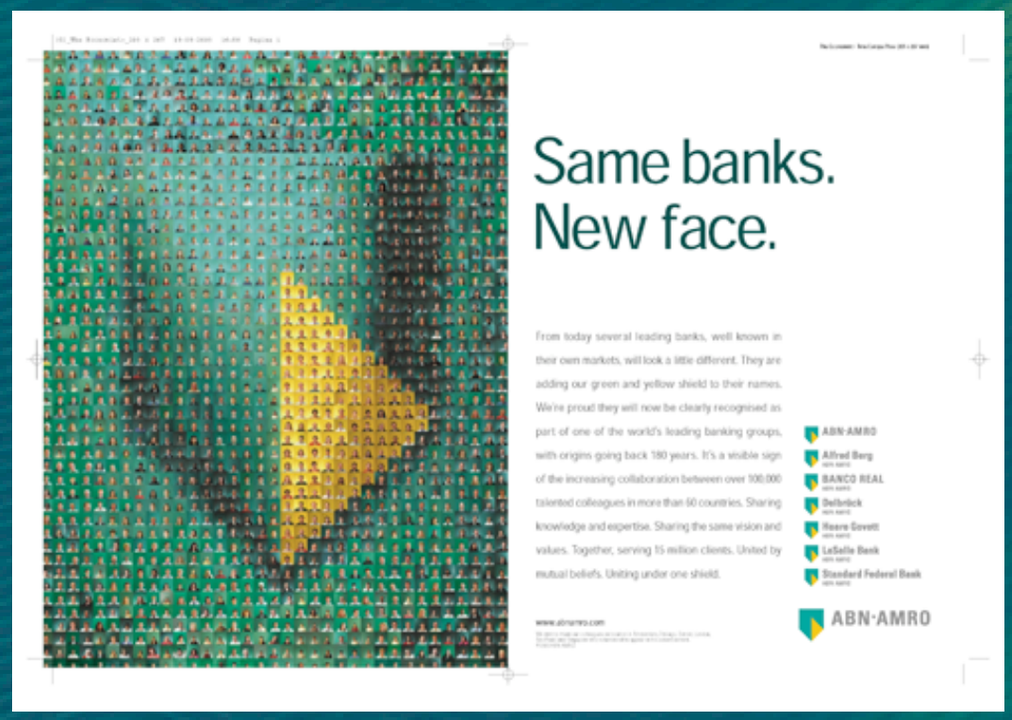
ABN·AMRO

NRX JSF MQ Hibernate

J2EE NetRexx with Java Server Faces WEB GUI
MQ Messaging Interface
and Hibernate Object/Relational Persistence Layer

René Vincent Jansen
I-Bizz IT Services and Consultancy
Rexx LA, LA 2005-04-18



Legal and Disclaimer

I-Bizz IT Services and 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.

About 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).

ABN·AMRO

Alfred Berg
ABN AMRO

BANCO REAL
ABN AMRO

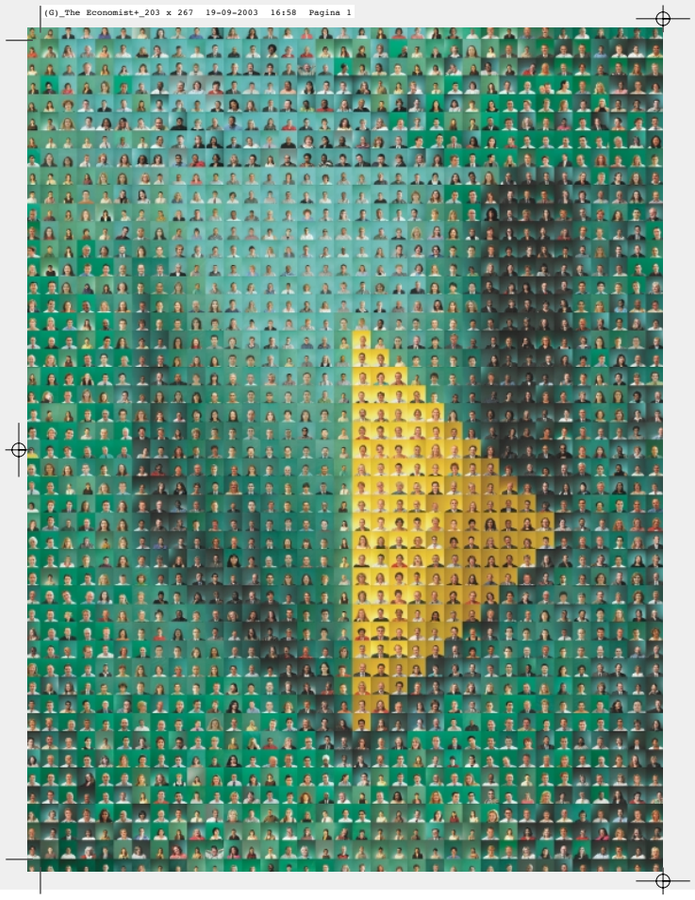
Delbrück
ABN AMRO

Hoare Govett
ABN AMRO

LaSalle Bank
ABN AMRO

Standard Federal Bank
ABN AMRO

ABN·AMRO



The Economist • Time Europe/Asia (203 x 267 mm)

Same banks. New face.

From today several leading banks, well known in their own markets, will look a little different. They are adding our green and yellow shield to their names. We're proud they will now be clearly recognised as part of one of the world's leading banking groups, with origins going back 180 years. It's a visible sign of the increasing collaboration between over 100,000 talented colleagues in more than 60 countries. Sharing knowledge and expertise. Sharing the same vision and values. Together, serving 15 million clients. United by mutual beliefs. Uniting under one shield.

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

www.abnamro.com

We wish to thank our colleagues on location in Amsterdam, Chicago, Detroit, London, São Paulo and Singapore who volunteered to appear in this advertisement.

