COM on

Developing Component based Information Systems
with tools supporting the Microsoft Component Object Model

Egil P. Andersen
Norwegian Computing Center
P.O.Box 114, Blindern, 0314 Oslo, Norway
Tel: +47 22 85 25 94, Fax: +47 22 69 76 60
Egil.Paulin.Andersen@nr.no
Outline

Some tools and technologies for the development of Component based Information Systems

- Object Management Architecture, CORBA (Common Object Request Broker Architecture)
- Microsoft COM (Component Object Model), DCOM (Distributed COM)
- Component Object Models, IDL (Interface Description Language)
- Layered Architectures, 3-tier/n-tier, Business Objects
- Rational Rose/UML (Unified Modelling Language)
- Visual Basic, Visual C++, ATL (Active Template Library), J++ (MS Java)
- Compound Documents, ActiveX Controls, ActiveX Documents
- Universal Data Access, OLE DB, ADO (ActiveX Data Objects)
- MTS (Microsoft Transaction Server)
- IIS (Internet Information Server), ASP (Active Server Pages), Scripting
- SOAP - Simple Object Access Protocol
- XML, DTD (Document Type Definition), DOM (Document Object Model), XSL (eXtensible Stylesheet Language)
- Microsoft Repository, Visual Component Manager (VCM)
- Microsoft Message Queue Server (MSMQ)
- OODB (??)
- …..and more….puh….
OMG’s Object Management Architecture

- Syntactical interoperability
- Semantic interoperability

OMG - Object Management Group
CORBA - Common Object Request Broker Architecture
Local in-process, Local out-of-process, Remote

Client Process

- In-Process object
- In-Process server

Local Server Process

- Stub
- COM
- Local Object

Remote Machine

- Local Object Proxy
- COM

Remote Server Process

- Stub
- COM
- Remote Object
- Remote Server

Client Application

Local in-process, Local out-of-process, Remote
Marshalling for Out-of-Process Components
Basic COM (Component Object Model)

- **VTable interfaces** - a binary standard with interfaces based on a memory layout corresponding to that of abstract classes in C++

  A COM interface and its functions is similar to an abstract base class with a set of virtual functions in C++

  The extra level of indirection provides flexibility with respect to how interfaces are implemented.

- **Dispatch interfaces** - query the interface for its functions and their signatures

- **Dual interfaces** - available both for efficient vtable access and for scripting languages
**Interfaces, Components/CoClasses, Objects, GUID (Globally Unique Identifiers), CLSID, IID**

**Interfaces:** Versioning - Multiple interfaces - Single inheritance - IUnknown

- f1(...) -> ...
- f2(...) -> ...
- g1(...) -> ...
- g2(...) -> ...
- h1(...) -> ...
- h2(...) -> ...

**CoClasses**

- C
- D

**Objects**

- c1
- c2
- c3
- d1
- d2

**DLL or EXE**
Integrating COM Components via Containment vs Aggregation
IDL - Interface Definition Language

[ object,
  uuid(EA762187-A99A-11d3-95F4-0060979B4844),
  oleautomation,
  dual,
  …..]
interface IOSSSMLogin : IDispatch
{[id(1), helpstring("Function LogOn")]
  HRESULT LogOn([in] BSTR user, [in] BSTR pwd, [out] VARIANT_BOOL* okLogOn);
}

[id(2), helpstring("Function LogOff")]
  HRESULT LogOff([out] VARIANT_BOOL* okLogOff);
};

[ object,
  uuid(EA762188-A99A-11d3-95F4-0060979B4844),
  oleautomation,
  dual,
  …..]
interface IOSSSMXML : IDispatch
{[id(1), helpstring("Function GetRecordInfo")]
  HRESULT GetRecordInfo([in] long recordID, [in] short retrievalMode,
  [in] VARIANT_BOOL getHTML, [out] BSTR* XMLString);
  …..
};
Component Object Models

- In component based systems an object model consists of classes, interfaces, functions, etc, typically specified by an IDL (interface definition language).

- MS Word COM/Automation interfaces illustrated in the Visual Basic Object Browser
Programming Languages and Development Environment

- **Microsoft Visual Studio** - an elaborate development environment
- **Visual Basic** - very(!) easy to learn and use - inflexible - performance
- **Visual C++** - powerful and flexible - complex - wizzardmania….
- **Visual J++** - no experience with it…..
- **ATL (Active Template Library)** - utility for creating COM components in VC++
Compound Documents
with ActiveX Controls and ActiveX Documents

USIT - Universitetets senter for informasjonsteknologi

USIT informerer
IT-tjenester ved UiO
USITs elektroniske oppslagstavle
Avisa info@usit.uio.no

