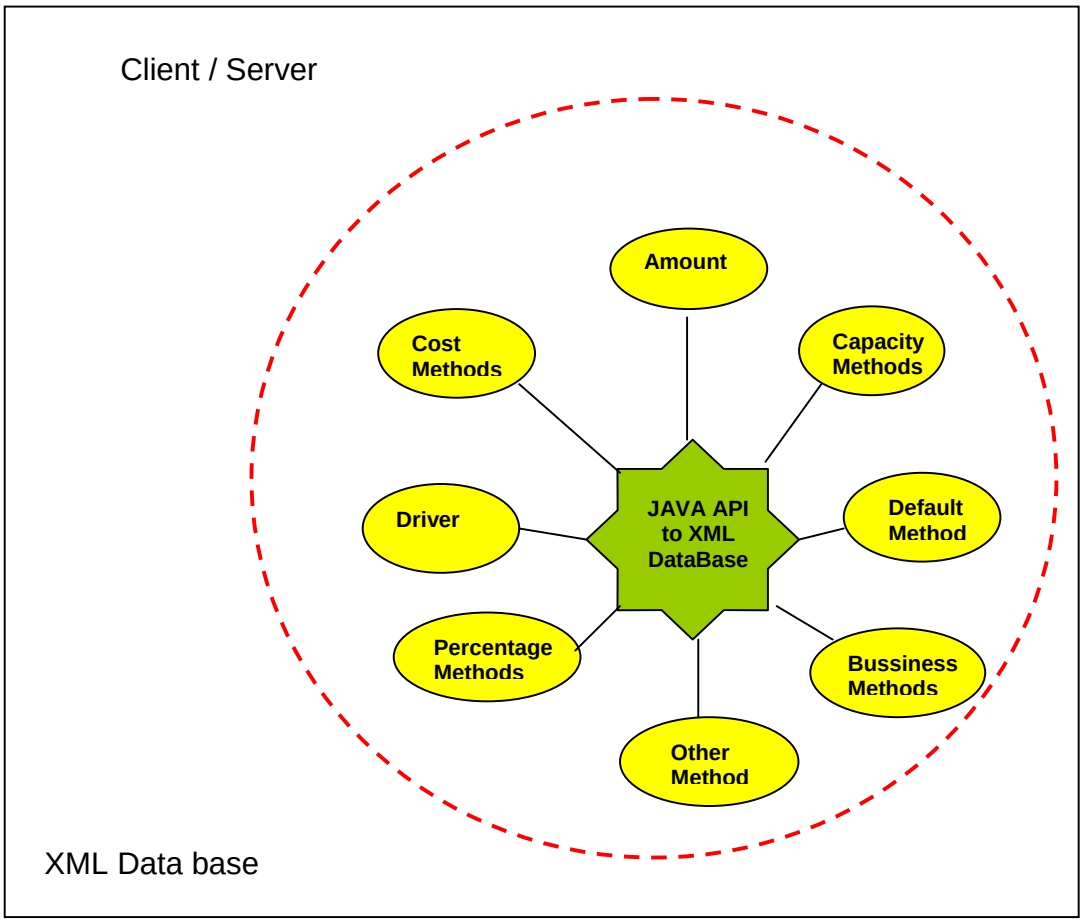

Fig 5: Calculation Object

The Calculation Object Model (COM) is an important step to calculate the cost by improving the transparency of classical allocations of cost by using the methods of allocation of cost. It permits fully a new kind of controlling which simplifies the cost issues in complex systems significantly by allocation of cost. Thus the cost setting of COM is fulfilling the task to find the optimum expected cost as well as measuring. It allows having at every level the transparent view of cost, even it allows relatively small cost not to be neglected.

Further the xml data file used in this project with having allocation of different XML base data files used in the project to calculate in cost at every level. By having the dimensional COM which provide us with the transparency by using different methods we can calculate the cost, percent or capacity in one level interlaced structure and simplifying the cost issues in complex systems significantly as shown in fig6.

```
<?xml version="1.0" encoding="UTF-8"?>
<DCOM xmlns="http://www.lk2.ifalt.com">
<!-- Trip.xml-->
<Allocations ID= "1">
<StartCO
base="http://www.resturant.com/Resturant.xml">1</StartCO>
<EndCO>6</EndCO>
<Quantity>20</Quantity>
<AllocationType>Capacity</AllocationType>
</Allocations>

<Allocations ID="2">
<StartCO base="http://www.hotel.com/Hotel.xml">2</StartCO>
<EndCO>6/EndCO>
<Quantity>25</Quantity>
<AllocationType>Cost</AllocationType>
</Allocations>

<Allocations ID="3">
<StartCO base="http://www.travel.com/Travel.xml">3</StartCO>
<EndCO>6</EndCO>
<Quantity>25</Quantity>
<AllocationType>Quantity</AllocationType>
</Allocations>
.
.
.
<CalculationObject ID="4" >
<Name> Total Trip Cost </Name>
<Description> Including travelling, hotel, dinning
</Description>
<Number>1</Number>
<Cost></Cost>
```

```
<DriverID>1</DriverID>
</CalculationObject>
.
.
</DCOM>
```

