A Large NetRexx Application

René Vincent Jansen, I-Bizz IT Services and Consultancy BV

Böblingen, May 6th 2004, 08.15 h.

Legal & Disclaimer

I-BIZZ IT Services
& Consultancy BV

- Every effort has been undertaken to ascertain this presentation does not contain sensitive material. The data of this application is classified IBM & ABN AMRO Internal and Confidential. The displays of the application that contain this data have been modified for presentation and do not reflect confidential data. Where actual components of the IBM Banking Data Warehouse Model have been shown, care has been taken that only information is been shown that is also publicly available through other sources, in casu the worldwide web or official IBM sales brochures.

- The previous notwithstanding, ABN AMRO Bank NV cannot be held accountable in any way, as the work has been carried out as an independent external party for the sole responsibility of I-BIZZ IT Services and Consultancy BV, Amsterdam, The Netherlands, and its intermediates in this assignment.

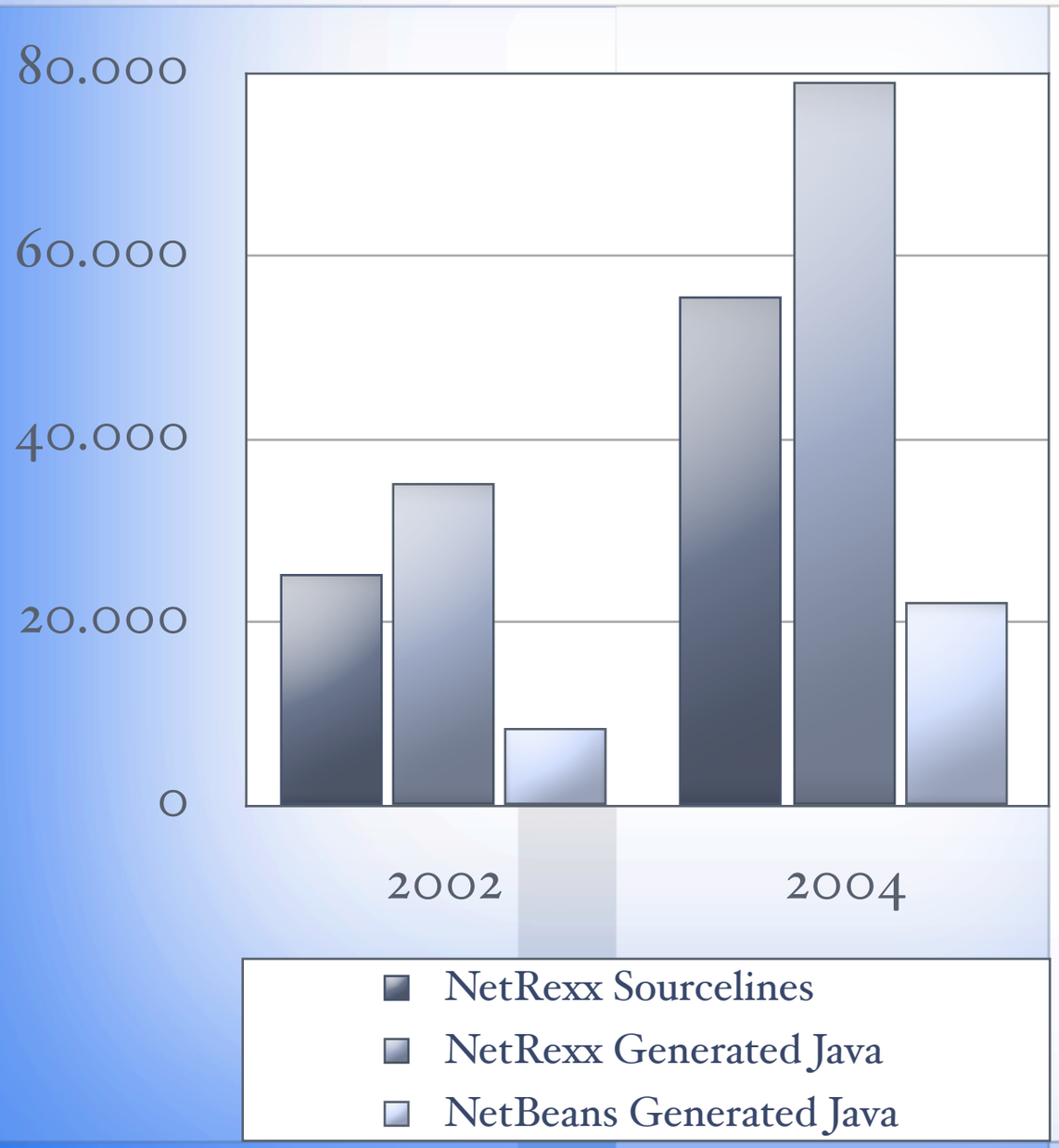
Goals

- Give an account of the design and implementation of a fairly large Object Oriented, Multi-User Client Server Application
- Show specific advantages
- Present practical experiences in combining NetRexx with generated Gui-builder code
- Show pitfalls and solutions

Non-Goal

- Preach to the converted

How large is Large?



- 52 Dialogs
- 232.427 Records in data base
- 18.834 Object Instances
- 376 NetRexx Classes
- 53 Java - Only Classes

About ABN AMRO

ABN·AMRO

Alfred Berg
ABN AMRO

BANCO REAL
ABN AMRO

Delbrück
ABN AMRO

Hoare Govett
ABN AMRO

LaSalle Bank
ABN AMRO

Standard Federal Bank
ABN AMRO

ABN·AMRO

- ABN AMRO is a prominent international bank, its origins going back to 1824. ABN AMRO ranks 11th in Europe and 23rd in the world based on tier 1 capital, with over 3,000 branches in more than 60 countries, a staff of over 110,000 full-time equivalents and total assets of EUR 560.4 billion (as of 31 December 2003).

About I-Bizz

IT Services & Consultancy

- I-Bizz IT Services and Consultancy is a privately owned limited company that operates since 1998 and is legally based in Amsterdam, The Netherlands. It has one Full Time Employee. It offers a portfolio of services that ranges from architectural consultancy to technical project management. Its current specialization is in Knowledge Management and Management Information Systems.

Requirements

Business Issue

Deliverables

Package Selection

The Business Issue

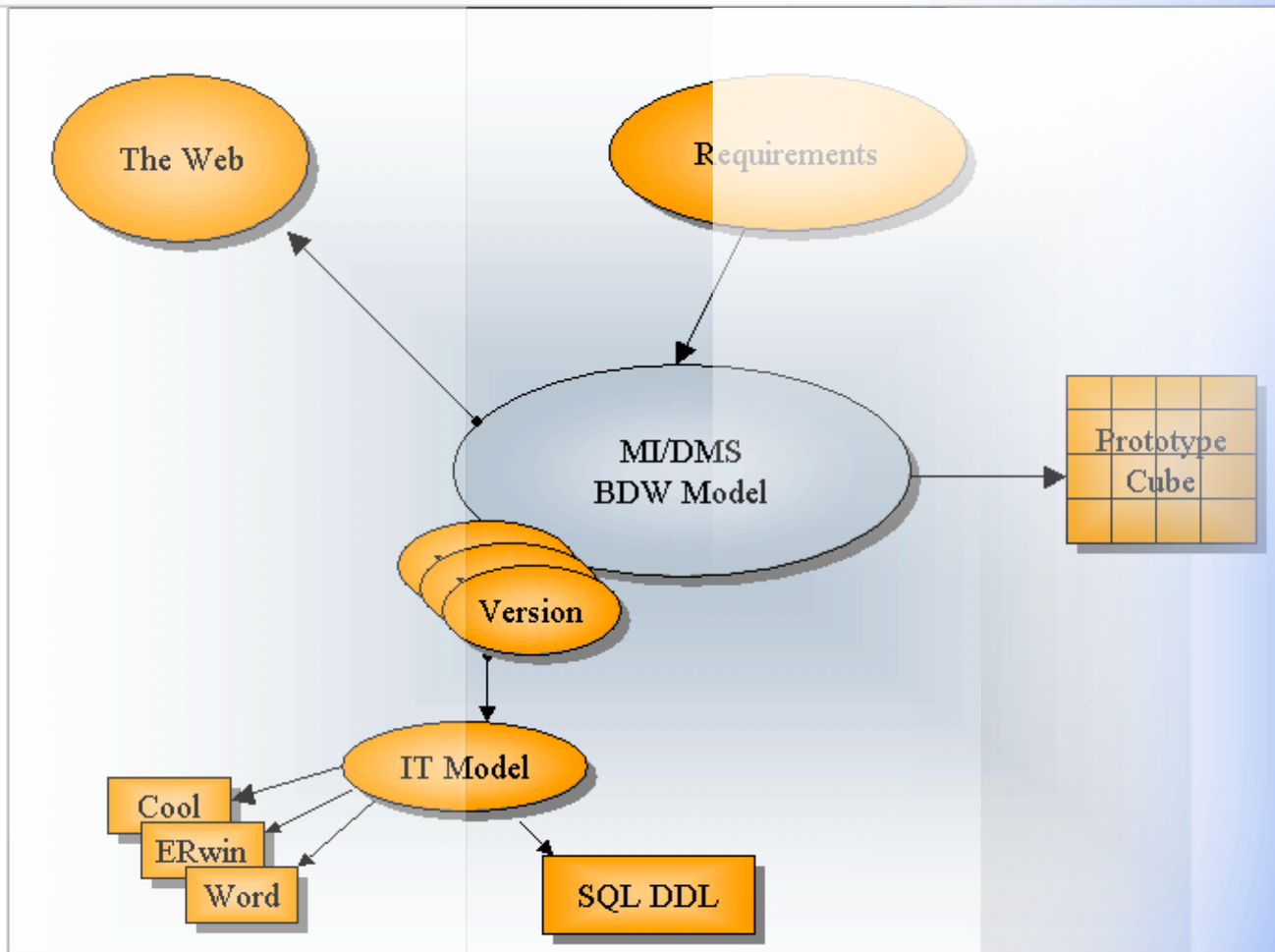
- ABN AMRO CFO decision too much was spent decentrally designing MIS Applications - "Data Warehouse Models."
- A centralized, Head Office department was tasked with developing and scoping an Enterprise Wide MIS Model
- The IBM BDWM was taken as a basis for the AAB MIS Model

Customers

- Our customers are ABN AMRO Business Units that need a data model for MIS Purposes
- We only do word-of-mouth advertising
- We are now 17 people strong but had to turn down some customers
- We only charge late implementations

Deliverables

- The main objective of a logical data model is to serve as a IT / business communication bridge
- It is delivered in various forms, as a structured MS Word Document and in forms suitable for different data modelling tools like Cool:biz, and ER*win*
- A Prototype *Cube* proves validity of the model



Logical Data Model

The logical model is always based on a requirements model. In this, e.g. the need for keeping history on an attribute/relationship level is specified.

Software Selection (August 2000)

- No package fit all requirements
- These are, amongst others
  - Multi User
  - Multi Project
  - Platform Independence
  - Integrated Versioning
- **Decision to build; team of 2**