Private Sub cmdCheck_Click()
Dim myWord As Word.Application
Set myWord = New Word.Application
If myWord.CheckSpelling(txtSpell.Text) Then
    lblResult.Caption = "Correct spelling"
Else
    lblResult.Caption = "Incorrect spelling"
End If
End Sub

Private Sub cmdGoto_Click()
WebBrowser1.Navigate txtURL.Text
End Sub
Layered Architectures - 3-tier/n-tier

User layer

Application layer

Data layer

Computer1
GUI

Business Object

Computer2

Business Object

Computer3

Business Object

Computer4

Data Source

Computer5

Data Source
Characteristics of Rational Rose/UML

+ “Mainstream” - well-known and seen as a standard
+ Information modelling and explicit object interaction modelling
+ Object model available via COM/automation - it can be extended and customised
+ Code generation (but not production code…)
+ Informal (is this a plus??)

÷ Business rules and behaviour other than explicit object interaction
÷ Conceptual errors cannot be detected - models are not correct/incorrect - no modelling tool can distinguish good from bad models (and this is difficult also for experienced modellers)
÷ Incomplete
÷ Slightly confusing organization (at least at first…)

• Consider it mainly as a drawing tool and as a model repository
• Use only those parts that are well understood/agreed upon, and use it consistently - do not “over-model”
• Modelling syntax is not essential, but you are not likely to do e.g. Class Diagrams any better...
• Assuming that analysis/design is essential to large-scale software development, then a modelling tool can be useful to establish good routines for planning and documentation, and as a means for unambiguous communication internally and externally.
Rose (cont.) - a model is organized into a set of logical packages
Single-valued, multi-valued and composite attributes of items of particular categories. Single-valued attributes are defined as ordinary attributes. Multi-valued attributes are defined with a type defined as "Self\(<\text{category}\)>". Composite attributes, i.e., attributes with their own set of attributes, are defined by classes with the stereotype

Rose (cont.) - Class Diagrams - Information Models
Rose (cont.) - customised views on the model

The diagram is meant to be used for queries against the logical model, i.e., to present different views on the overall model. Below is a complete view of the User module components except Category inheritance relationships.
Rose (cont.) - COM/Automation - the Rational Rose Object Model

```
<table>
<thead>
<tr>
<th>Classes</th>
<th>Members of 'RoseApplication'</th>
</tr>
</thead>
<tbody>
<tr>
<td>RoseAction</td>
<td>Height</td>
</tr>
<tr>
<td>RoseActionCollection</td>
<td>Loft</td>
</tr>
<tr>
<td>RoseAddIn</td>
<td>PathMap</td>
</tr>
<tr>
<td>RoseAddInCollection</td>
<td>ProductName</td>
</tr>
<tr>
<td>RoseAddInManager</td>
<td>Top</td>
</tr>
<tr>
<td>RoseApplication</td>
<td>Version</td>
</tr>
<tr>
<td>RoseAssociation</td>
<td>Visible</td>
</tr>
<tr>
<td>RoseAssociationCollection</td>
<td>Width</td>
</tr>
<tr>
<td>RoseAttribute</td>
<td>CompileScriptFile</td>
</tr>
<tr>
<td>RoseAttributeCollection</td>
<td>ExecuteScript</td>
</tr>
<tr>
<td>RoseCategory</td>
<td>Exit</td>
</tr>
<tr>
<td>RoseCategoryCollection</td>
<td>GetLicenseApplication</td>
</tr>
<tr>
<td>RoseCategoryDependency</td>
<td>GetProfileString</td>
</tr>
<tr>
<td>RoseCategoryDependencyCollection</td>
<td>NewModel</td>
</tr>
<tr>
<td>RoseClass</td>
<td>NewScript</td>
</tr>
<tr>
<td>RoseClassCollection</td>
<td>OpenExternalDocument</td>
</tr>
<tr>
<td>RoseClassDependency</td>
<td>OpenModel</td>
</tr>
<tr>
<td>RoseClassDependencyCollection</td>
<td>OpenModelAsTemplate</td>
</tr>
<tr>
<td>RoseClassDiagram</td>
<td>OpenScript</td>
</tr>
<tr>
<td>RoseClassDiagramCollection</td>
<td>OpenURL</td>
</tr>
<tr>
<td>RoseClassRelation</td>
<td>Save</td>
</tr>
</tbody>
</table>
```

Function `OpenModel(fileName As String) As RoseModel` as Member of `RationalRose.RoseApplication`
Rose (cont.) - retrieving the classes defined in a particular model

Dim roseApplication As RationalRose.roseApplication
Dim roseModel As RationalRose.roseModel
Dim roseClassCollection As RationalRose.roseClassCollection
Dim roseClass As RationalRose.roseClass
Dim i As Integer

Set roseApplication = New RationalRose.roseApplication
Set roseModel = roseApplication.OpenModel("C:\siemens\objectmodelEPR\diepr-objectmodel.mdl")
Set roseClassCollection = roseModel.GetAllClasses