Common Reference Data Project

Making more possible

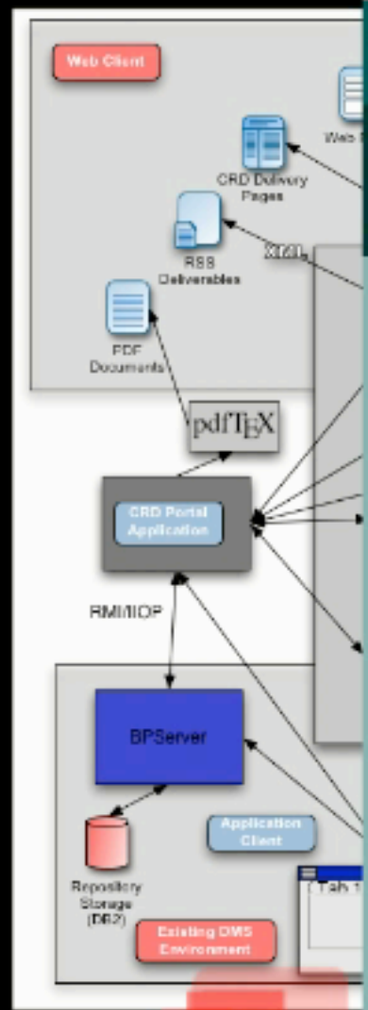
ABN·AMRO

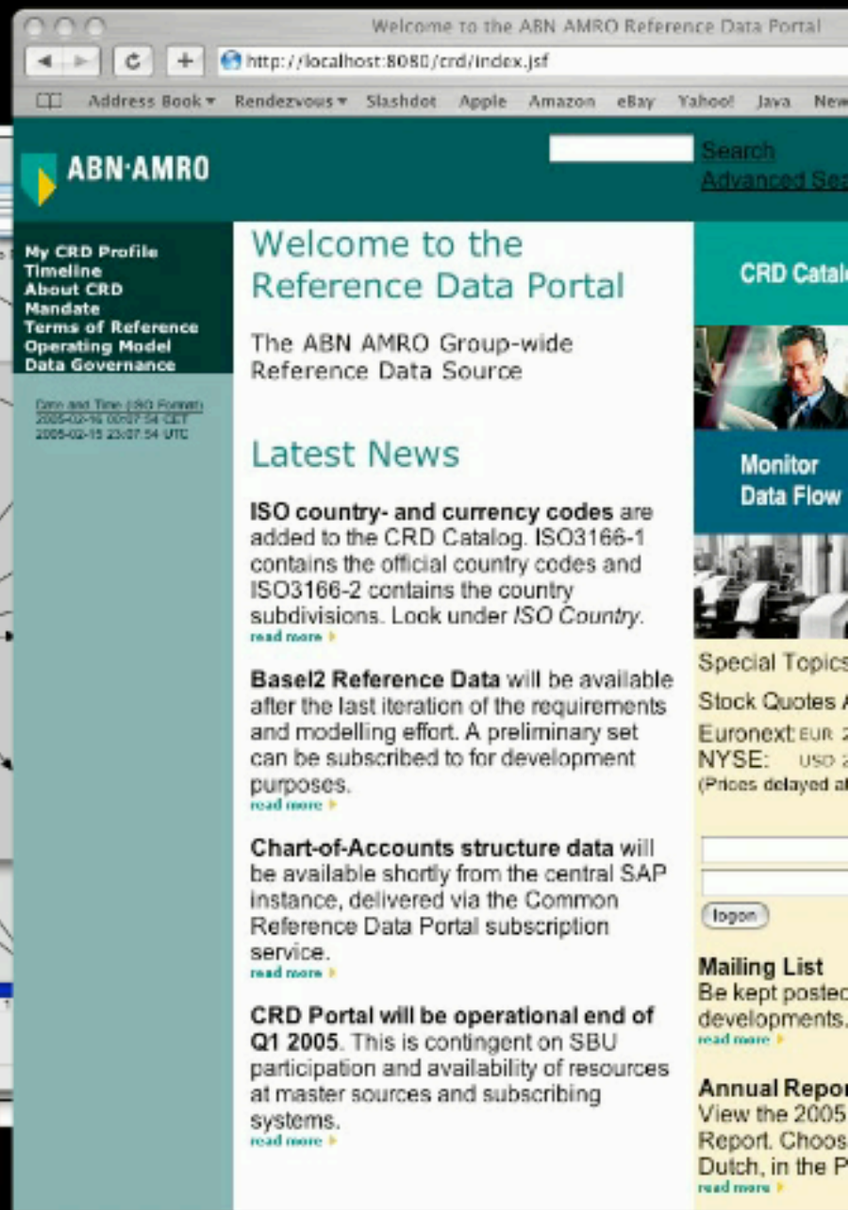
Agenda

Portal

Without a portal
With a portal
Portal Architecture
Build Components
Infrastructure
User Agent
Cache Strategy
Instrumentation
Multilingual Support
Propagation
Screenshots

CRD Technical Architecture





Reference Data Definition: Chisholm

A1

Reference data : data used solely to categorize other data found in a database, or solely for relating data in a database to information beyond the boundaries of the enterprise.

Transaction structure data : (e.g. Client, Product data) represents data required to create a framework within which transactions occur. It has the scope of the types of transactions to which it pertains in information systems. It typically has a much higher volume and update rate than Reference data.

Enterprise Structure data : describes the structure of the enterprise, e.g. organisational structure or chart of accounts. The information is used to track business activities by responsibility. It has a higher update rate than reference data.

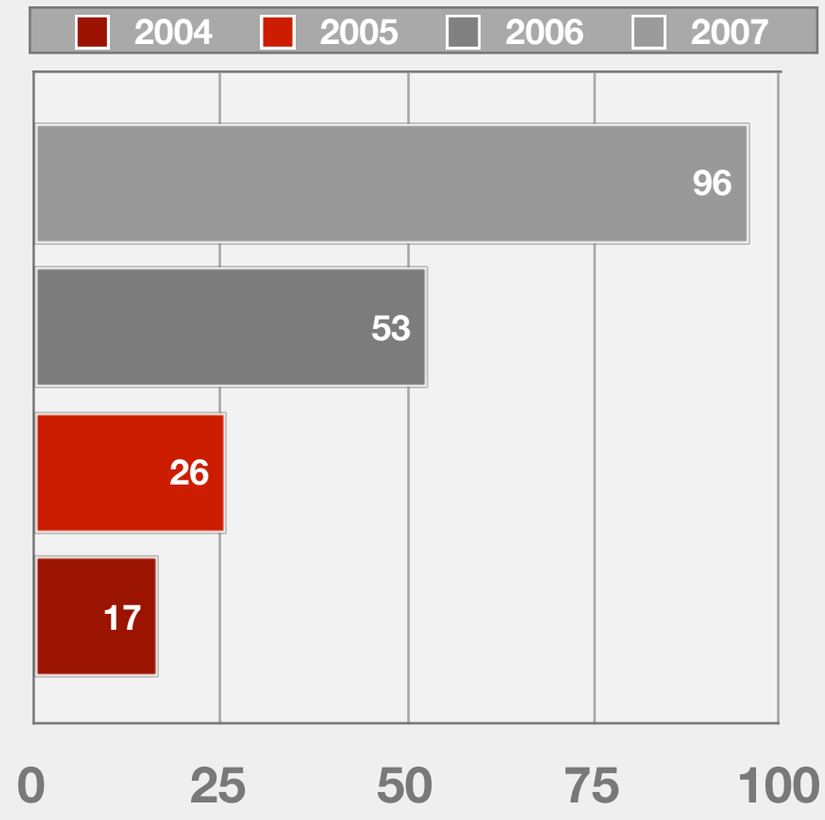
Scope: Data

A1

Initial Scope Is **Basel II Reference Data**[1]

Use **ISO** Standards Source Data in **pilot project**

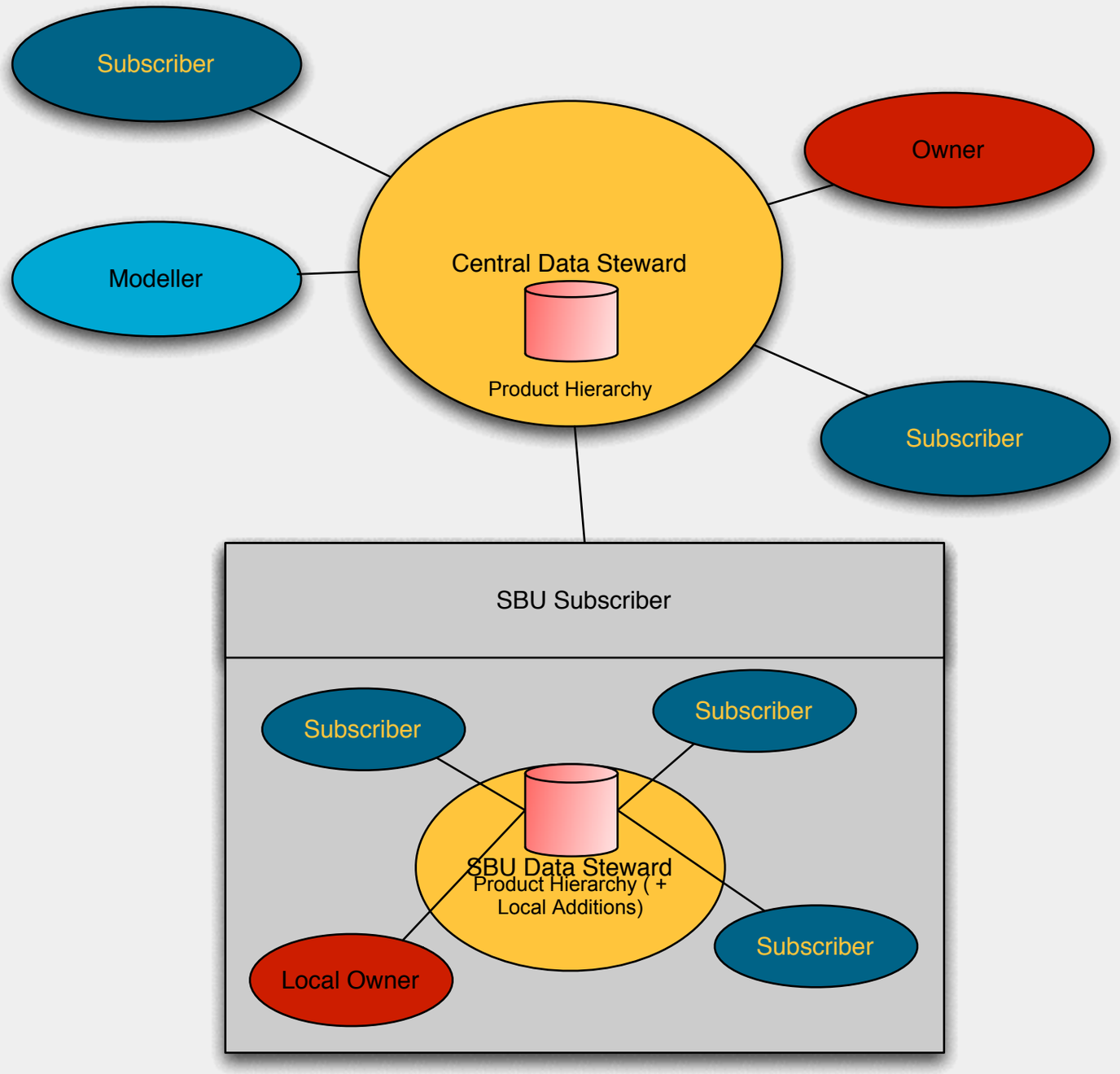
- Task: describe the scope of the common reference data as a function of time.
- Think Big, Start Small
- First increments dictated by Basel II Reference Data requirements, with clear timeframe



This is an indication of the expected progress in covering the scope of the Common Reference Data project.