Design

Meta Model

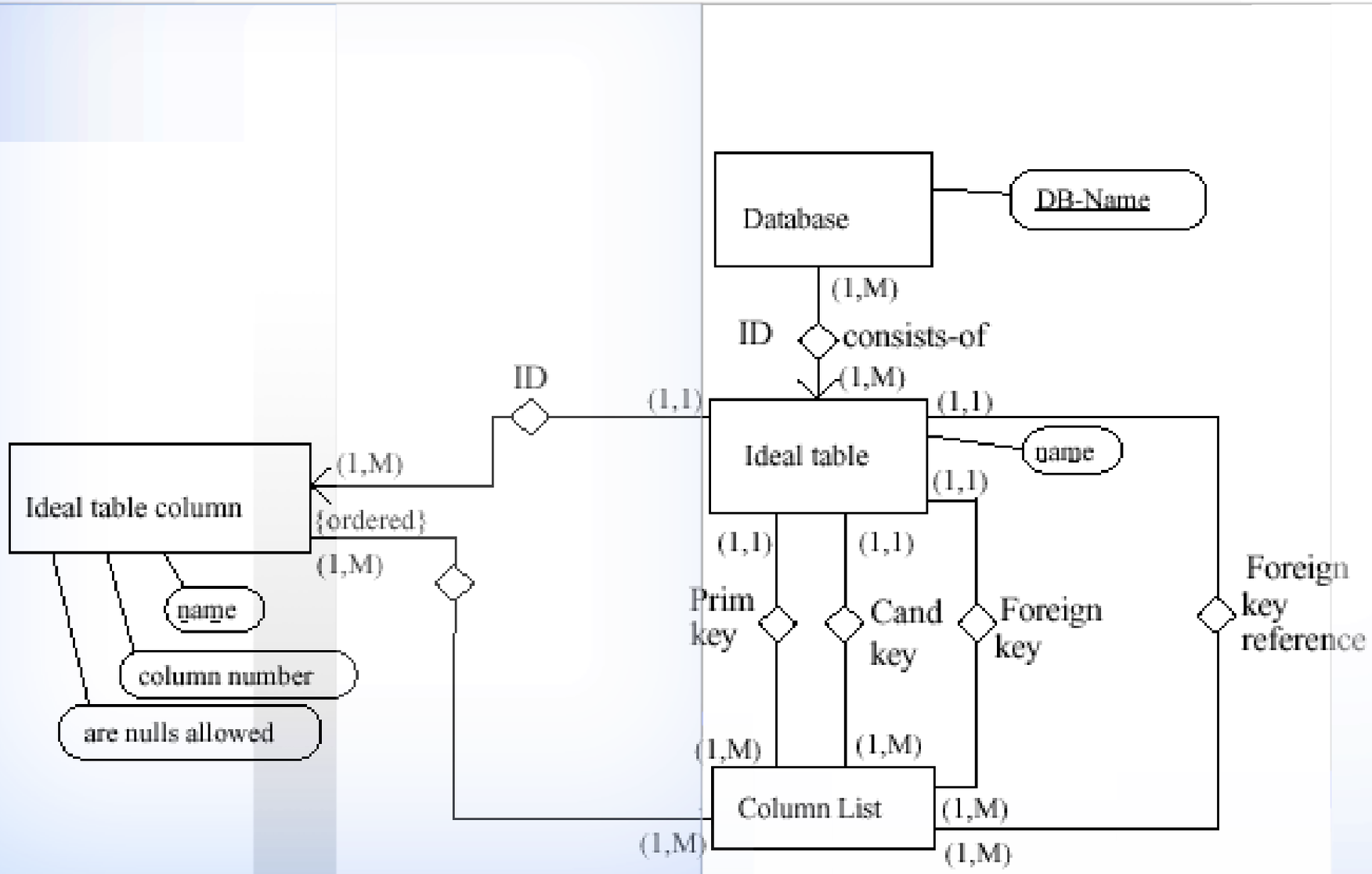
UML

Versioning & History

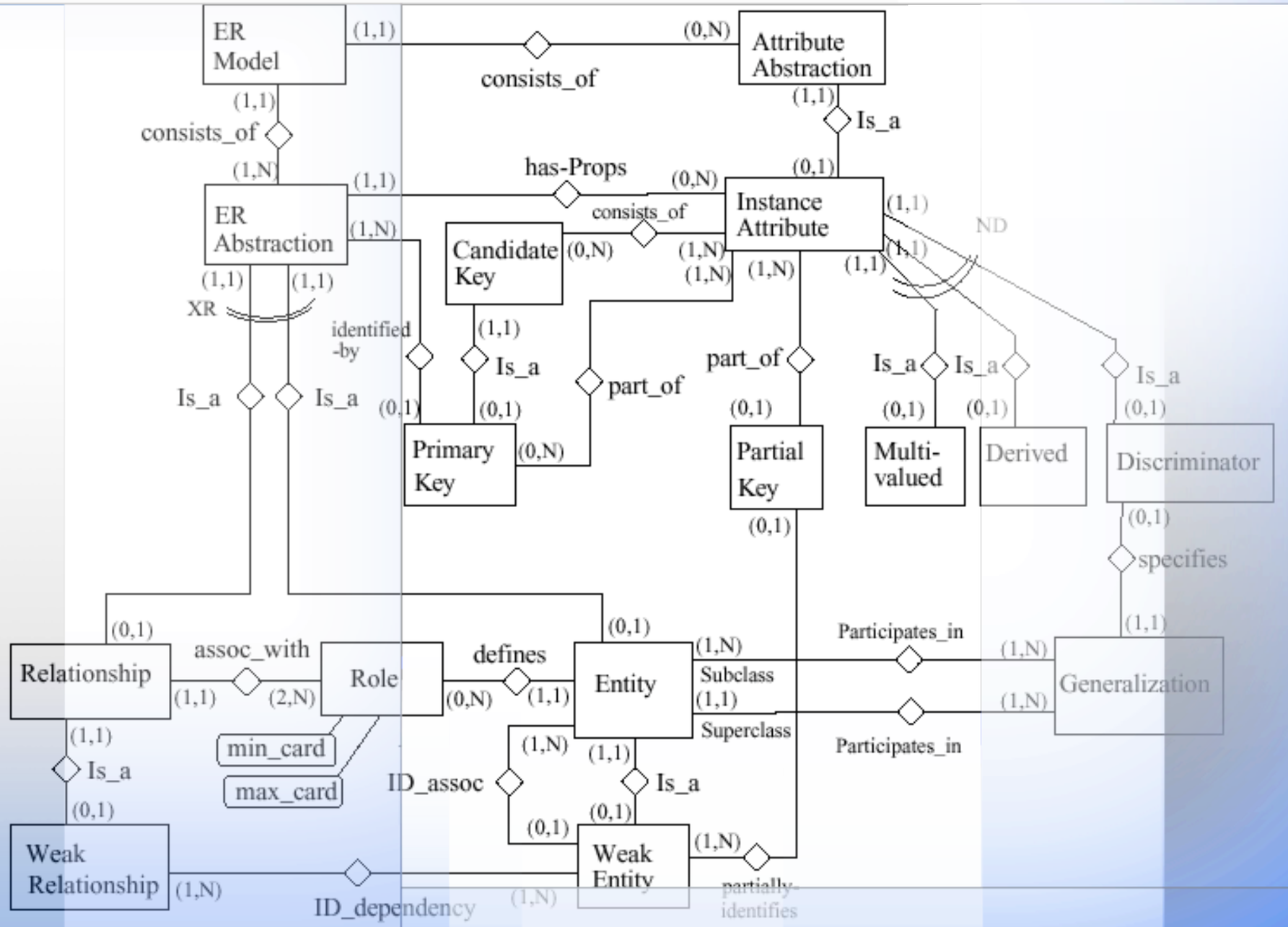
Relational & Object Tech

Persistance Engine

Relational Meta Model



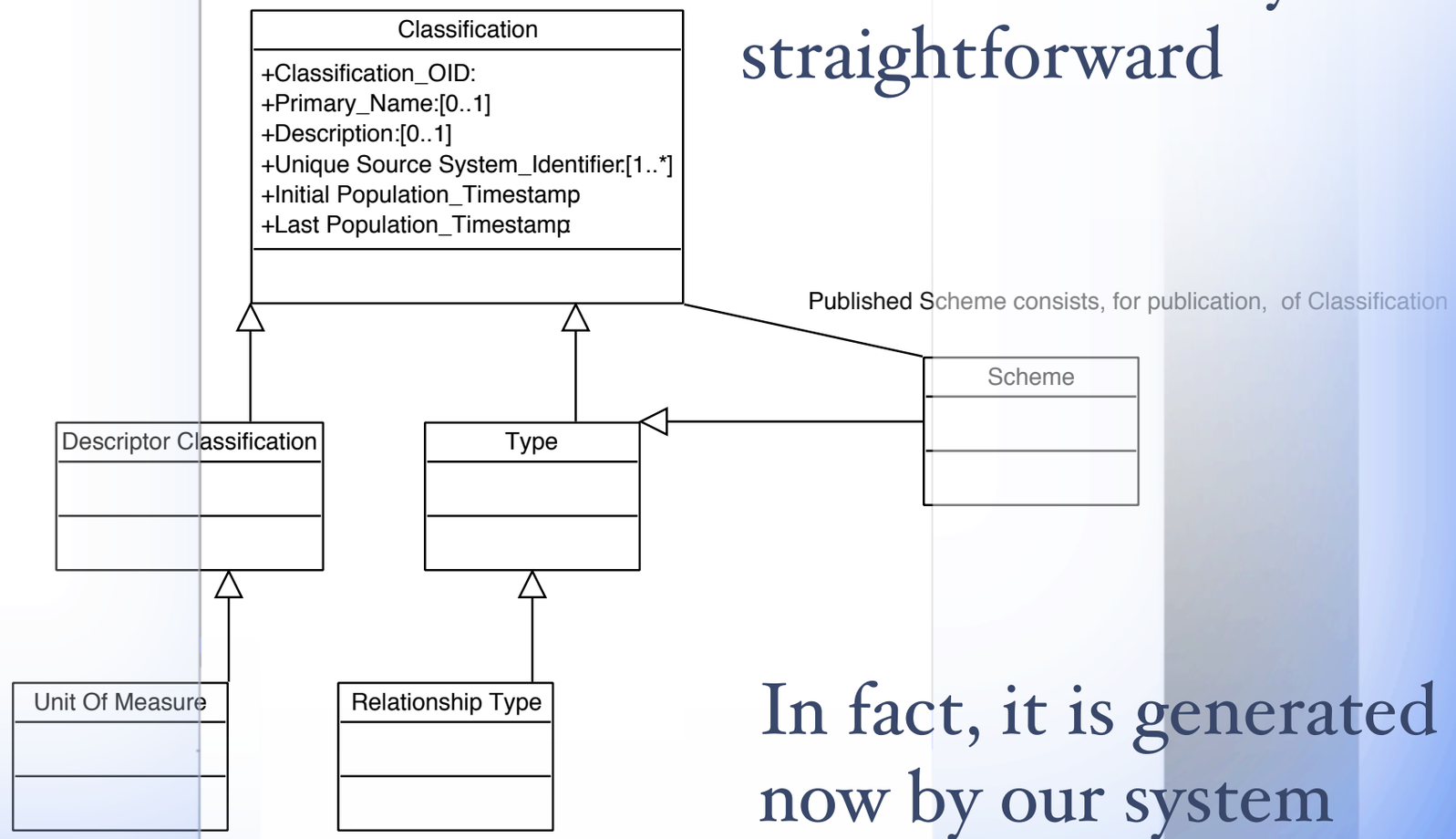
Extended ER Meta Model



Design using UML

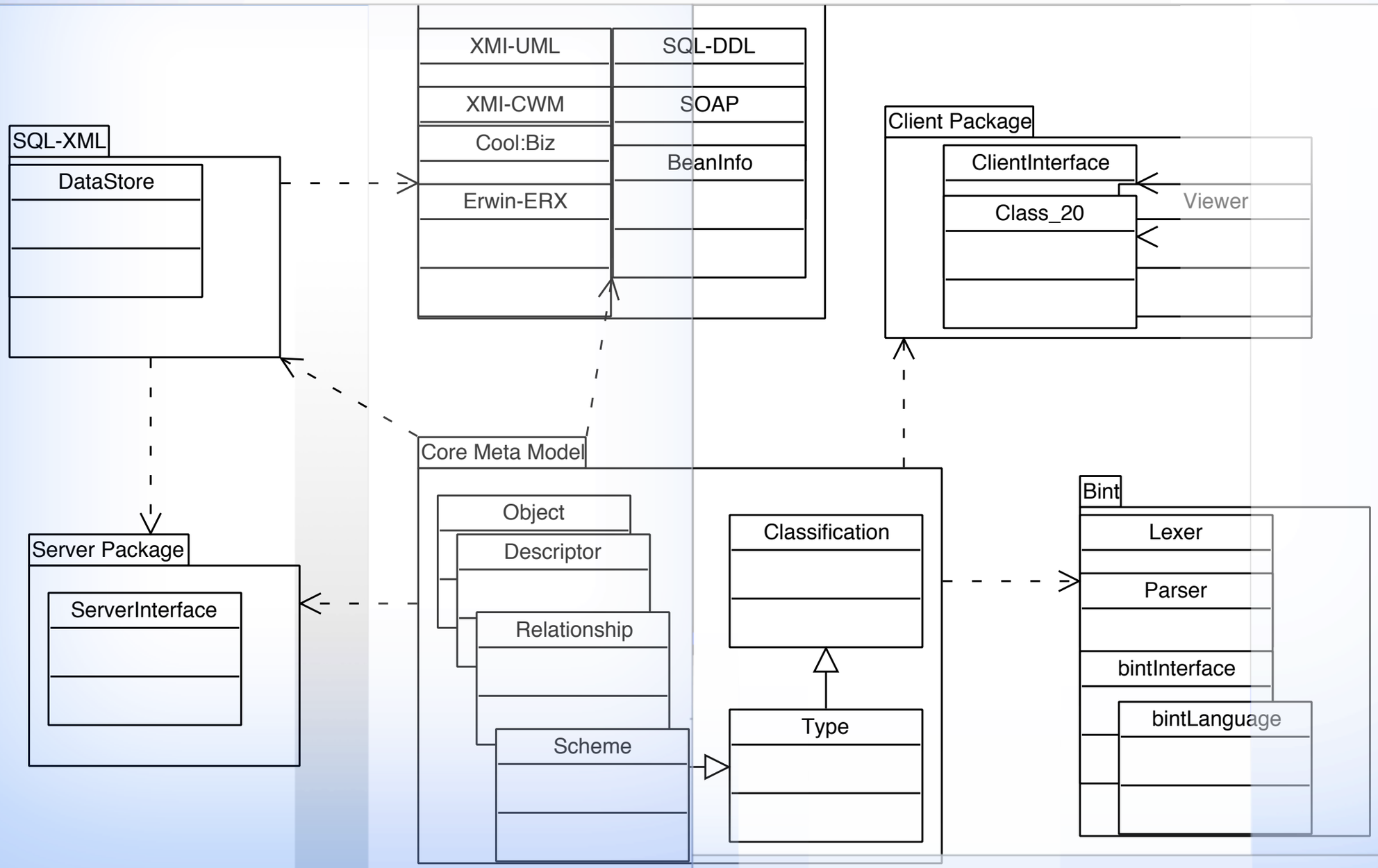
Works for Code and Database Design

From UML to NetRexx is totally straightforward



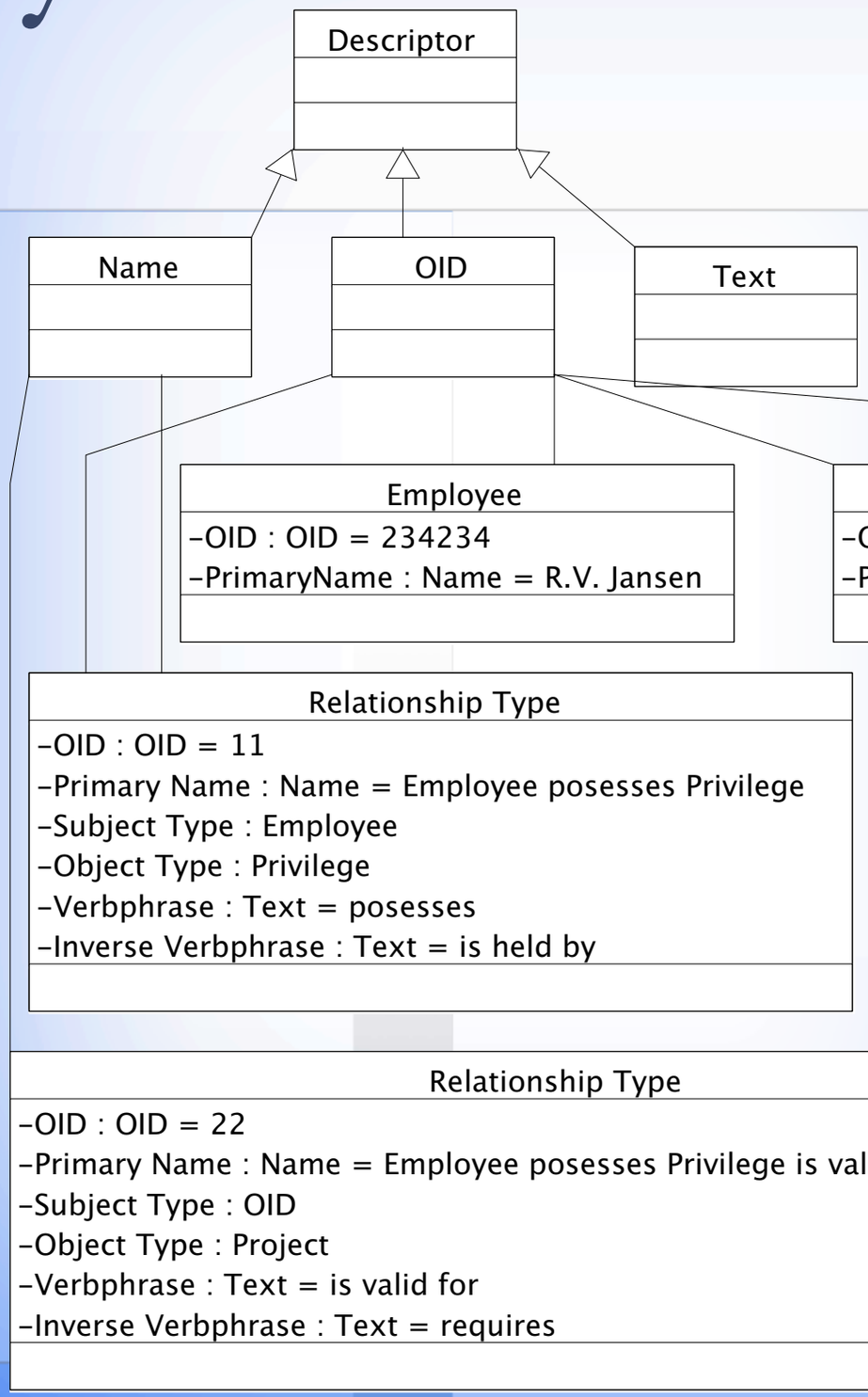
In fact, it is generated now by our system

Package Structure



Dyadic Relationship Modeling

(We say *binary*)



These relationship instances ...

… end up as the following propositions

O20013284 = {T = DomainOnly,N = Employee} (1)
O10009986 = {T = DomainOnly,N = Project} (2)
O22222222 = {T = DomainOnly,N = Privilege} (3)
O234234 = {T20013284,N = R.V.Jansen} (4)
O456456 = {T22222222,N = Update} (5)
O567567 = {T10009986,N = MetadataV 3} (6)
Rt11 = {ST20013284,OT22222222} (7)
Rt22 = {ST11,OT10009986} (8)
Rt33 = {ST20013284,OT22222222, IOT10009986} (9)

This is the setup needed to assert the following facts:
R999 = {Rt11, s234234, 0456456} (10)
R1000 = {Rt22, s999, 0567567} (11)

Scope Specific Properties

2637	Project *explicitly involves* Classification
10034282	Project explicitly includes Classification *specifies value for* Project Specifiable Relationship Type