For i = 1 To roseClassCollection.Count
    Set roseClass = roseClassCollection.GetAt(i)
    Debug.Print "Class: " & roseClass.Name
Next

Call roseApplication.Exit
Rose (cont.) - Generating Visual Basic from a Model

```vbnet
'##ModelId=3831DF9D0096
Private mPrivilegeAccess As String

'##ModelId=3831DF8A1064
Private mPrivilegeAdmin As String

'##ModelId=3831DF9A4037A
Private mPrivilegeDocumentDesign As String

'##ModelId=3831DFC40CC8
Public Function CreateAlias() As IUser_Alias
End Function

'##ModelId=3831DFC50CFA
Public Function CreateDocumentAccess() As IUser_DocumentAccess
End Function

'##ModelId=3831DFC60122
Public Function CreateOtherPermission() As IUser_OtherPermission
End Function

'##ModelId=383291100209
Public Property Get PrivilegeDocumentDesign() As String
    PrivilegeDocumentDesign = mPrivilegeDocumentDesign
End Property

'##ModelId=3832911801A5
Public Property Get PrivilegeAdmin() As String
    PrivilegeAdmin = mPrivilegeAdmin
End Property
```
Universal Data Access (UDA) with OLE DB and ADO

Data Provider

- RDBMS
- Directory Services
- E-mail
- Other Data Stores

Data Consumer

- C++
- Visual Basic
- Script
- Java

ADO (ActiveX Data Objects)

OLE DB (Object Linking and Embedding Database)
SynEx - Synergy on the Extranet
Shared, Federated Electronic Healthcare Records
SynEx Demonstrator Architecture

1. SynEx Client
   - IE5 - Internet Explorer 5
   - record browser (client input)
   - XML formatted request
   - ActiveX, XML browser, XML parser, XSL processor

2. Simple Client
   - any web browser
   - XML/XSL response
   - DCOM response

3. DCOM Client
   - DCOM enabled client application

ASP - Active Server Pages
   - web server interface - session information (VBScript)
   - HTML
   - SynEx XSL
   - IIS - Internet Information Server

Session Manager (stateful, COM, dll)
   - web/XML services
   - (stateless, COM, dll)
   - (VB or VC++/ATL)

Server Component (stateless, COM, dll)
   - (VB or VC++/ATL)
   - MTS - Microsoft Transaction Server

SynXML generation
   - request for XML
   - DB interface (TSQL, Stored Procedures)
   - OSS - Oslo Synapses Server
     - SQL Server
     - SynXML generation

OSS
   - SynXML generation
   - record browser (client output)

ActiveX, XML browser, XML parser, XSL processor

DCOM Client
   - DCOM enabled client application
SOAP - Simple Object Access Protocol
XML formatted Server Requests via http

Server request:

http:\www.nr.no\synexdemo\oss.asp?<OSSrequest>
<RecordInfo RecordID="12082373463" Retrieval="all"/>
</OSSrequest>

The parameter ”<OSSrequest>…</OSSrequest>” is received by the Active Server Page ”oss.asp”, at the specified address, for further server-side processing.

Benefit:
Enable access to server-side business objects via http - less problems with Firewalls

PS: SOAP (Simple Object Access Protocol) uses a different XML format.
Active Server Pages (ASP)

- Avoid the use of scripting languages (e.g. VBscript) except as “glue” between COM components.
  VB, VC++ or J++ offers better development environments

<%@ Language=VBScript %>
<%
'--| The ASP script for retrieval of record and document XML from the Oslo Synapses Server.
'--|
'--| The script creates a COSSASPServer object and forwards its parameter client request to this
'--| object's PerformSynXMLRequest function. The COSSASPServer object will handle the request
'--| and forward the requested XML information, or an error message, back to the client via the
'--| Response object of this ASP script.

On Error Resume Next

Set objServer = Server.CreateObject("OSSSynExDemo.COSSASPServer")
objServer.PerformSynXMLRequest(Request.ServerVariables("QUERY_STRING"))

If Err.Number <> 0 Then
  Response.Write("…error message to client - e.g. XML formatted...")
  Err.Clear
End If
%>
Document Object Model (DOM) of the MS XML Parser

Function hasChildNodes() As Boolean
Member of MSXML.IXMLDOMNode

Members of IXMLDOMNode:
- nodeTypeString
- nodeValue
- ownerDocument
- parentNode
- parsed
- prefix
- previousSibling
- specified
- text
- xml
- appendChild
- cloneNode
- hasChildNodes
- insertBefore
- removeChild
- replaceChild
- selectedNodes
- selectSingleNode
- transformNode
- transformNodeToObject
MTS - Microsoft Transaction Server