The actual achieved scope is dependent on many variables such as requirements, realization of budgets and chosen implementation path.

Data Governance Agents and Roles

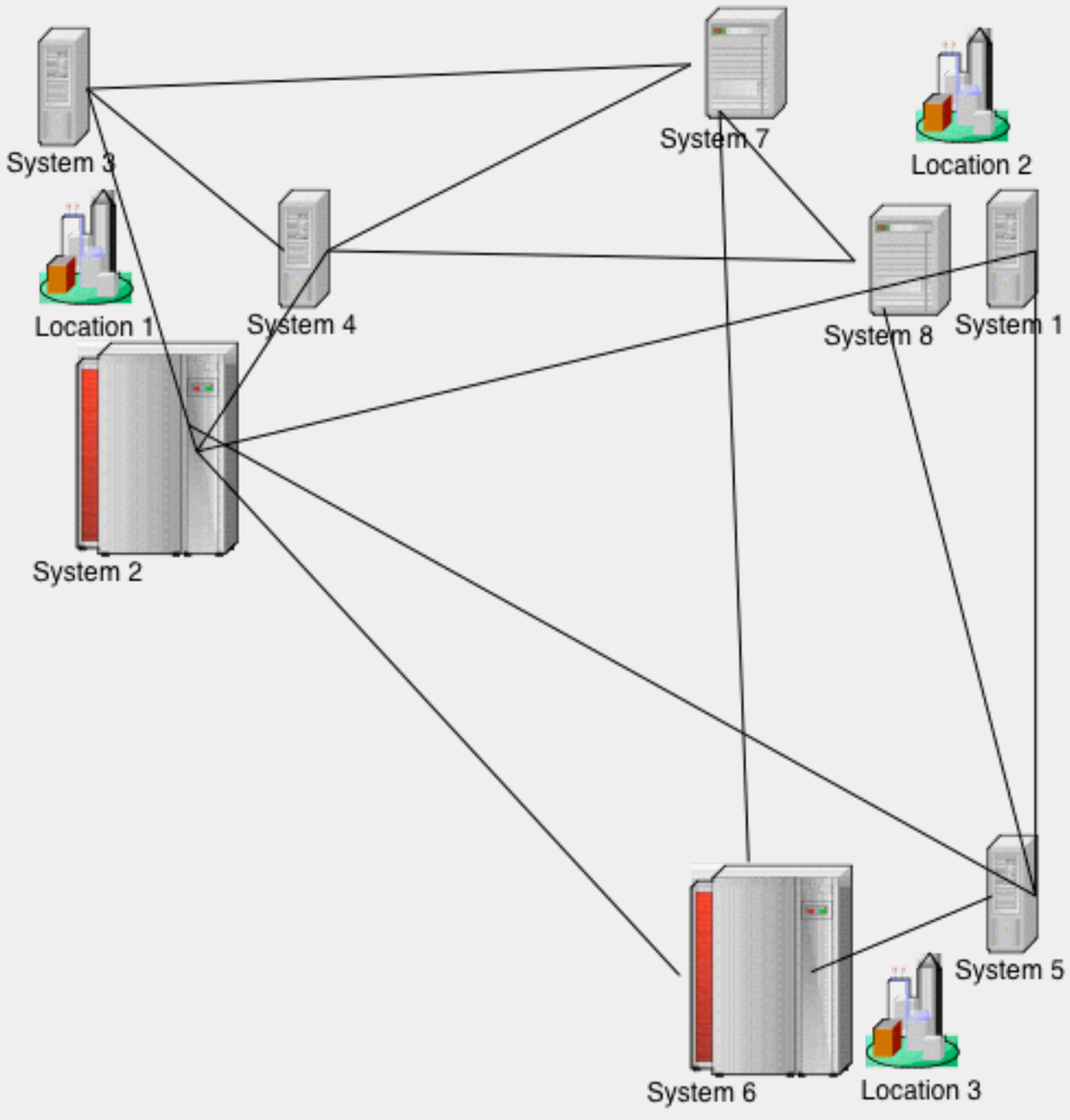


There are Data Owners, Data Stewards and Subscribers to Reference Data

Data Owners and Data Stewards have local counterparts

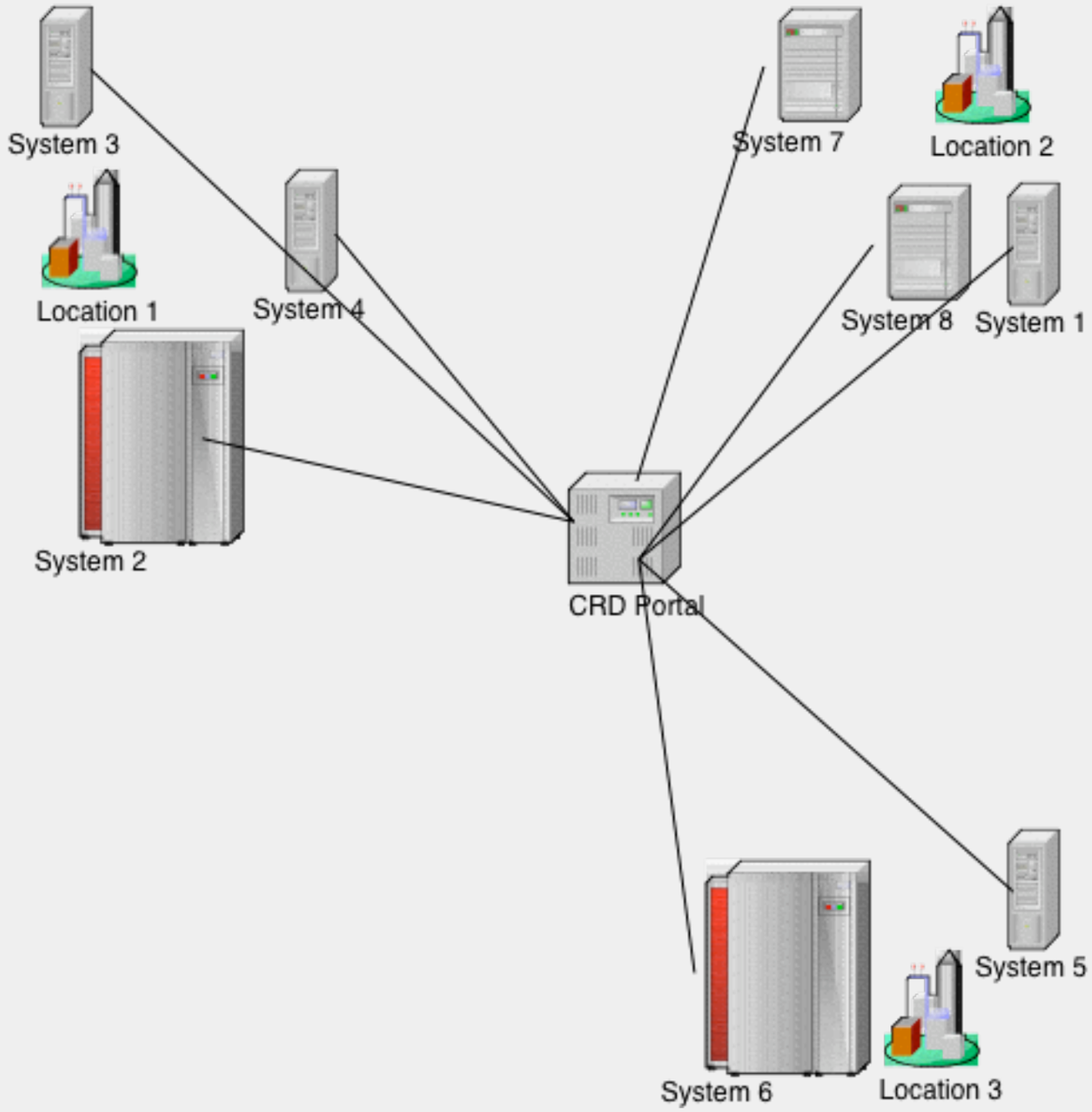
Without a Portal

- Numerous point-to-point connections between systems
- No central control
- Need to develop multiple expensive interfaces
- No governance of definitions