10034283	Object *is supplied value for* Project Specified Relationship Type
10050690	Descriptor *is supplied value for* Project Specified Relationship Type (see above)

Physical Table Design

OID	OBJ_TP	PPN_DT	PPN_USR
1166	1	2000-09-01	DI2359

SBJ_OID	VERB_OID	OBJ_OID	RANK	OID	EFF_DT	END_DT
1166	2763	4567	9999	2009765	2000-01-01	9999-12-31

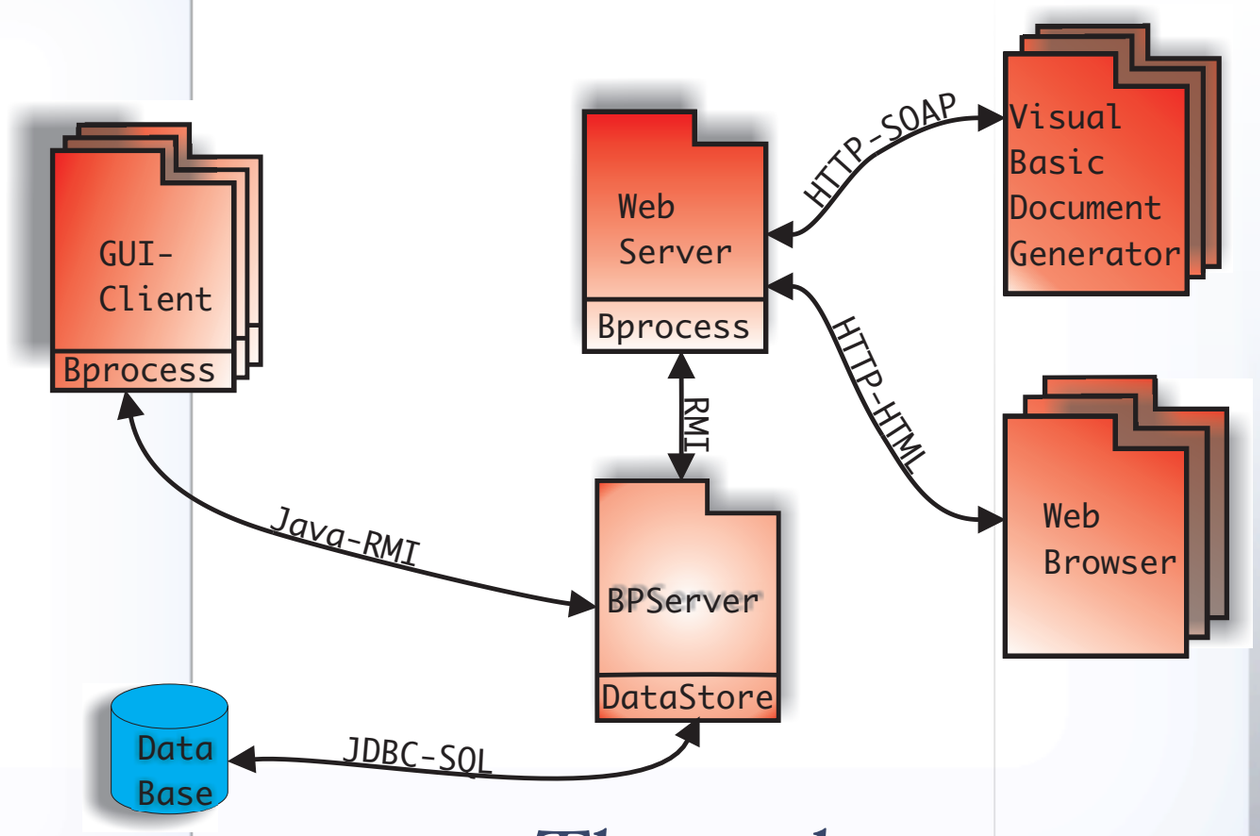
dsctr_x_obj_oid	obj_oid	nm
2710	1166	Involved Party

(Only Three Tables)

Components

Communication to VB & MS Office through SOAP (as a webservice)

Three Tier Architecture with Application Server



The package structure conforms more or less to these components

Versioning Mechanism

- Relationship instances have effective date and end date
- Every version of Every relationship is saved, by only setting its end date and never physically deleting it
- Every historic deliverable is reproducible
- A project is a subtype of Event; one timestamp suffices to save an entire project version.

Graphical User Interface

Swing

Look & Feel

Netbeans

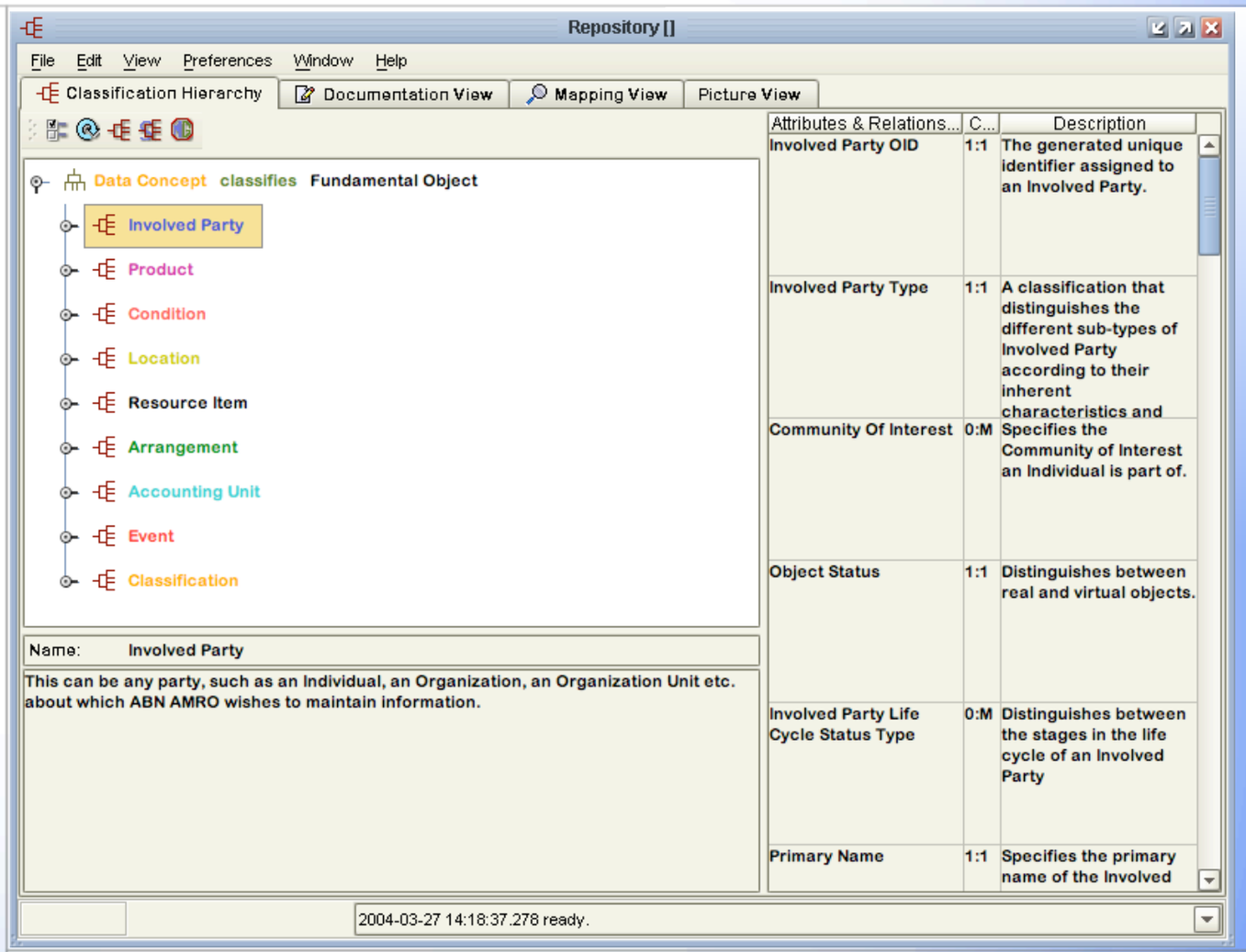
The Tree

Swing - Some Misconceptions

- Swing is ugly and slow
- Swing is hard to understand if you're not into Smalltalk Model - View - Controller
- Swing does not look like Windows at all
- Swing can't use threads

Actually ...