Fig 6: Calculation Object and Allocations

5.1 Illustration of COM in XML

XML is a markup language for document containing structured information [6] where Structured information contains both content (words, pictures, etc.). The Markup language is a mechanism to identify the structure in document. CO is designed to describe data and focus on what data is, CO allows defining its own tags and its own document structure [7] according to the XML standards. Tags in the example above (like <DCOM> and <Allocations>) are not predefined in any XML standard but they are invented by the author of CO. Furthermore, it has been created to structure, store and send information or exchange data between incompatible systems, which solves the problem of data incompatibility in real world in computer systems and databases as it has been always a challenge for developers to exchange data over the internet. XML format used in CO greatly reduces the complexity when the data is to be read by many different types of applications and it can also be used just to store data in files or in databases. There are very strict rules for XML syntax, though they are easy to learn as XML documents use self-describing and simple syntax as used in COM.

First line in COM is XML declaration, defining the XML version and character encoding used in the document. It above case the document conforms to the 1.0 specification of XML and uses the "UTF-8" character set.

```
<? xml version="1.0" encoding="UTF-8"?>
```

Next line of the COM describes the root element <DCOM> with an attribute of xml namespace giving the reference of the document, where namespace provide a simple method for qualifying element and attribute names used in Extensible Markup Language documents by associating them with namespaces identified by IRI references[14].

```
<DCOM xmlns="http://www.lk2.ifalt.com">
```

In COM the "Allocation" is the designation of the referenced COs where by declaring the type of an attribute like "ID" [8] we make sure that when parser will validate the document the ID value must be unique for the whole COM described in the XML file

```
<Allocations ID= "1">
```

Further, we have described in COM, the starting point "<StartCO>"of our calculation with an attribute value of "base " in the case if CO does not exist in the same XML file and the ending point of the calculation "<EndCO>" .Thus giving the range from where to start calculation and where to end . By using this definition CO allocates the cost from other COs in by improving the transparency of classical allocations of cost by using the methods of allocation of cost

```
<StartCO base="http://www.resturant.com/Resturant.xml">
<EndCO>
```

Further the "AllocationType" makes a substantial contribution to the flexibility and adaptability of COs. It indicates the methods, in which the Calculation Object is to be measured. The methods used here are Capacity, Cost or Quantity.

```
<AllocationType>
```

The "DriverID" represents as an identification number that can be referenced to this Calculation Object which represent itself as the main Calculation Object calling other Calculation objects.

```
<DriverID>
```

The worth of the calculation objects is the "Quantity" or "Cost" or "Capacity", which is to be calculated given by each Allocation by the method of their AllocationType starting from "StartCO" and ending its calculation at "EndCO".

6 L- K2 Scenario

L-K2 [2] scenario which acts around a new calculation model of education scenario, It is based on flexible accounting of individual COs, the similar characteristics as individual calculation positions possesses (designation, quantity, price).

An object however can be so quite everything that has data and values and the values to be determine. How finely one stretches the model, depends substantially on which transparency of the costs one would like to have and how it must be presented with creation of value chain.

In the reality several hundreds (and more) CO can occur in COM. In fig 7 is given a very well descriptive diagram about these objects and how the interrelate with each other. Web of Cost diagram used in L-K2 project, explains the mechanism of COs used in an Educational Costing Scenario [3].