- Distributed transactions
- DTC (Distributed Transaction Coordinator)
- Object pooling
- Database connection pooling
- The ”stateless” programming model
Microsoft Message Queue Server (MSMQ)

- Message - “piece” of information sent between two applications
- Messages can be formatted in e.g. XML
- MSMQ allow different applications to communicate with each other using “store-and-forward”
- MSMQ is similar to E-mail servers - more mechanisms for assuring the reception of messages sent
The DNV BRIX Architecture

- Domain model
- CIR - Relational database
- IPserver
- CIR generation
- CIS object cache
- Domain model concepts
- Domain model objects
- IPclient
- CIS generation
- Transaction management
- Concurrency control

Domain Model
Rational Rose/UML class diagram

Text file with domain model information

CIR - Common Information Repository

IPserver (MTS components)

Domain Model Information
Object State Information

SQL Server 7.0

GenCIR (generates DB schema and part of IPserver)

COM (possibly a slow network or temporarily not connected (offline))

Local (Client) Computer

COM Object cache

Central (Server) Computer

IPclient

DomainModel

Object State
Information

Domain Model Concepts

Concept Manager

Domain Model Concept

Domain Model Objects

Object Manager

Domain Model Object

Tool 1
View Vessel Information

Tool 2
Plan Survey

Tool n
Record Survey Observations

Application layer

Tool 1
Tool 2
Tool n

CIS - Common Information Services

Domain Model Concepts

CIS object cache

Domain Model Objects

IPserver (MTS components)
Motivation for BRIX

• Uniform access to a shared domain model

• Application developers only need to know the domain model to know how to operate on persistent objects instantiated from classes defined in the model

• Genericity is achieved without a generic domain model

• Services offered and transactions executed cannot be entirely predefined

• Change control - being able to handle changes in a flexible manner
  Different parts of the architecture provides relatively good encapsulation such that major changes or revisions do not affect the entire architecture

• Model independent CIS - easier to change the server centrally than to change all the distributed clients
Problems and Challenges for BRIX

- Performance
- Transaction handling and Concurrency control
- Caching strategies
- Enforcing general business rules
- Managing the software development process
- Roles and role modeling
  The generic caching services and the transaction and concurrency control mechanisms does not utilize knowledge on how the objects involved will be used and operated on by the application tools.
  Instead of creating a single overall domain model, create several smaller sub-domain models that each model a more specific and narrow domain; e.g. a single task or activity.
Microsoft Repository in the BRIX Architecture

Activity 1

Domain Model

UML Class Diagram

record

MS Repository

Domain Model Database

record

Activity 2

CIS

Meta Model

read

Concept Manager

COM interface

Activity 2

GenCIR

GenCIR Database

record

Activity 3

IPserver

generic part

domain dependent part

create

stored procedure interface

CIR Database

record

COM interface

stored procedure interface

CSD File

Manager

record

read

Creating a Microsoft Repository in the BRIX Architecture
Microsoft Repository

- Meta-information management
- Object Information Model
- Extendable Subject Areas
- COM/Automation access
Object Information Models of Microsoft Repository

RTIM - Repository Type Information Model
- A domain independent information model
- Made to record and retrieve meta-information on a variety of domains (e.g. UML, DB Schemas, Components, Datatypes, and more)
- Basic concepts: Class, Interface, Property (attribute and method), Collection, Relationship

IClass ==
    GetRoles() -> IAssociationRoleColl
    + [GetAssociations() -> IAssociationColl]
IAssociation ==
    GetRoles() -> IAssociationRoleColl
    + [GetClasses() -> IClassColl]
IAssociationRole ==
    GetAssociation() -> IAssociation
    GetClass() -> IClass

Repository UML Information Model
- A domain dependent information model
- Made to record and retrieve meta-information on UML models (e.g. from Rational Rose models)
- Accompanies COM/Automation interfaces
- Implemented by RTIM
- Currently “too normalized” - should allow for “redundancy”; e.g. Class-Association-Role relationships
Visual Component Manager (VCM)

A repository for organizing and storing information on components, models, projects, and more, to make them readily available to the development organization.
What about Object-Oriented Databases?

- **Main benefit**
  
  Avoids mismatch between relational data and object-oriented applications; e.g. inheritance relationships, recursive structures, ….

- **Do they scale well, do they perform well - may be - but many in industry considers it an added risk to rely on this for large enterprise information systems**

- **Developers will be happy with them - but will this reduce development cost enough to outweight the “risk” (real or perceived)?**

- **What about customers - do they benefit from it?**

- **Main problem**
  
  No “killer application” - there seems to be no undisputable need for it
Object-Oriented to Relational

a) Single table

b) Leaf tables only

c) One partial table per class

d) One full table per class

e) Logical split in the inheritance hierarchy
“Componentifying” FS?