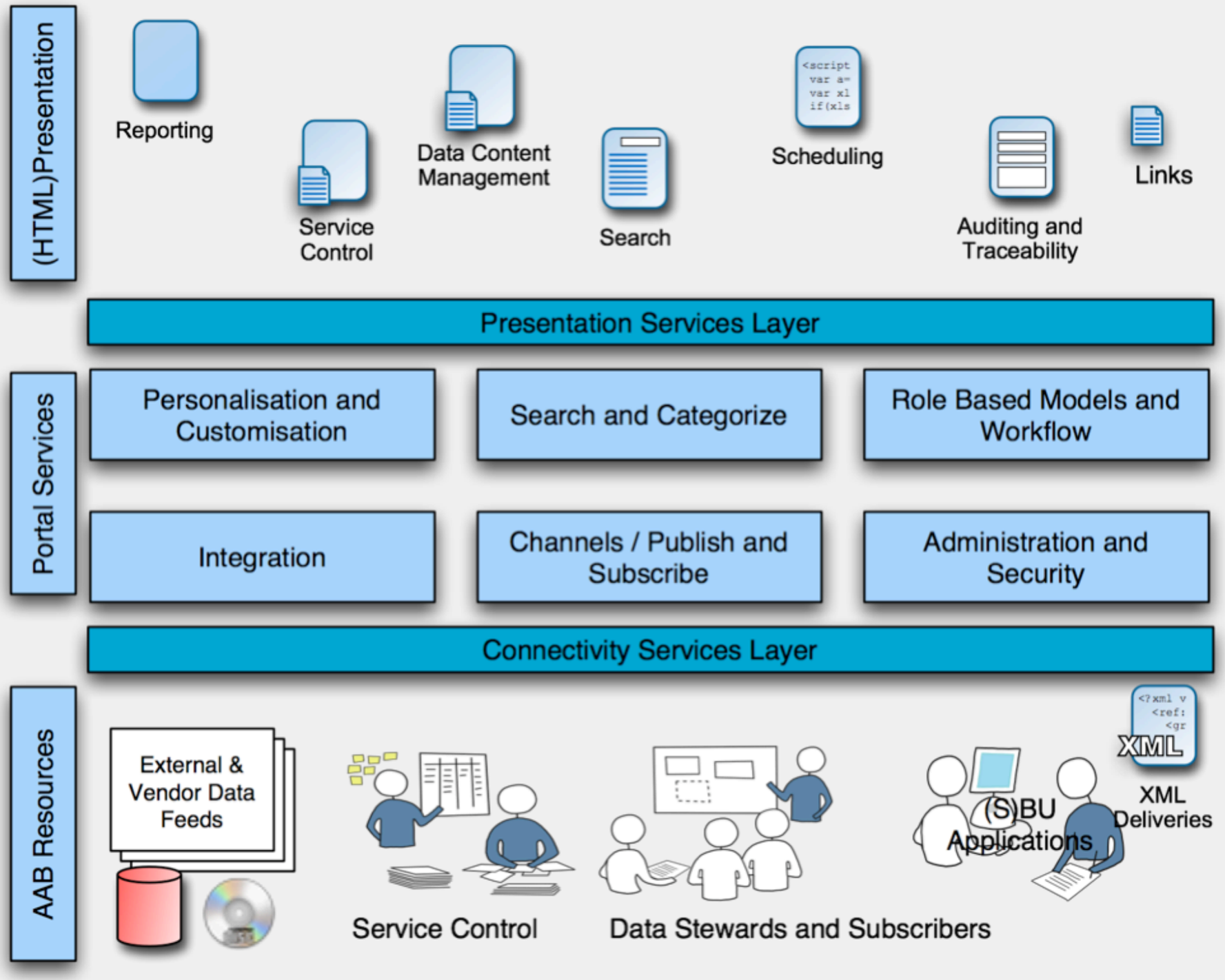


With a Portal

- One single access point for CRD in AAB
- Shielding subscribers from actual location of data
- High flexibility in changing Master Sources
- Central record of all CRD, systems, owners, subscribers, schedules, delivery methods, delivery content
- Facilitate central governance over CRD
- Facilitate roll-out of standards : message formats, exchange protocol



Portal Service Architecture



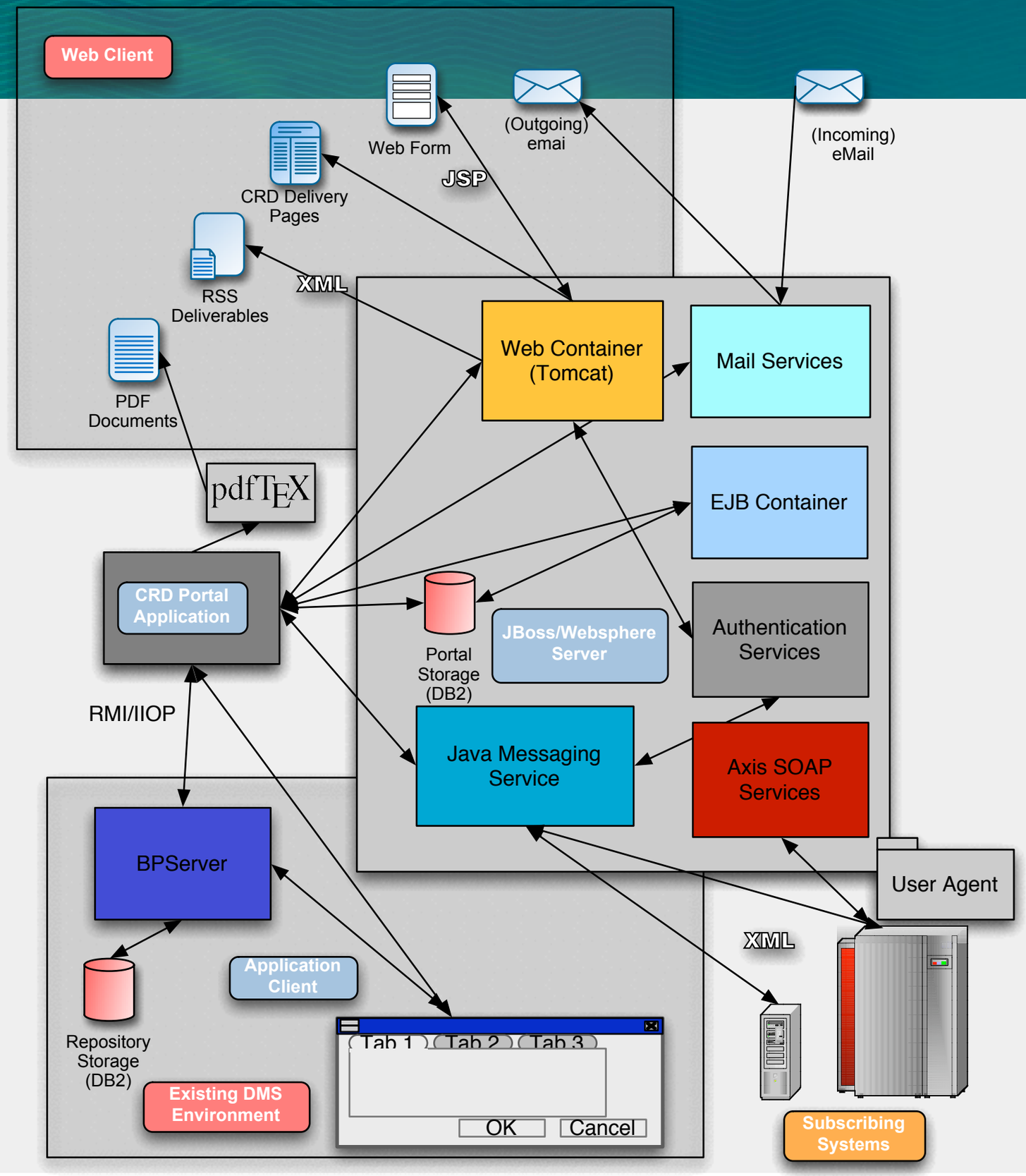
Technical Architecture

The infrastructure is a standard J2EE Container

All code is written to J2EE API specifications

There are a number of different J2EE Containers on various platforms that can run this application

The target environment is IBM's WebSphere Application Server.