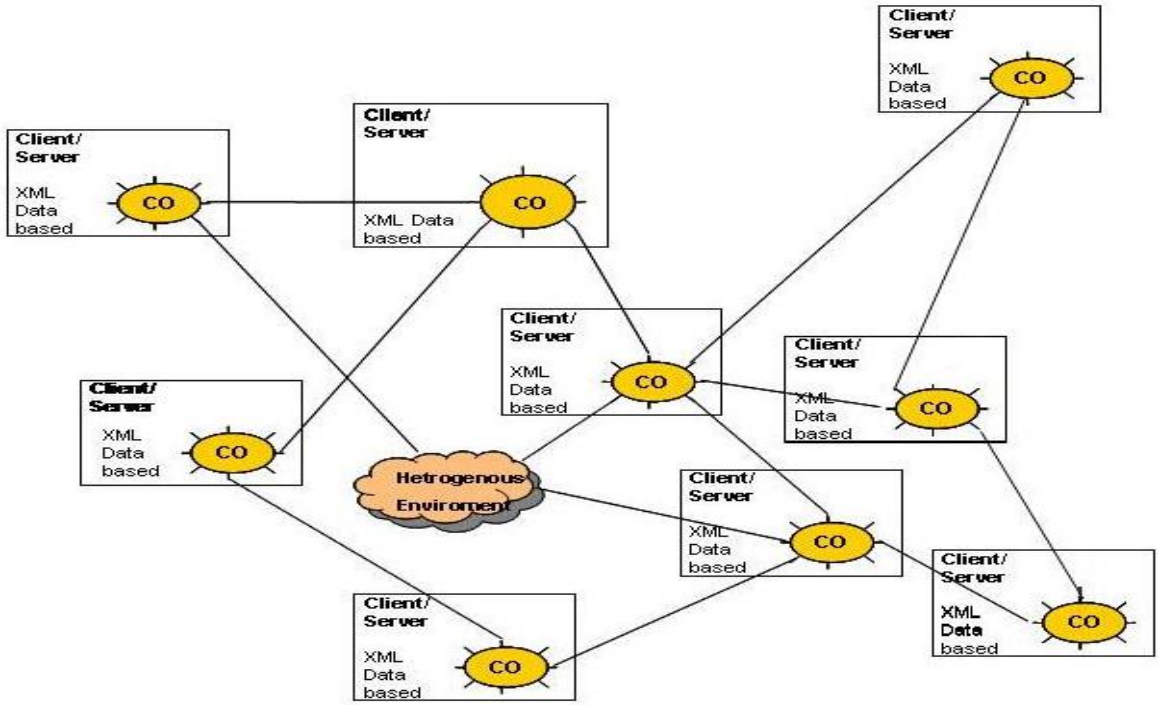

Fig 7: Network of Calculation Objects

COM as being used in L-K2 project as a standardized scenario to calculate the cost of different educational scenarios, which serves as basic module for the implementation and it shows the transparency of cost at every level residing in distributed resources in heterogeneous platform. The educational measures improved by the implementation of COM of educational measures, as it aims for the transparency of classical cost factors and can be set up as a prototype for the process of calculation of models and to be considered to develop further models.

7 CONCLUSION

By using the concept of Calculation Object Modelling, we can make a business scenario for cost control, management, and purchase order like cases. The main advantage we get is the clear picture of cost at every steps of processing in the form of percent, amount or cost. It does not need traditional databases but only with client side using xml based technologies and the parsing is done by using DOM parser in java.

CO concept, where all the data is stored in XML database makes it very adaptable and portable to the requirement and needs, and solve the chain cost control issues in planning of some project. It gives us enough compatibility for further development and integration with other business objects.

8 ABBREVIATIONS

API Application process interface

CO Calculation Object

COM Calculation Object Model

DOM Document Object Model

JAXP Java API for XML processing

RDBMS Relational Database Management System

SAX Simple API for XML

SQL Sequential Query Language

XML Extensible Markup Language

XQuery XML Query Language

XSLT XML Stylesheet Language Transformation

W3C World Wide Web Consortium

9 REFERENCE

[1] **Elliotte Rusty Harold** Adjunct Professor, Polytechnic University, *Managing XML data: Native XML databases*, 06 Jun 2005

[2] **Eric Armstrong, Stephanie Bodoff, Maydene Fischer, Dale Green and Kim Hasse** *The J2EE Tutorial for the Sun ONE Platform,* Feb 13, 2003.

[3] **Martin Gutbrod** *L-K2 Education Costing*, 2005.

[4] **Ulf-Daniel Ehlers & Peter Schenkel** *Bildungscontrolling im E-Learning, Springer-Verlag Berlin Heidelberg*, 2005.

[5] **Benoit Marchal** *The Parser and DOM,* Sep 20, 2002.

[6] **Norman Walsh** *A Technical Introduction to XML* , October 03, 1998.

[7] **Refsnes Data** *W3Schools. XML Tutorial*, 2005.

[8] **W3C Recommendation** *xml: id version 1.0,* September 09, 2005.

[9] **Elliotte Rusty Harold** *Processing XML with Java,* Copyright © 2001, 2002.

[10] **W3C Recommendation** *Document Object Model (DOM) Level 2 Core Specification Version 1*, November 13,200

[11] **Eric Armstrong, Jennifer Ball, Stephanie Bodoff, Debbie Bode Carson, Ian Evans, Dale Green, Kim Hasse & Eric Jendrock** *The J2EE 1.4 Tutorial for Sun Java Systems Application Server Platform Edition 8.1 2005Q2 UR2*, June 8, 2005.

[12] **Michael S. McGarr** *Taking XML to next level,* EC World, September 2000.

[13] **Elliotte Rusty** Adjunct Professor, Polytechnic University, *Managing XML data: Native XML databases*, 06 Jun 2005

[14] **W3C Recommendation** *Namespaces in XML 1.1,* 4 February 2004