- You have to care for its appearance
- It is not slow at all - slowness is mostly your own doing
- You have to work on application startup to get it faster - multithread and load through introspection

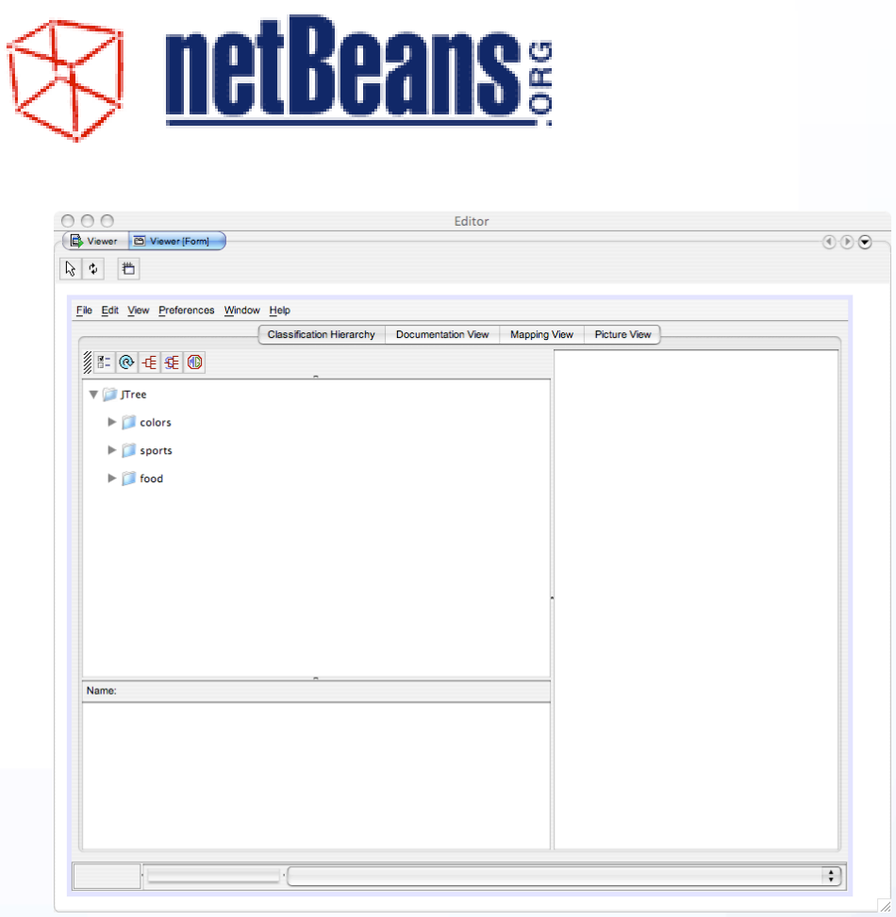


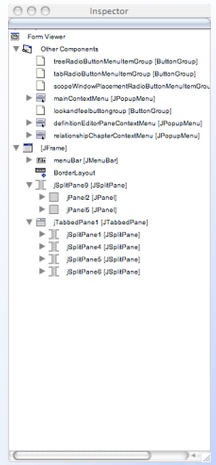
We use the Alloy Look and Feel

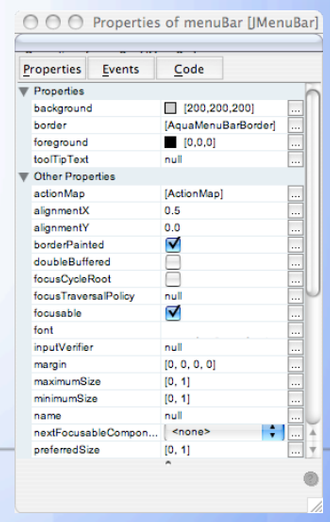
- For Cross-Platform consistency
- Frames and Widgets look better than native Swing (ok, not on Macintosh)
- JTree behaviour is more consistent - it seems to be influenced by the look and feel component

All of GUI made with NetBeans

- Paint the screens
- Add the widgets
- Double click
- Add call to NetRexx code
- NetBeans is pure Java
  - Works on all platforms



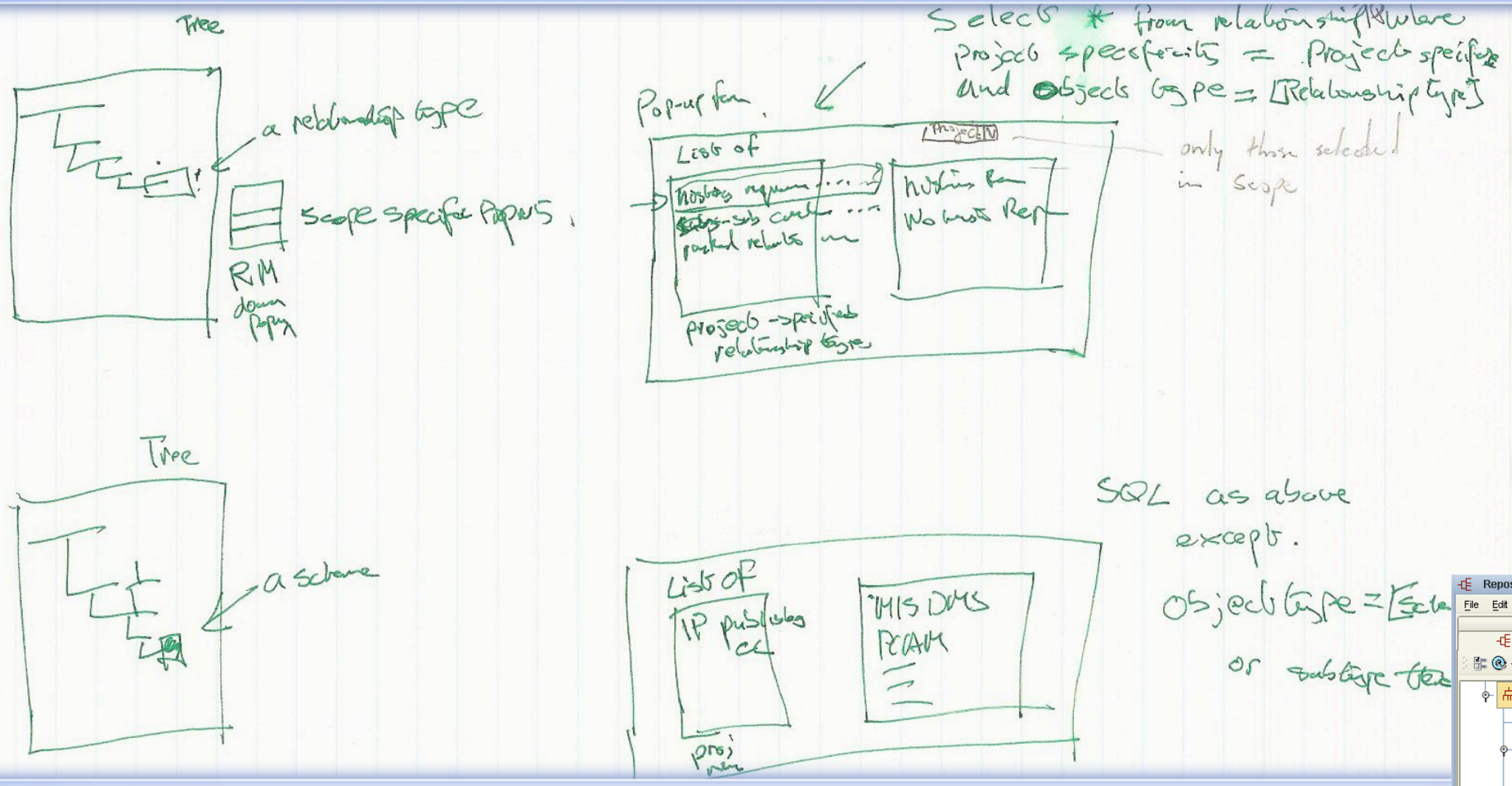


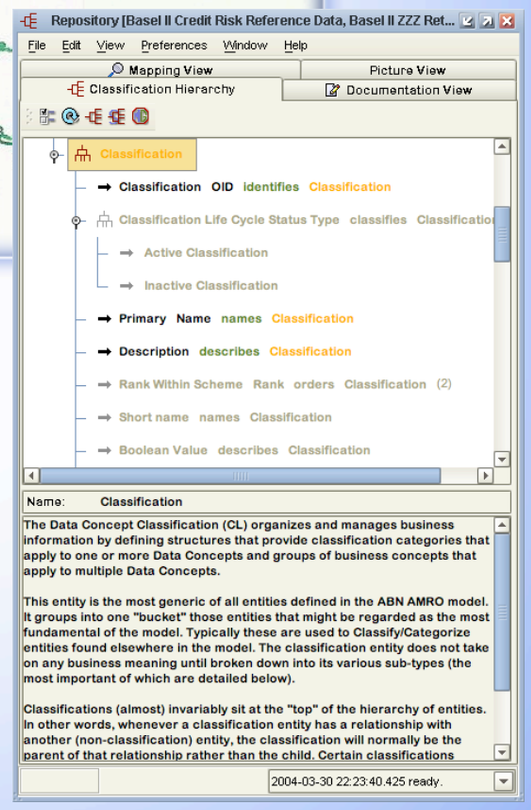


Adding calls to NetRexx methods

```
    private void exitMenuItemActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FI
med
        try {
            this.bp.BPDisconnect(System.getProperty("user.name"));
        } catch (Exception s) {
            JOptionPane.showMessageDialog(this, s.getMessage());
        }
        System.exit(0);
    }//GEN-LAST:event_exitMenuItemActionPerformed
```