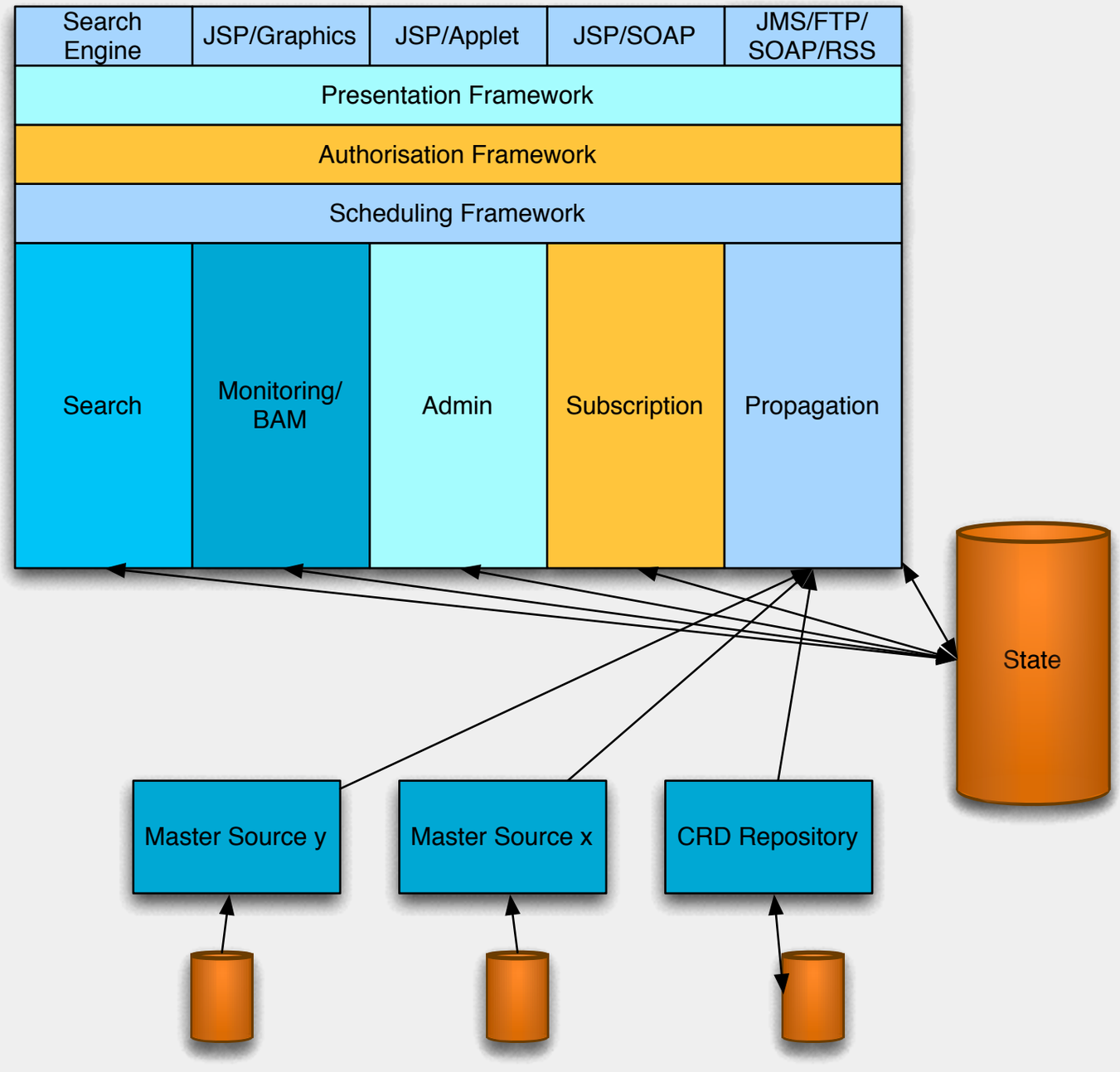
Proof of Concept Portal Build Components

There are a number of distinct components that implement functionality as specified in the Operating Model

The package hierarchy follows this component structure

During the proof-of-concept only these elements of the Portal Architecture will be implemented

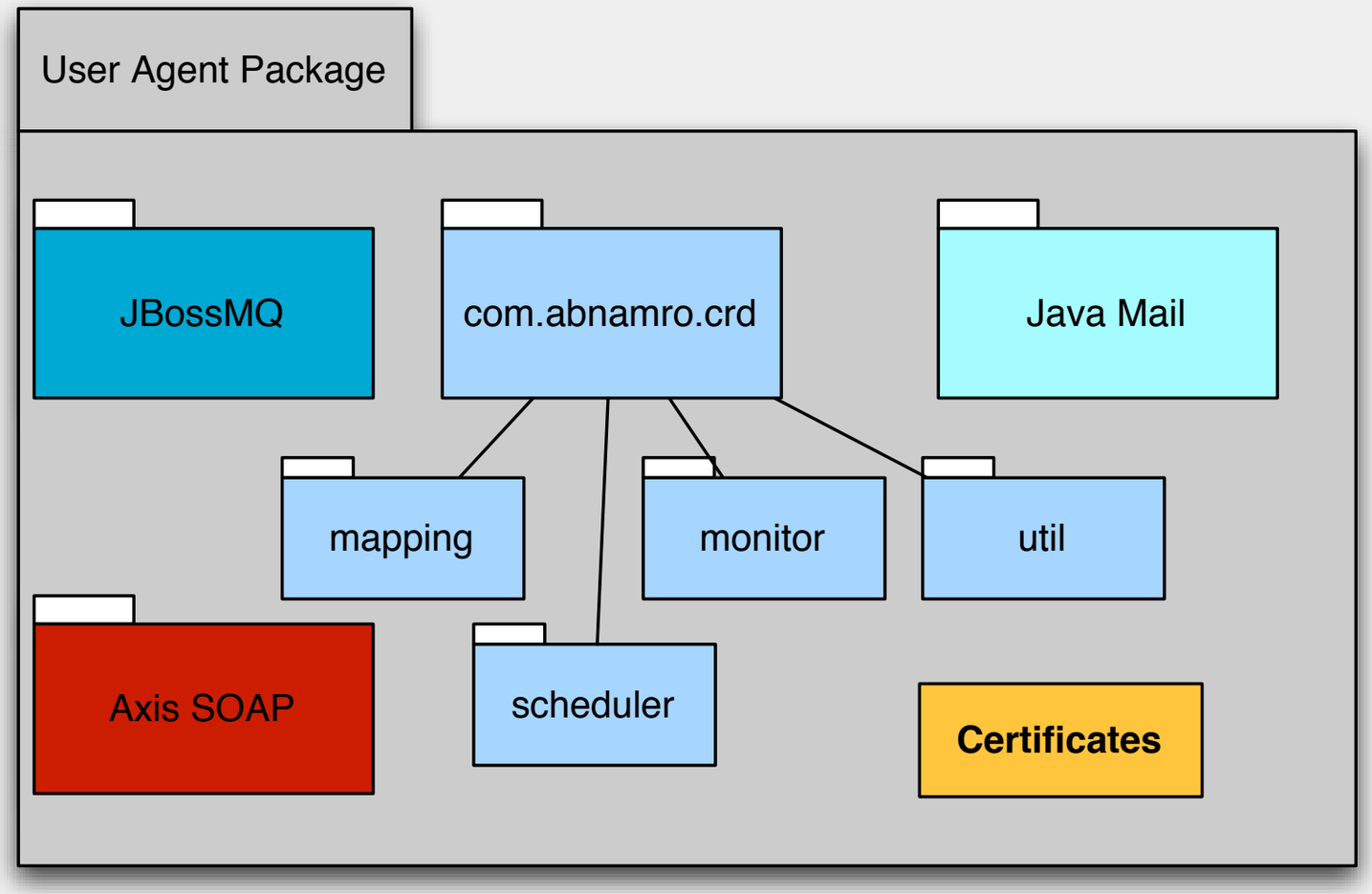
After this phase the requirements analysis will be re-iterated, leading to product evaluation and buy-or-build decision resulting from an RFI/RFP process



User Agent

The user agent package contains functionality for MQ functionality and statistics feedback.

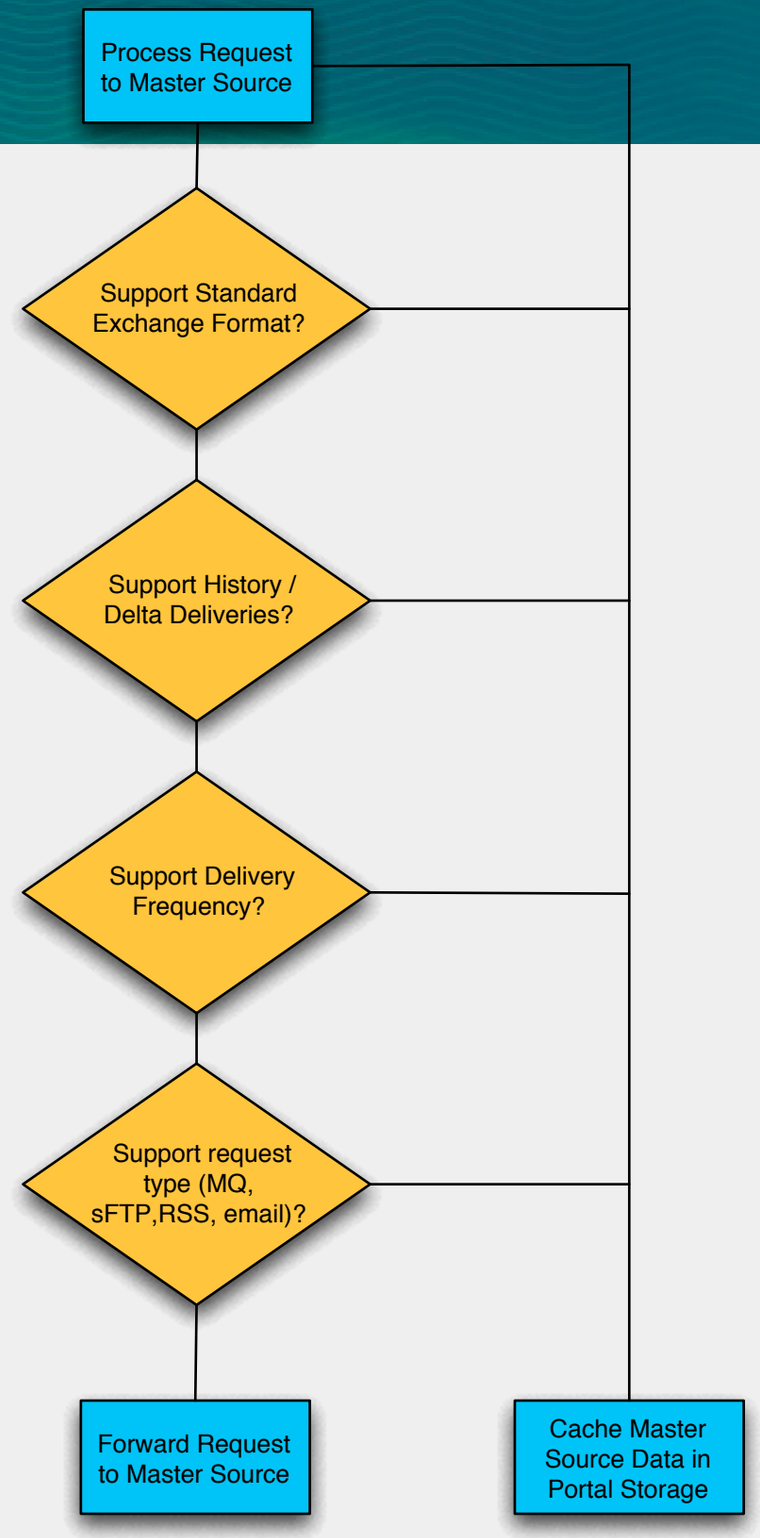
This enables every BU to participate in CRD propagation regardless of availability of messaging infrastructure.



Cache Strategy

The most pure implementation of the CRD Portal as a **broker** is feasible when the Master Sources implement the minimal facilities required.

When they do not (yet), the Portal Application can cache the reference data for delivery in its own storage. It becomes a subscriber to changes to the data respective to the Master Source.

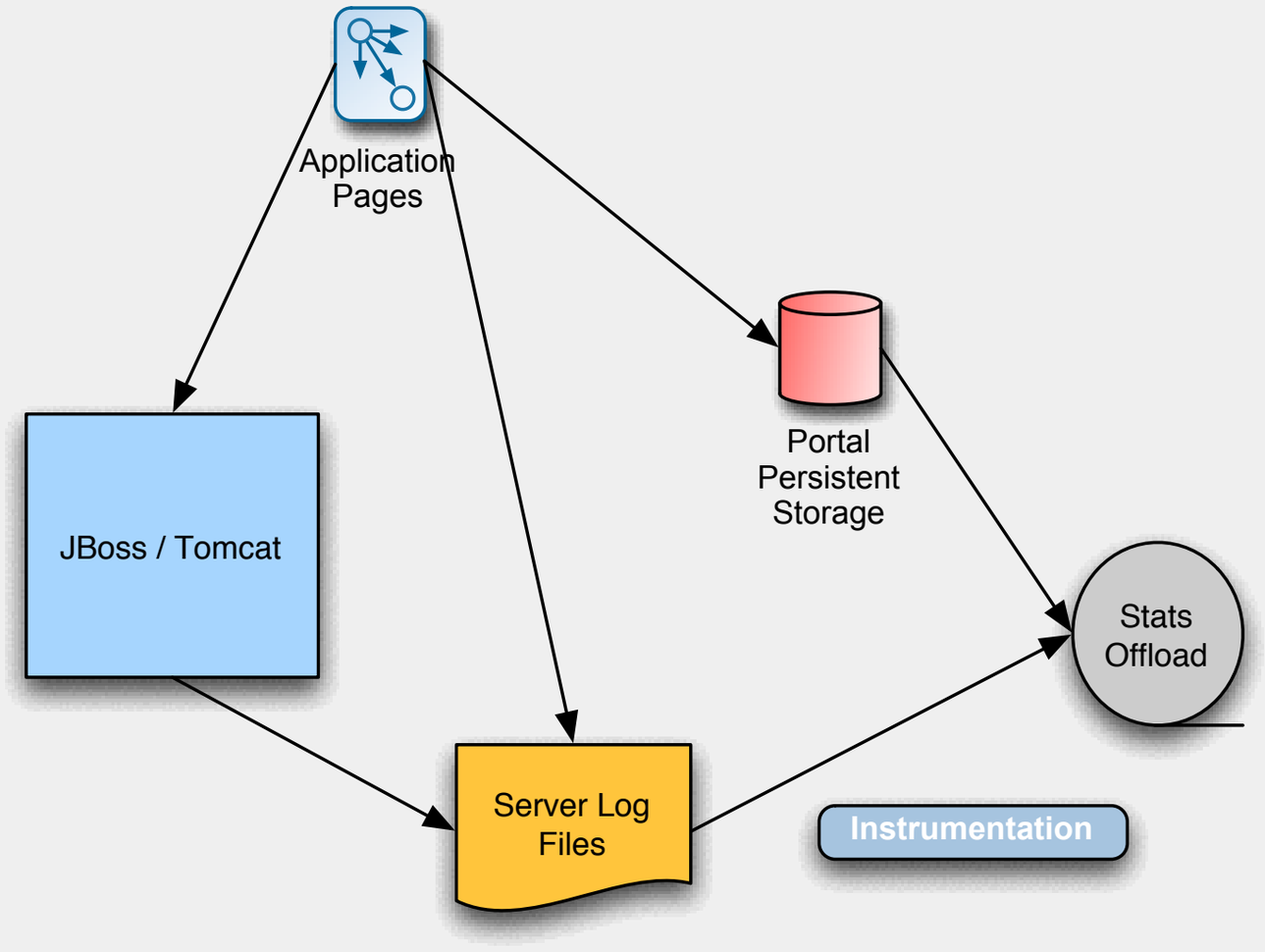


Instrumentation

Instrumentation in the various components implements the requirements for **statistics** and **usage feedback**.

The proof-of-concept will have a mix of information gleaned from standard infrastructure logs and custom logging in application code

For the presentation of the infrastructure logging produced data an off-the-shelf component can be used



Coded Character Sets and Multilingual Aspects

Multilingual, multi-alphabet support is a *sine qua non* in any AAB application

There is great variety in support for coded character sets and other language aspects in rdbms engines and operating systems. Unicode support by itself is not enough (if at all available). The exchange format takes this into account by url-encoding characters when not basic latin.

Development Environment

Language	Java / NetRexx	1.4.2/2.0.5
DBMS	DB2	8.1
Application Server	JBoss	3.2.3
Version Management	Subversion	1
Application GUI	NetBeans	3.6
Enterprise Messaging	JBoss MQ	3.2.3
Build Tool	Make / Ant	3.79/1.6.2
Interface Framework	JSF	1.0.1
Web Container	Tomcat	4