Catch an event from a GUI Widget and call a method in a NetRexx Class



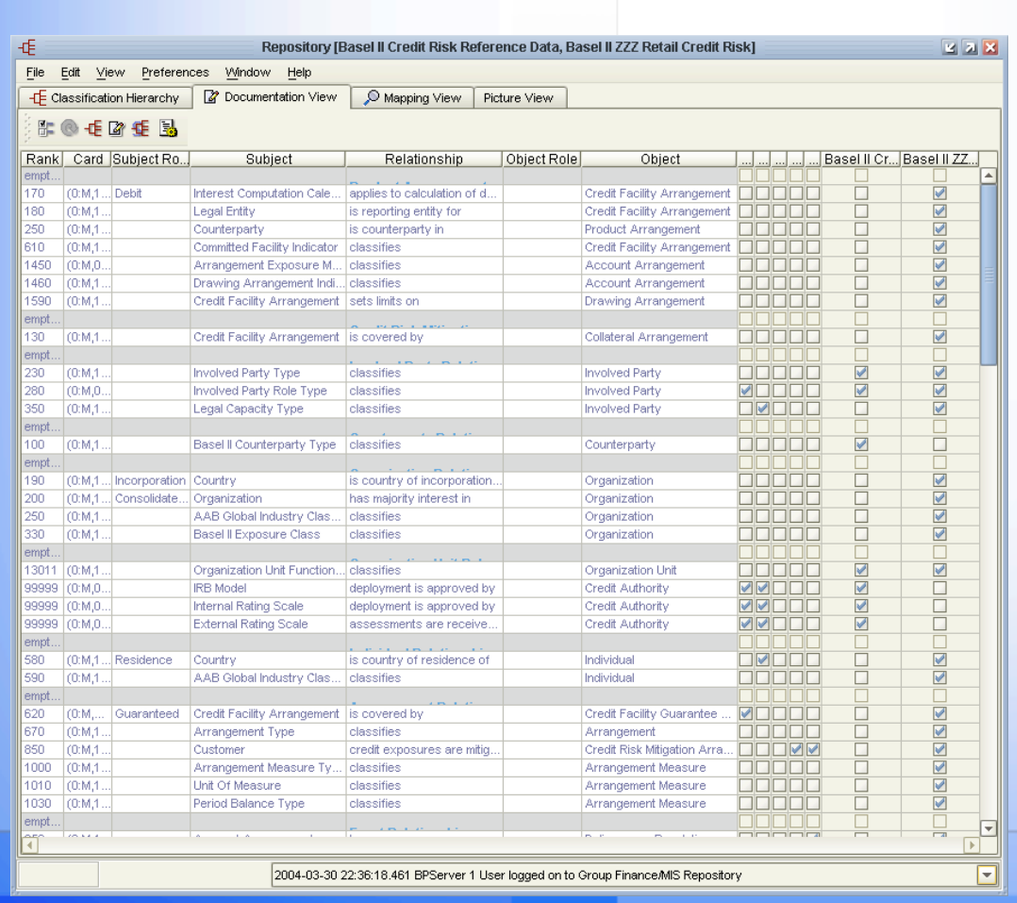


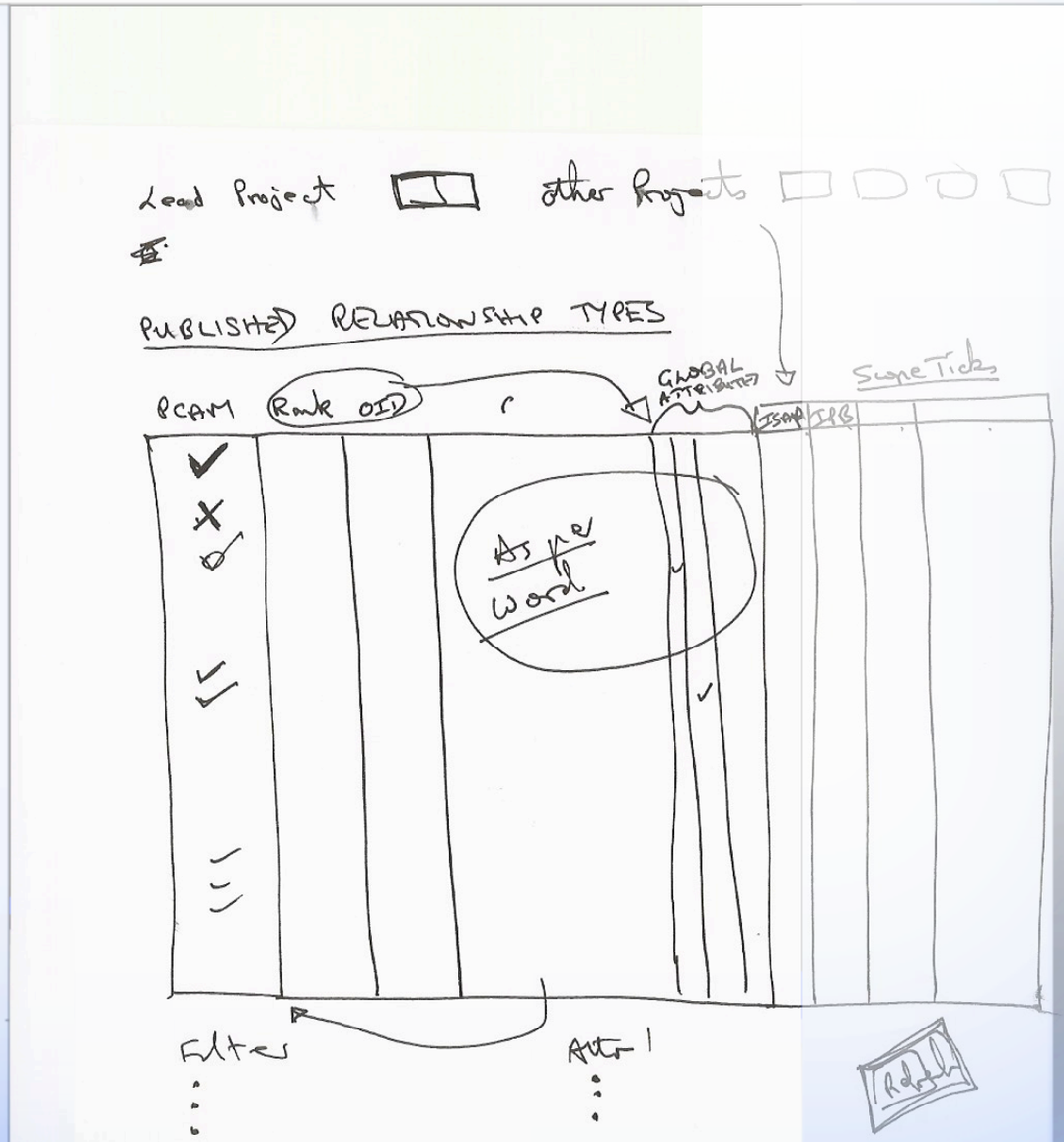
The Famous Tree GUI Design

by Kieran McKeown

Relationship Types Screen Spec

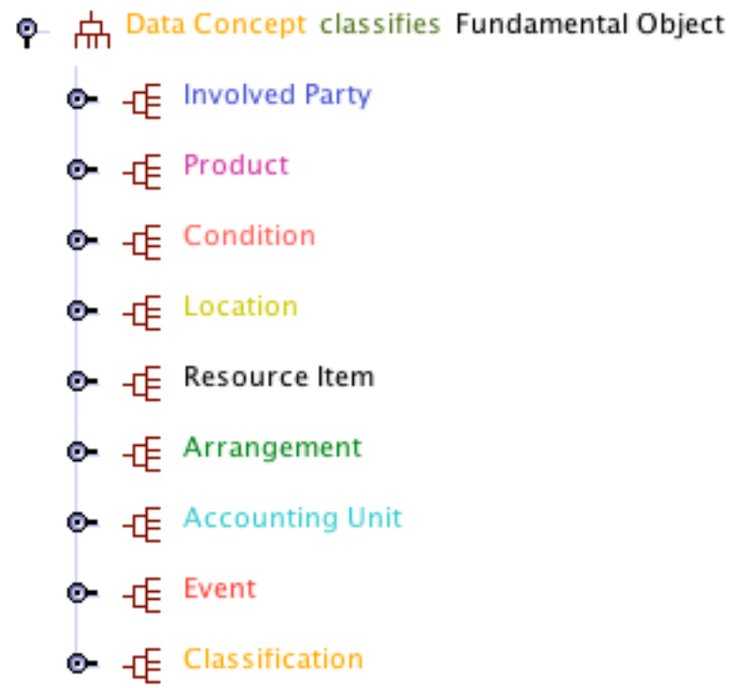
The famous relationship types screen design by Mike George





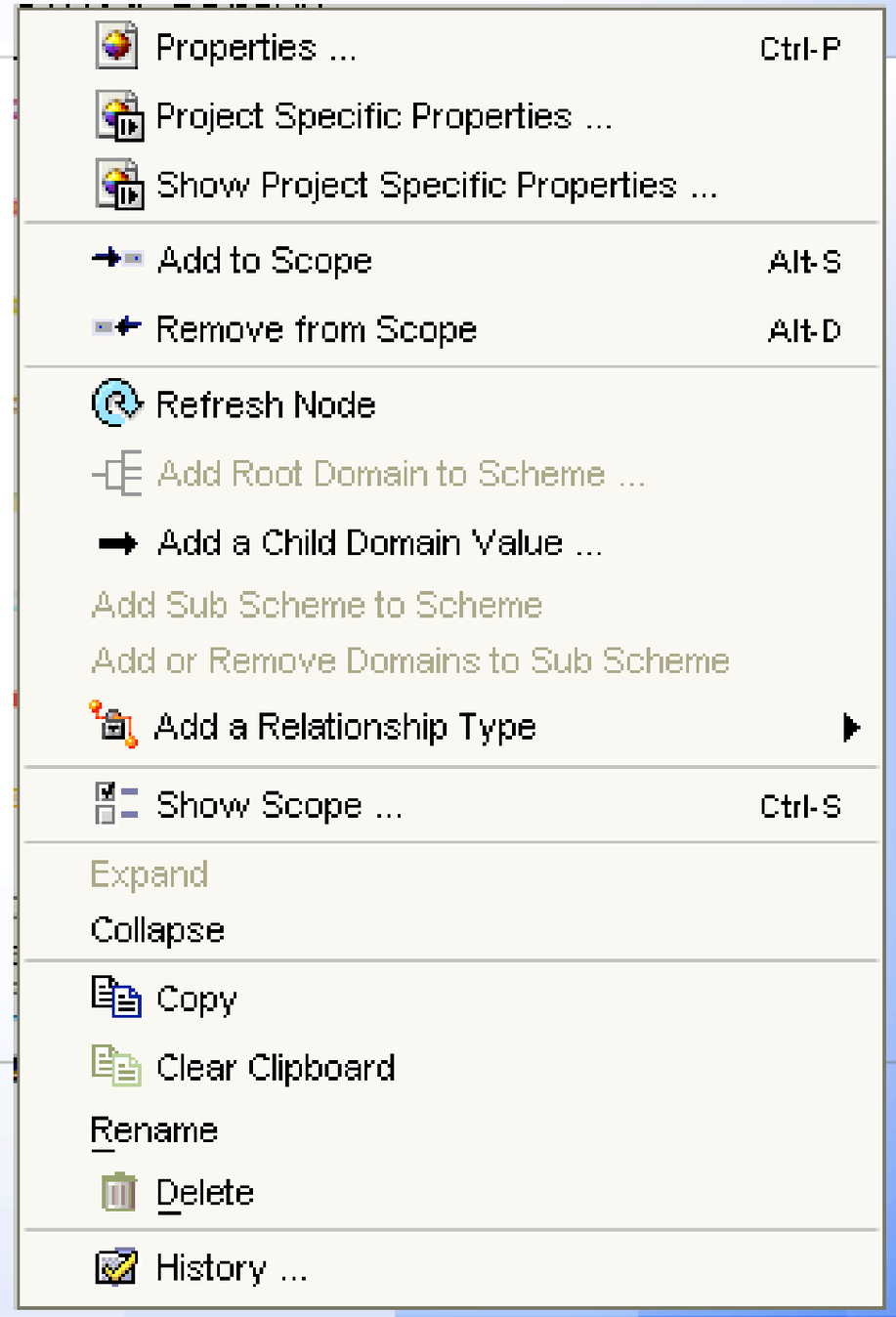
The Tree

- Is a JTree
- Action is in the tree Model
- Our most performance critical component
- Delivers to the user a scoped view to a shared object model

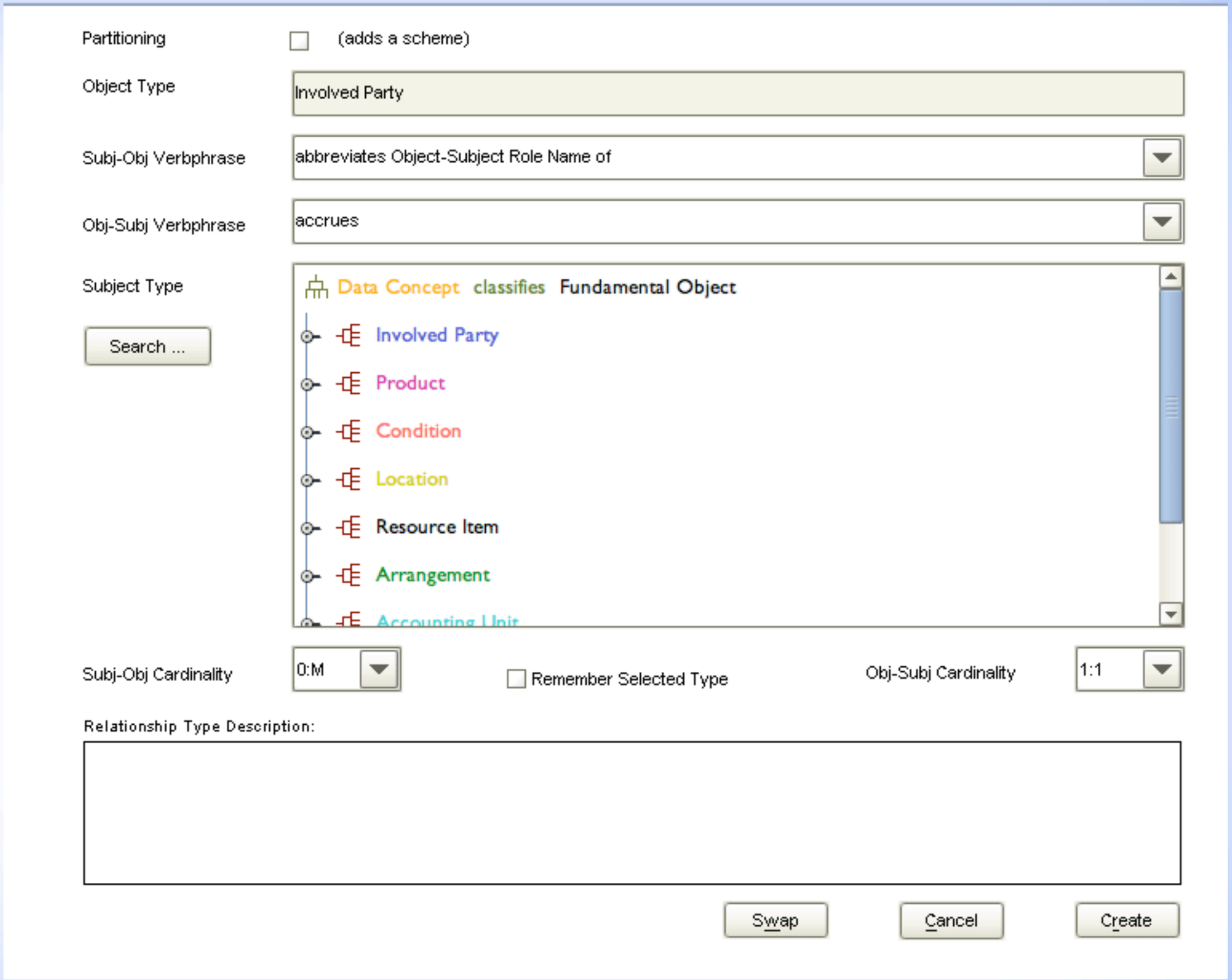


Right Clicking the Tree

- On every object in the repository
- Does scoping of the Common Model and additions and deletions
- Add Relationship Types