Development Principles

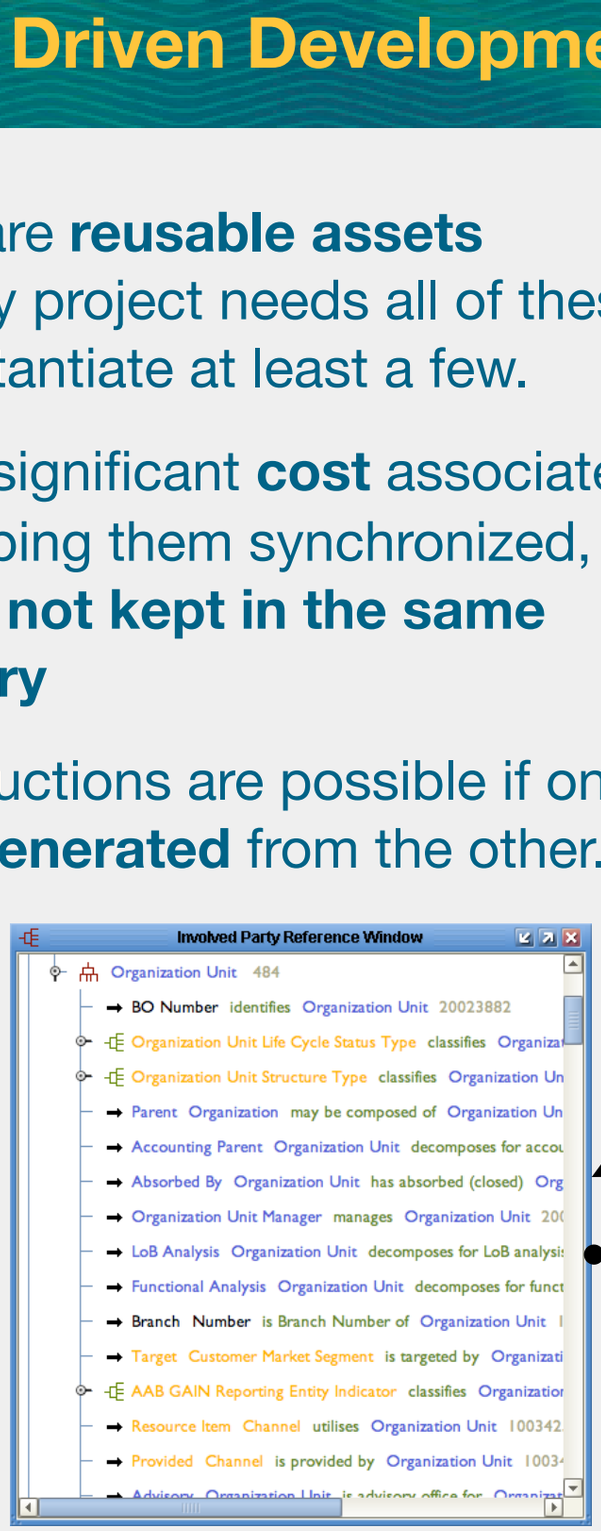
Everything is checked in into version management immediately	Do **not break** the build	The system builds **from scratch**
All is **generated from** and **documented in** the DMS Repository	**Share the knowledge**, so everybody can step in when needed	There is **no ownership** anywhere
We write to **standards**, not products	Most (if not all) classes have their **unit tests built in**	**Avoid** code **inter-dependencies**

Model Driven Development

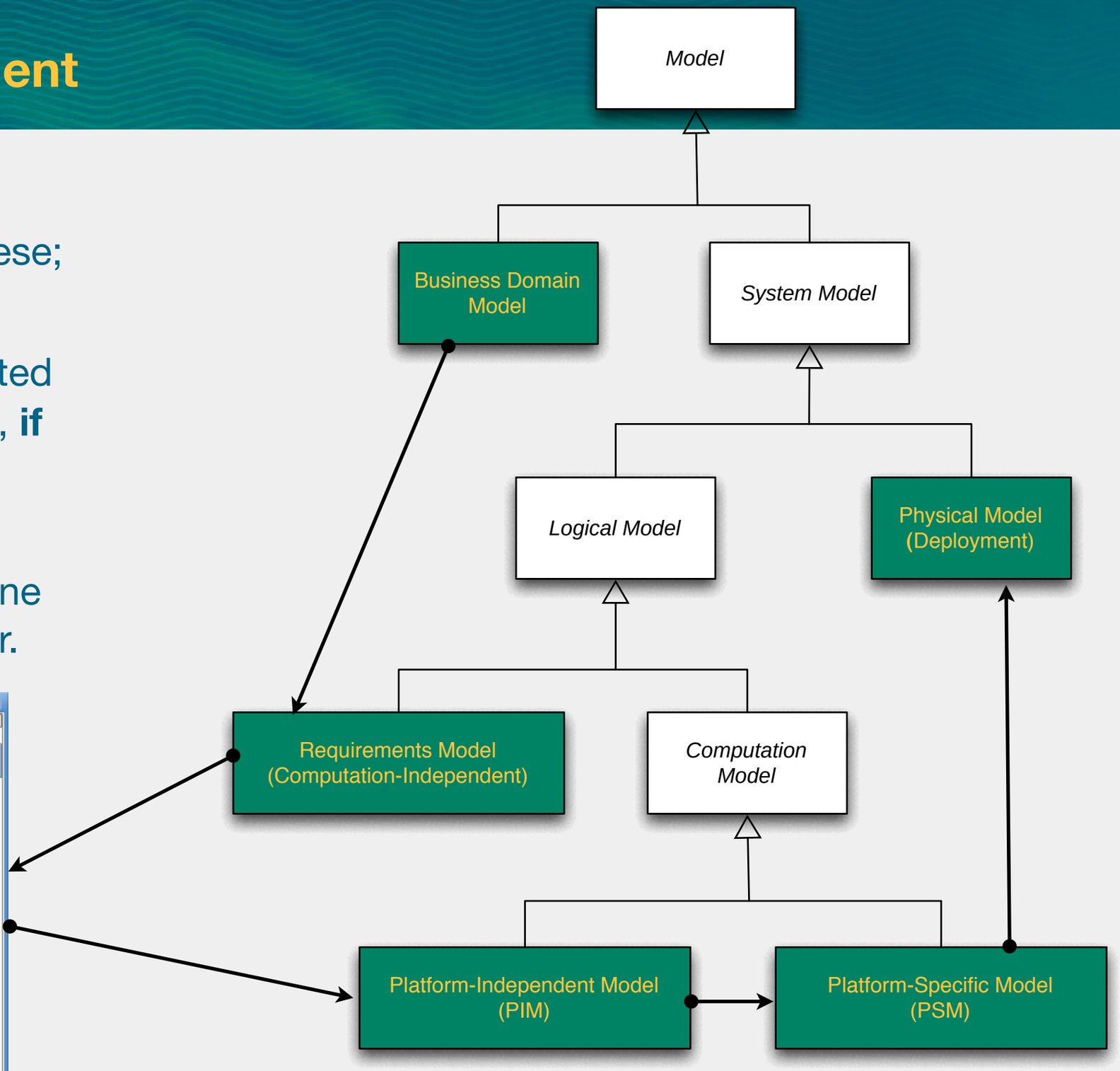
Models are **reusable assets**
Not every project needs all of these; most instantiate at least a few.

There is significant **cost** associated with keeping them synchronized, **if they are not kept in the same repository**

Cost reductions are possible if one can be **generated** from the other.



AAB Logical MIS Model Repository



Model Levels: Model, Meta Model and Meta-Meta Model

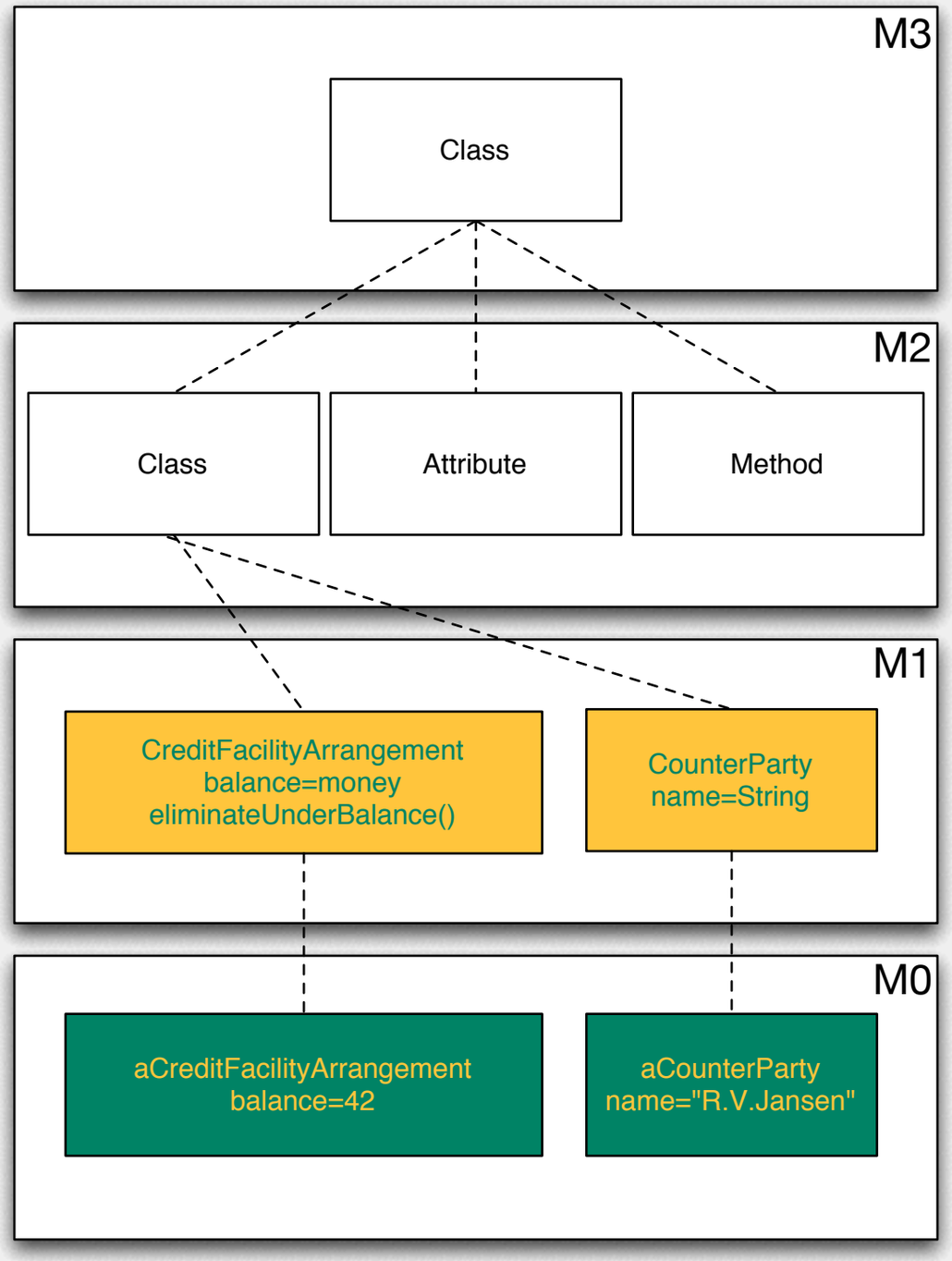
- Every instance item carries its instance UUID and its type UUID
- The semantic mapping is done on the model level