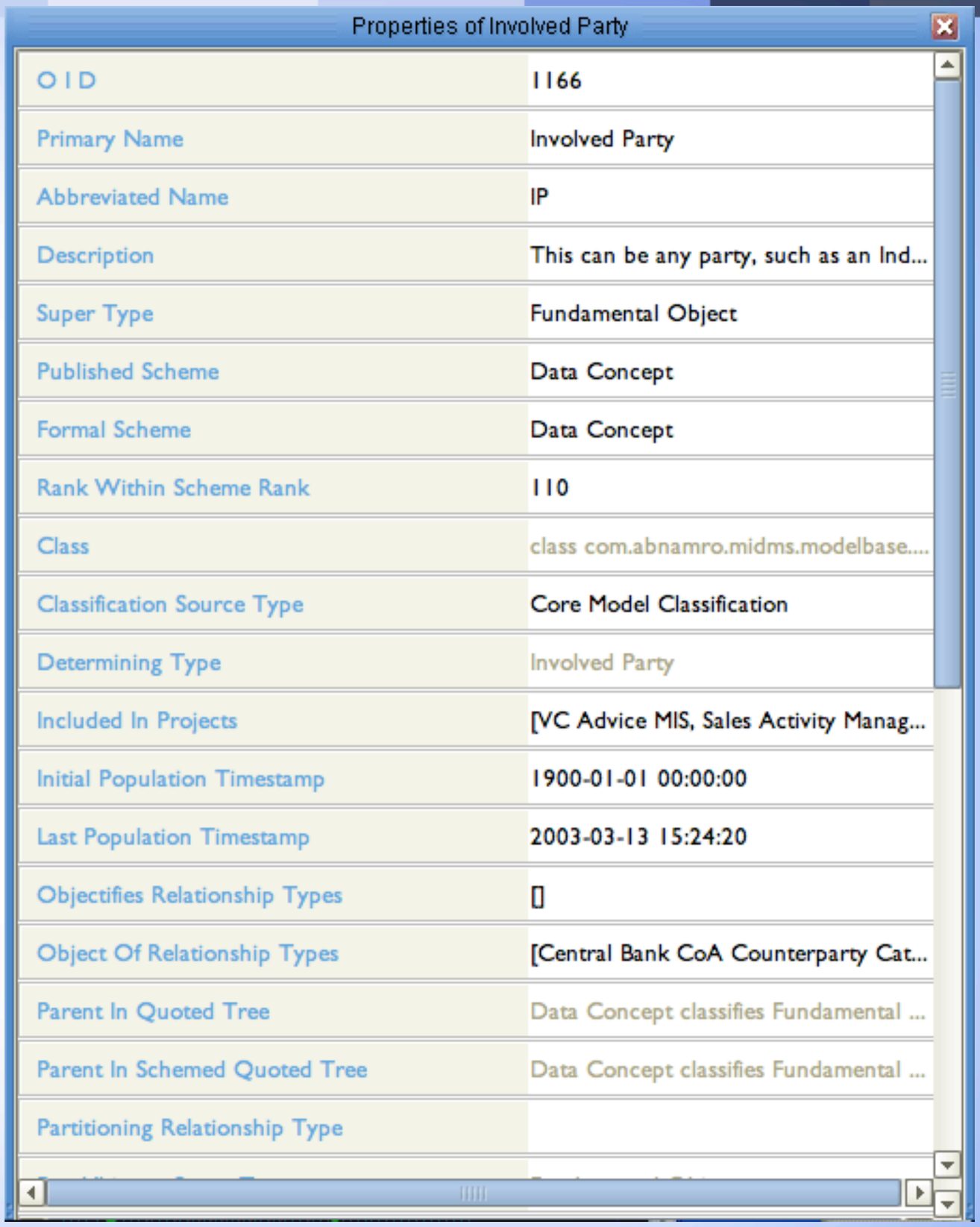
Adding a Relationship Type



Generic Object Editor

Edits any object thrown at it

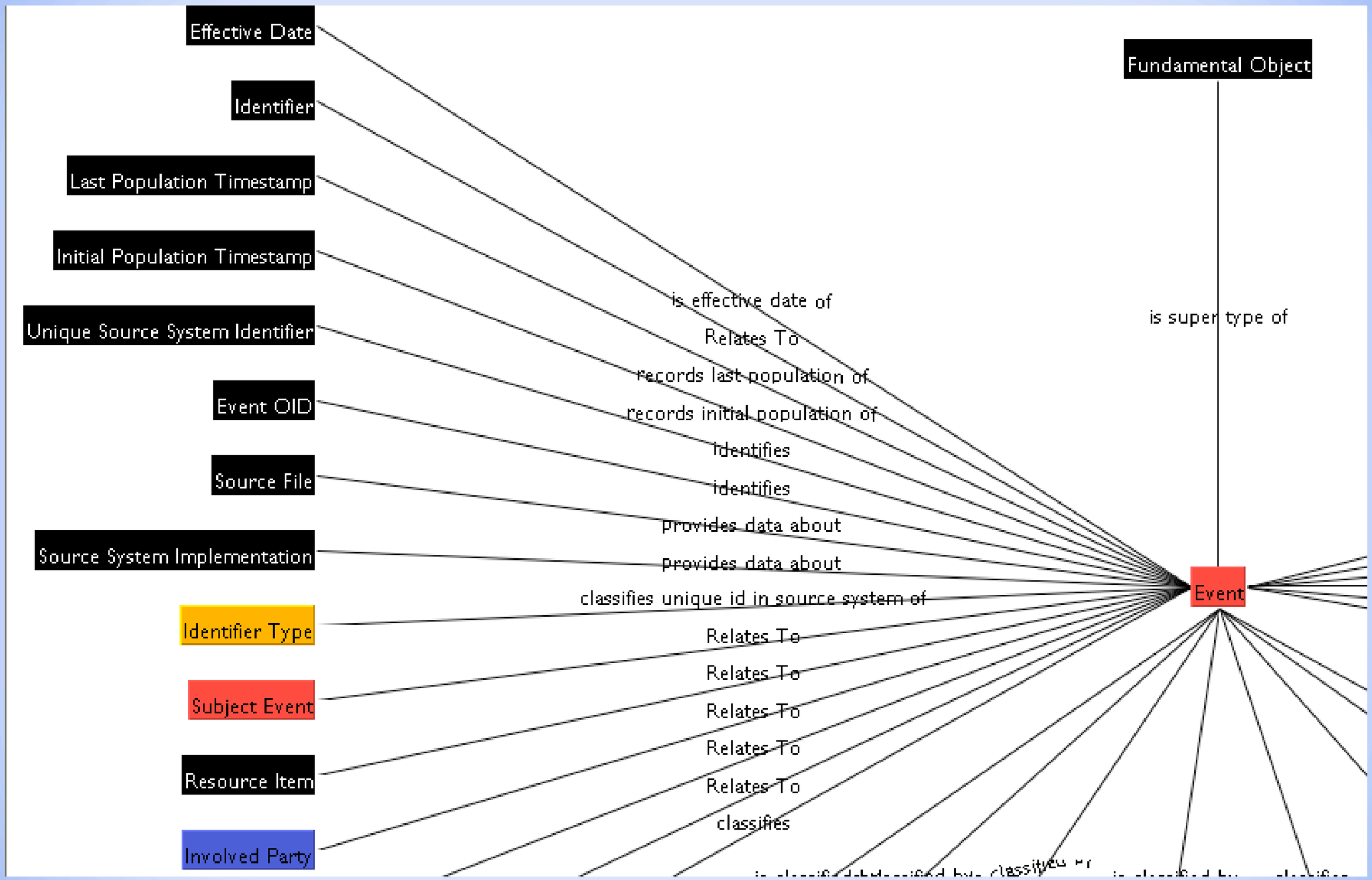
Uses beanpatterns and introspection to determine type of editor



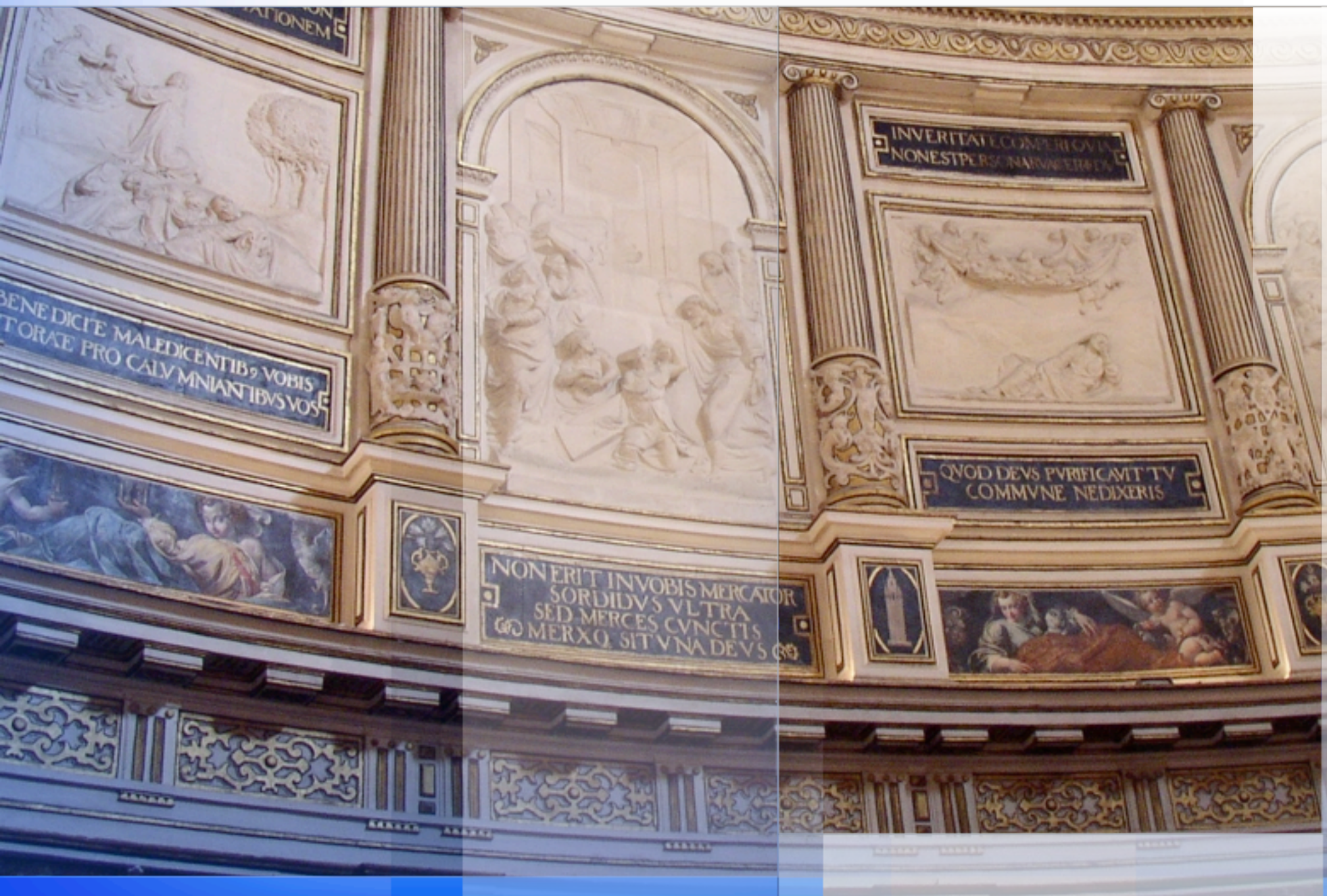
The Editor is a Visitor

```
do
  /* check whether the getter returns an object instance. if it
   * does, we pass it an EditorVisitor instance that handles the
   * editing this of course polymorphically with double dispatch
   * on the indirect object.
   */
  invokeResult = this.globalGetter.getMethod().invoke(this.globalObject, null)
  /* if the result from the Getter invocation is null, we
   * instantiate a new object and also have it accept an
   * editorvisitor.
   */
if invokeresult = null then
  do
    do
      cls = Class.forName(this.globalGetter.getMethod().getReturnType().getName())
      clz = cls.newInstance()
      (Visited clz).accept(edV)
      edV.getEditor.setFont(this.dialogFont)
      ppp.validate()
      this.panel.validate()
    catch Exception
      say "Exception instantiating object-to-be-edited"
    end
  end
else
  do
    (Visited invokeResult).accept(edV)
    edV.getEditor.setFont(this.dialogFont)
    ppp.validate()
    this.panel.validate()
```

Graphical Navigator Screen



The Server



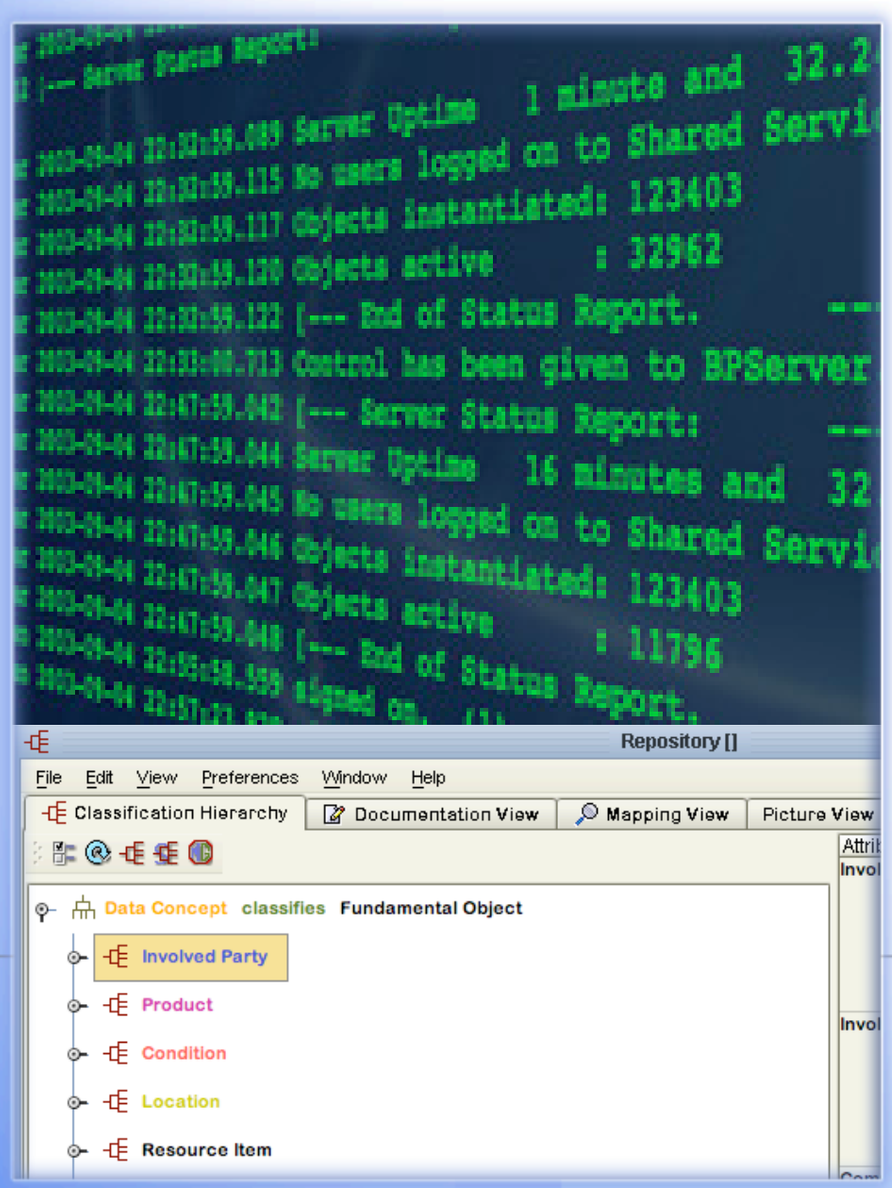
Why?

RMI

Caching Objects

Deconstructing Objects

Why an Application Server?



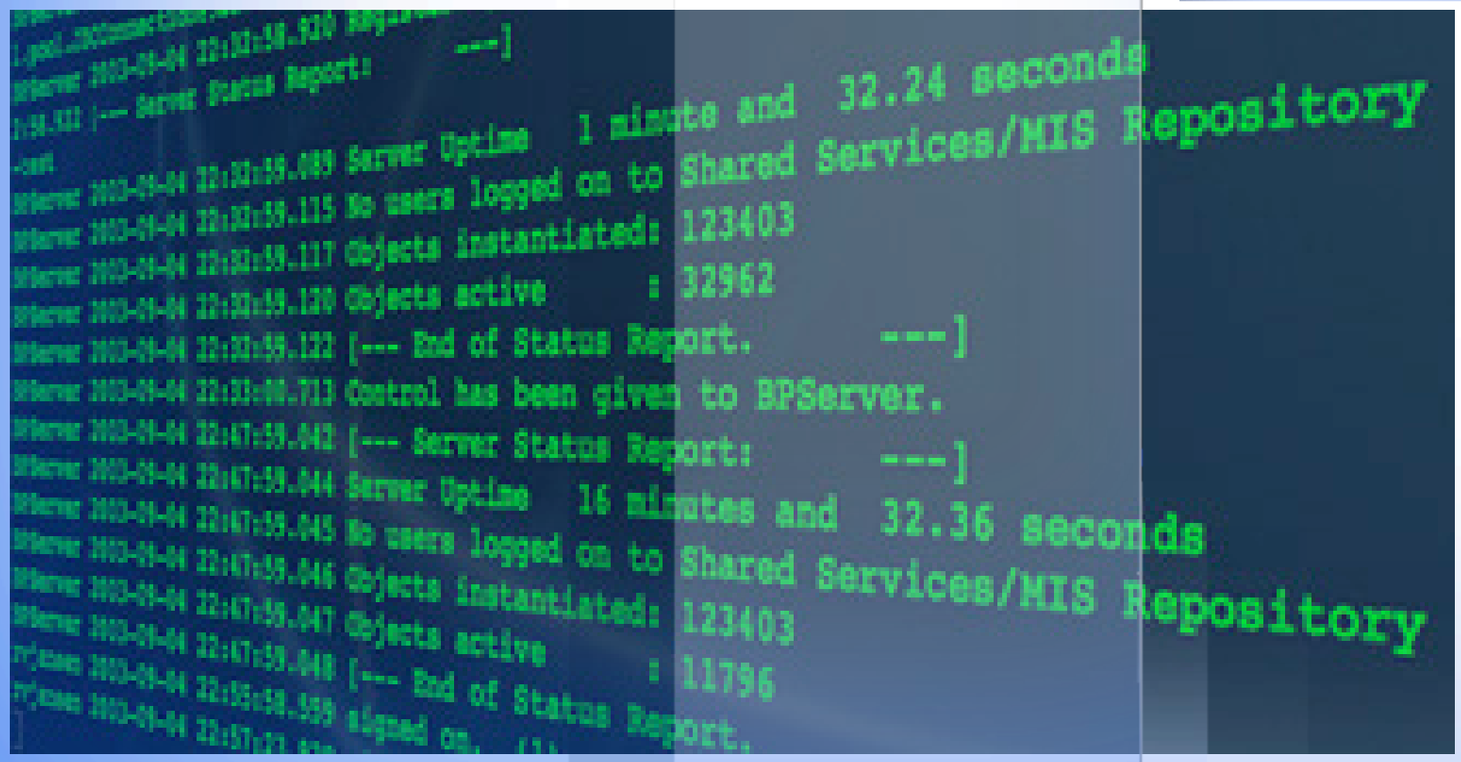
Needed robust multiuser access to the repository

Handles transactions

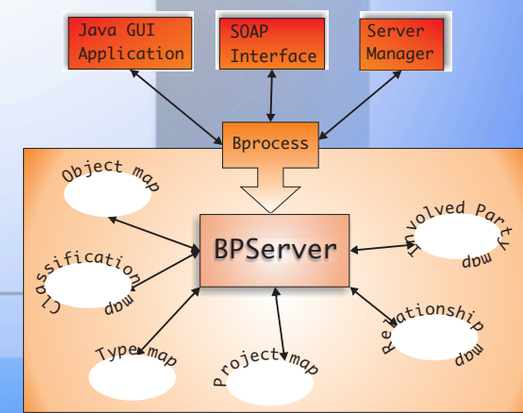
Caches Objects in storage

The Server Component

The Server is a text mode NetRexx-only Application that can run on any Java 2 VM. At startup the Relational storage is converted to Objects that are indexed and stored on Class and their participation in Relationships.



Note the cute "Control has been given to ...



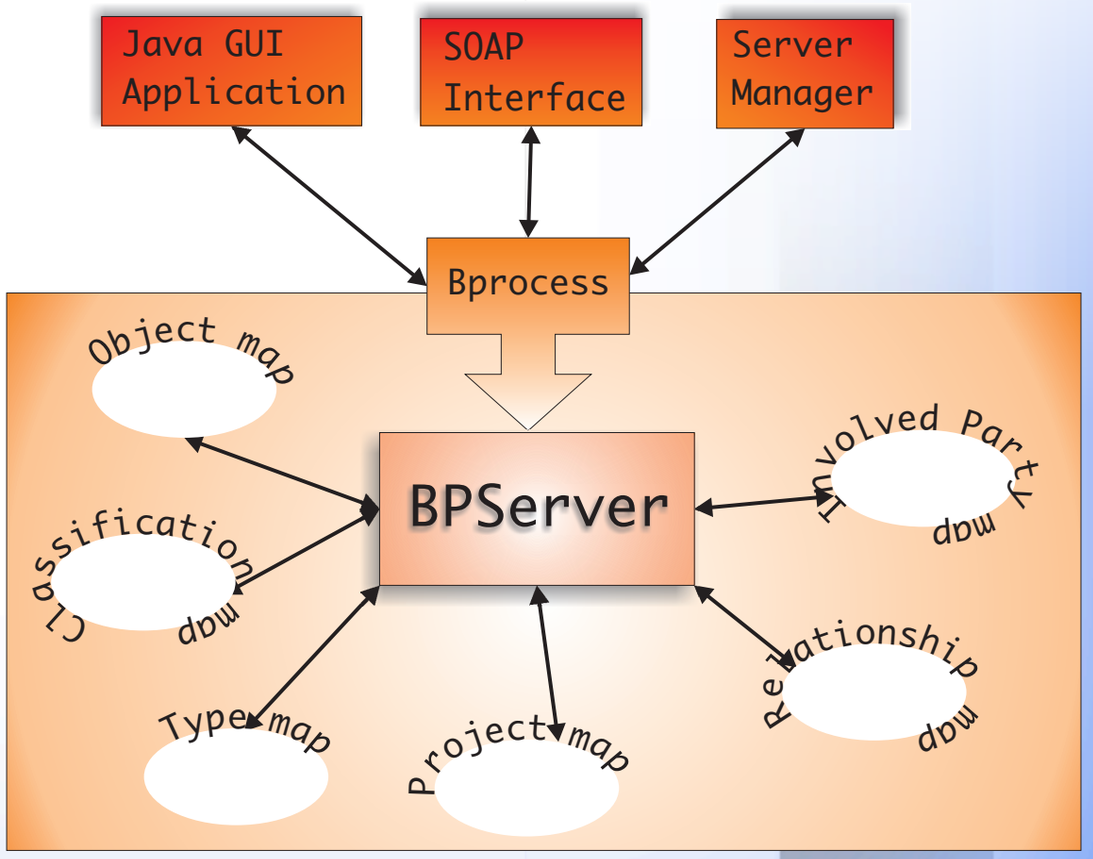
Is a Remote Object

BPServer.nrx: /Volumes/Workspace/com/abnamro/midms/RepositoryTool/BPServer.nrx

```
options strictcase
  -- we do not usually use STRICTCASE but in this file it is neccessary
  -- because the Collections class has a static method synchronizedList
  -- and also an inner class SynchronizedList etc.

package com.abnamro.midms.RepositoryTool
import com.abnamro.midms.modelbase
import com.abnamro.midms.modelbase.soap
import com.abnamro.midms.util.
import com.abnamro.midms.util.pool.
import com.abnamro.midms.util.exceptions.
import com.abnamro.midms.util.ssl.
import java.rmi.
import java.util.
import java.sql.

/**
 * The Business Process Server is the interface between the application layer
 * and the database, and between the data and the client application. <br>
 * The database is accessible via DataStore, but only the Business Process
 * Server may access it. A DBMS connectionpool is set up, and every user
 * (client) of this server uses a separate connection lease to do the work;
 * this work can thus be individually committed or rolled back.<br>
 * SK 13092002: added security manager to authenticate the user connecting to
 * BPServer.
 * @see DataStore
 * @author <a href=http://www.rvjansen.com target=new>R.V. Jansen</a>
 * @author <a href=http://zaaf.tripod.com target=new>A.J. Bos</a>

 */

class BPServer extends UnicastRemoteObject implements BPServerInterface, BPClientInterface, ScopeChanging final

  properties static inheritable
  db                    = DataStore
  oidObjectMap          = ObjectMap
  oidRelationshipMap    = TreeMap
  oidRelationshipTypeMap = RelationshipTypeMap
```

Object TreeMaps in Server

```
 * @author <a href=http://www.rvjansen.com target=new>R.V. Jansen<
 * @author <a href=http://zaaf.nl target=new>A.J. Bos</a>
 */
class BPServer extends UnicastRemoteObject implements BPServerInte

  properties static inheritable
  logger_                    = Logger.getLogger("BPServer")
  sh                         = ConsoleHandler()
  db                         = DataStoreInterface
  oidObjectMap               = ObjectMap
  oidRelationshipMap         = TreeMap
  oidRelationshipTypeMap     = RelationshipTypeMap
  oidTypeMap                 = TypeMap
  oidSchemeMap               = TreeMap
  oidRlnTpPblGrpMap          = TreeMap
  oidProjectMap              = TreeMap
  oidProjectVersionMap       = TreeMap
  oidScopeMap                = TreeMap
  oidScopeVersionMap         = TreeMap
  oidClassificationMap       = TreeMap
  oidInvolvedPartyMap        = TreeMap
  oidProjectSpecificMap      = TreeMap
  verbOidSbjRelsCache        = TreeMap
  verbOidObjRelsCache        = TreeMap
  serverLog                  = ArrayList
  tracing                    = boolean
  sessionsMap                = LinkedHashMap
  oidPropertiesMap           = TreeMap
```



Update: Hook the local Setters

Meta Model Objects have NetRexx Indirect Properties

Setter Methods are overridden to do remote update

```
    * t_. It then also adds itself
    * @see #addSubType
    */
method setSuperType(t_ = Type) signals InvalidSupertypeException

  if this.bp <> null then -- we are in the client
    do
      oldSuperType = this.superType
      if this.superType = null then
        do
          this.superType = t_
          t_.addSubType(this)
        end
      if this = t_ then
        do
          signal InvalidSupertypeException()
        end
      this.bp.updateSupertype(this.getOID(), oldSuperType.getOID(), t_
    catch RemoteException
      System.err.println("RemoteException in Type.setSupertype(" t_ ")
    end

if this.superType = null then
  do
    this.superType = t_
    t_.addSubType(this)
  end
else
  do
    if this = t_ then
      do
        signal InvalidSupertypeException()
      end
    else
      do
        this.superType = t_
        t_.addSubType(this)
        if oldSupertype <> null then oldSupertype
      end
```

Object *Decomposition*

Java
Client

New Domain Value

BPServer

Dissect and
Disassemble into
Relationships

Scope in
Domain Value

5 new Relationships

N Relationships

Data
Base

For every new domain value,
5 new relationships are added to the system,
and as much scoping relationships as there are active scopes.

The **bint** Language

*batch int*erpreter

daughter (arabic)

a strict headmaster
(in a story by F. Bordewijk)

lower class girl
(british slang,
usage: outdated)

Why?

Syntax

Examples

Antlr Grammar

Usage in Client

bint

Next to the Graphical User Interface, we also defined a specialized metadata access language, called **bint**.

bint

Due to the use of the generic universal relation, SQL access to the metadata soon becomes very cumbersome. Bint uses the objects after they are assembled in the Object Factory.

Syntax of **bint**

Syntax is very loosely inspired by CMS Pipelines, APL and Predicate Logic

Works on stacked maps of **key:Oid** and **value:Object**

The data appears to flow through pipes with stages

Interaction between Antlr and NetRexx in construction of **bint**

Antlr .g grammar file

.java Lexer Class

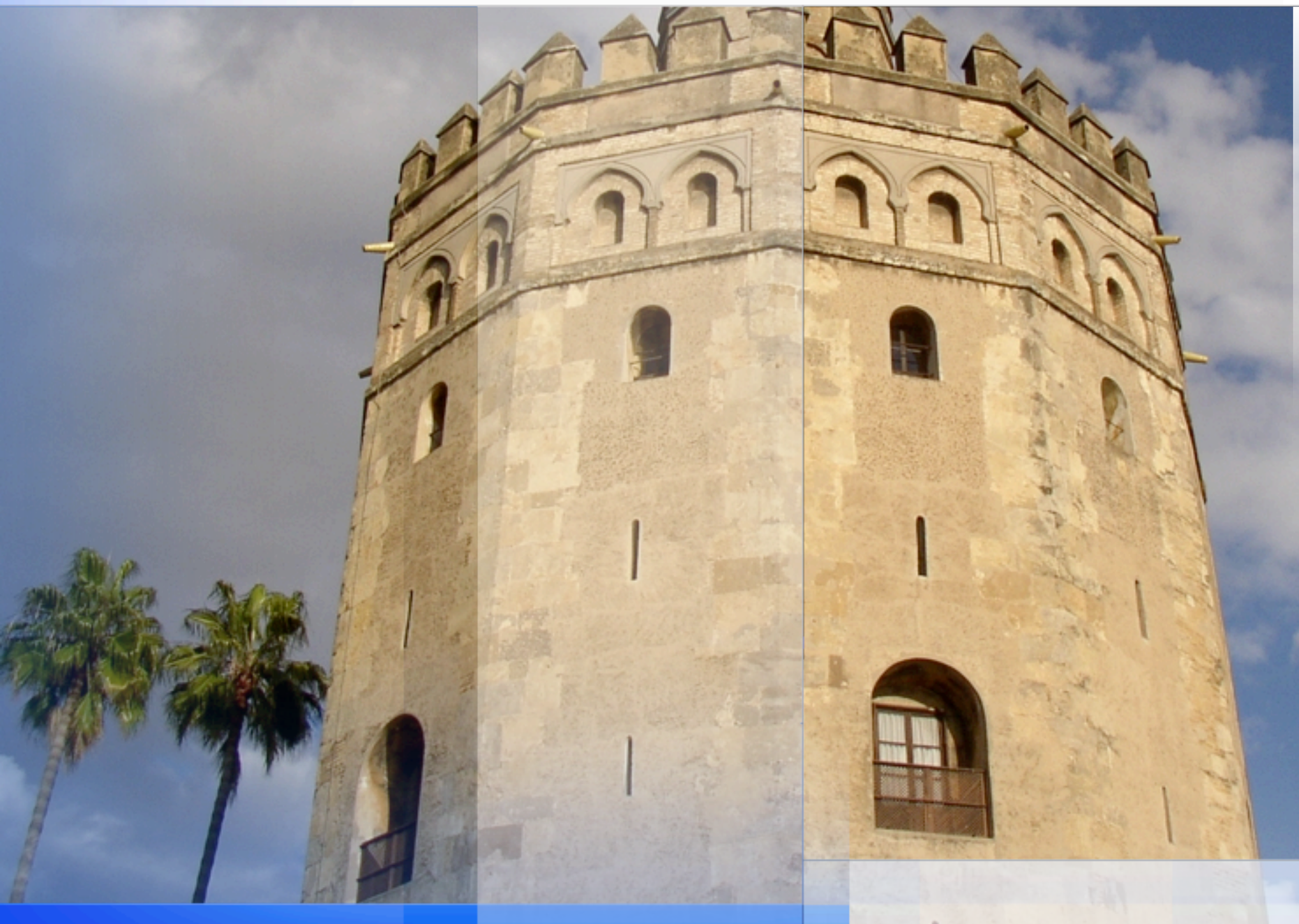
.java Parser Class

.nrx language methods interface

.nrx Driver and language methods

```
header
{
    package com.abnamro.midms.bint;
}

{
    import java.io.*;
    import java.rmi.*;
}

class bintParser extends Parser;
options {
  k = 2;
  exportVocab=bint;
  codeGenMakeSwitchThreshold = 2;
  codeGenBitsetTestThreshold = 3;
  buildAST = false;
}

{
 public bintInterface bintInstance ;
}

program
    : (statement)+ EOF
    ;

statement
    : connectto
    | disconnect
    | settimer
    | setverbose
    | setcolumnseparator
    | setsyntaxcheck
    | collect
    | subset
    | list
    | listproperties
```

Infrastructure and Security



Platform

Development Environment

Middleware Components

RMI &SSL

JNI Based User Authentication

Platform

Microsoft Windows Server System

Due to the Application being all NetRexx, it runs unchanged on Windows NT, MacOS X, Windows XP and Linux.

No testing has been done on z/OS, but we are confident that it will run within a day.

Production Server is a Dual Xeon with 1 GB storage under NT 4.

Development Environment

Emacs, Java, NetRexx, Netbeans,
CVS, Make

← Switched recently to *Subversion*

On NT, we used *Cygwin* (so we have (the taste of) Unix anywhere)

NetRexx editor mode is very important (color, indentation)
(Elisp, cross-platform)

The NetRexx Compiler Server

DBMS

MySQL

PostgreSQL

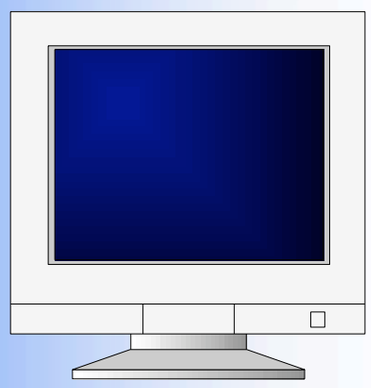
DB2 UDB 8.1 is used for production

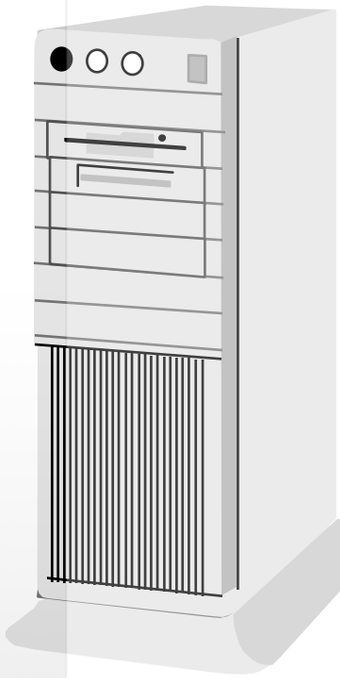
Development and regular testing takes place on the Open Source DBMS'ses MySQL and PostgreSQL

The Server Component can run off MS-Access via ODBC-JDBC Bridge (for people who need to travel with copies on generic Windows Laptops)

All DBMS access is done by Server Component using JDBC in NetRexx

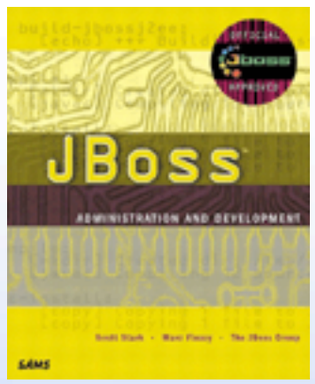
Middleware





Tomcat is used as Servlet Container

Axis is used for SOAP access to server



We just migrated these to JBoss

RMI and SSL Encryption

```
package com.abnamro.midms.util.ssl
import java.io.
import java.net.
import java.rmi.server.
import javax.net.ssl.
import java.security.KeyStore
import javax.security.cert.X509Certificate
-- mind: if this does not compile you probably do not have jsse.jar on your classpath
class RMISSLServerSocketFactory implements RMIServerSocketFactory, Serializable

  method  createServerSocket(port=int) returns ServerSocket signals IOException
  do
    -- set up key manager to do server authentication
      passphrase = char[]
      passphrase = "passphrase".toCharArray()

      ctx        = SSLContext.getInstance("TLS")
      kmf        = KeyManagerFactory.getInstance("SunX509")
      ks         = KeyStore.getInstance("JKS")

      ks.load(ClassLoader.getSystemClassLoader().getResourceAsStream("dmskey"),passphrase)
      kmf.init(ks, passphrase)
      ctx.init(kmf.getKeyManagers(), null, null)
      ssf = ctx.getServerSocketFactory()

    catch e=Exception
      e.printStackTrace()
    end
                                    verSocket(port)
```

JNI Based User Authentication

```
JNIEXPORT jstring JNICALL Java_com_abnamro_midms_platform_NTPlatformSecurity_getUserId(JNIEnv *env, jobject o
{
  jchar * buffer[100];

  DWORD length = 100;
  NET_API_STATUS rc = 0;

  rc = GetUserName((LPTSTR)buffer, &length);
```

```
/** This Class is the implementation of the methods of PlatformSecurity for the Windows NT Operating System.
    Most of the methods are native WIN32 code, implemented in NTPlatformSecurity.c and loaded from
    NTPlatformSecurity.dll. It is know to run on NT and W2K */

class NTPlatformSecurity extends PlatformSecurity

    /** This constructor loads the NTPlatformSecurity Dynamic Link Library for Windows NT */
method NTPlatformSecurity()

    System.loadLibrary("NTPlatformSecurity")

/** This method overrides PlatformSecurity.getUserid() and is implemented in a native method, which uses the
    call <code>
    rc = GetUserName((LPTSTR)buffer, &length);
    </code>
    to get its results. This is deemed more secure than using the Java System Property user.name
    */
method getUserId() native returns String
```

Post Mortem

These points were moot since we had a working prototype database in MS-Access

That sometimes failed spectacularly, for example when adding RI dropped a random index ...

Our three-tier NetRexx solution never failed yet.

- We wasted some time by being not generic enough, for examples in the hooks in the setters - we invented class metadata in BeanInfo objects
- We should have used more association objects instead of collections in classes - only one place to update and less RMI trouble
- Dyadic relationship modelling is hard

Questions?

- Ask them now
- By email
  rene.vincent.jansen@nl.abnamro.com
- Or alternatively
  rvjansen@xs4all.nl

Thank you very much for your attention!