M3: The **meta meta model** (repository meta model)

M2: The **meta model** (repository model)

M1: The **application model** (classes)

M0: The (instance) **data** of the application



Meta Level Tags

The name of this document

Project and Version scope

Entity

Key(s)

Attributes and their Types

```
<exchange-document name="test">
 <scope name="Basel II Reference Data" version="1.2">
  <extension name="ISOCountry" uuid="23A3D87A-7E8E-11D9-A4FF-000393123340">
   <identifying-attribute-set>
      <attribute-name>Oid</attribute-name>
   </identifying-attribute-set>
    <tuple>
     <attribute name="Oid" type="integer">2134</attribute>
     <attribute name="PrimaryName" type="string">NL</attribute>
     <attribute name="Description" type="string">Nederland</attribute>
    </tuple>
  </extension>
 </scope>
</exchange-document>
```

Designed to enable straightforward transformations to Relational Tables (including keys) using standard ETL

Delta Deliveries

- Every deliverable is a delta deliverable when the request from the subscriber contains an “all after YYYY-MM-DD” clause
- When delta processing is not needed, current data is implied (as valid on transmission date) and effective dating can be foregone
- Even in delta deliveries, there is no need for eff_dt and end_dt when current data is implied.
- Current data is implied when **eff_dt** $\leq$ **now && end_dt == highdate**
  - Effective dating is always needed when
    - **end_dt <> highdate** (we need to state that validity will end)
    - **eff_dt > now** (for a scheduled fact, a future valid value)

Portal

Queue definitions

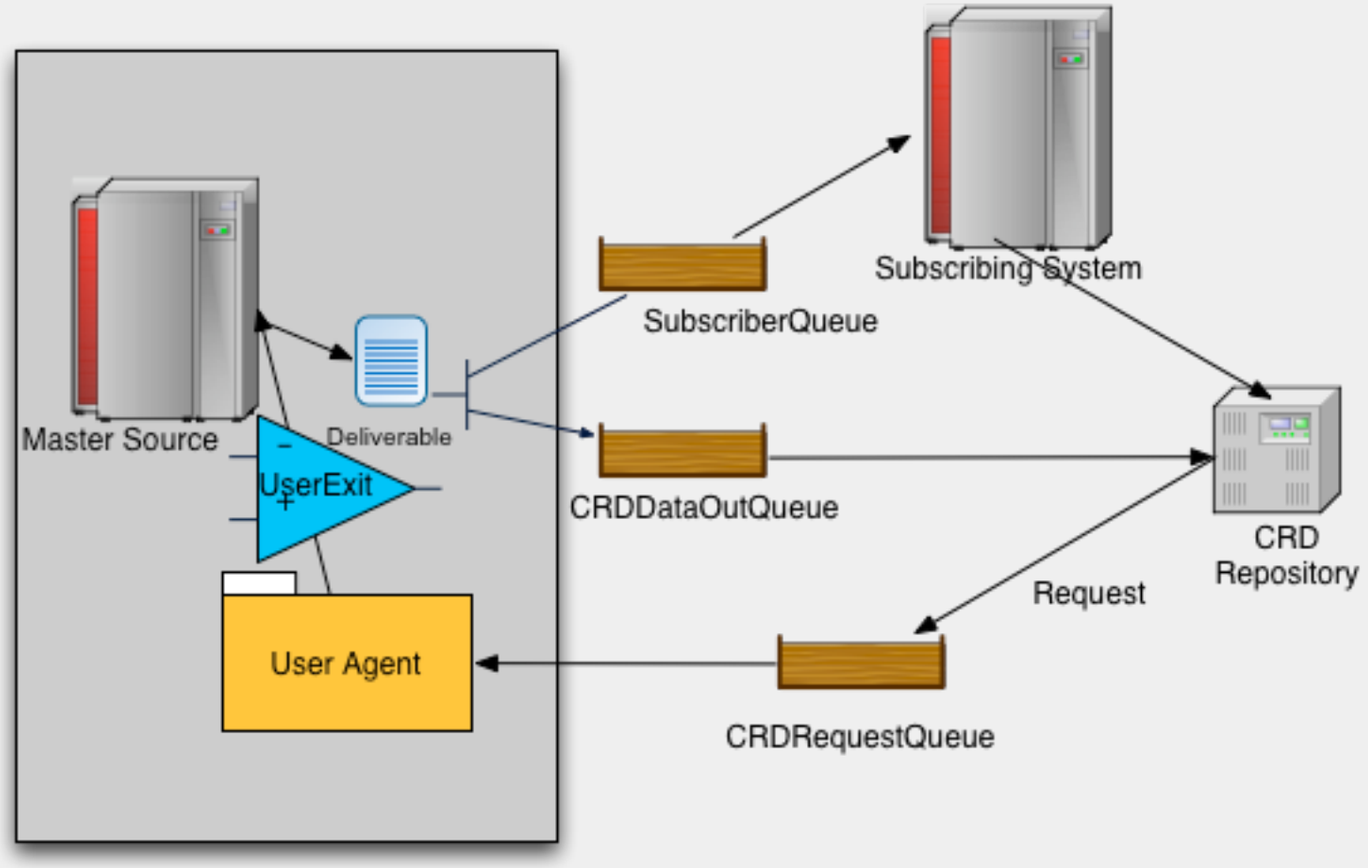
CRDRequestQueue	Listens to Requests
CRDResponseQueue	Transmit Responses
CRDDataInQueue	For reception of CRD
CRDDataOutQueue	For transmission of CRD

Sample Usage Scenario : Master Source

CRD Agent program waits for message on Request Queue, then starts User Exit

User Exit delivers file in specified directory

Message containing deliverable is sent to either the Central CRD instance, or to the subscriber(s)



Where in the chain exactly the transformation to CIEF is undertaken, is dependent on whether the CRD Repository acts as a Master Source on behalf of the original source. In that case, the transformation can be done at the Central CRD Organisation.

Connect to MQ Queue and receive object message

```
/* connect to the CRDCommandQueue */
do
  myQConnFactory = QueueConnectionFactory
  commandQueue = javax.jms.Queue

  properties_ = Properties()
  properties_.put(Context.INITIAL_CONTEXT_FACTORY,"org.jboss.naming.HttpNamingContextFact
  properties_.put(Context.PROVIDER_URL, "http://localhost:8080/invoker/JNDIFactory")

  parent.ctxRequest = Context InitialContext(properties_)
  myQConnFactory = QueueConnectionFactory parent.ctxRequest.lookup("UIL2ConnectionFactory"
  commandQueue = javax.jms.Queue parent.ctxRequest.lookup("queue/CRDRequestQueue")

con = QueueConnection myQConnFactory.createQueueConnection()
session_ = QueueSession con.createQueueSession(0,Session.AUTO_ACKNOWLEDGE)
receiver_ = QueueReceiver session_.createReceiver(commandQueue)

con.start()
parent.logger_.info("RequestInQueueTask listening on CRDRequestQueue")
loop forever -- in this part the agent is waiting for things to
             -- do and responding accordingly
  objectMessage_ = ObjectMessage receiver_.receive()
  s = CRDObject objectMessage_.getObject()
    parent.logger_.info( s.getClass().getName() )
    parent.logger_.info( "CRDAgent: Received: " s)
```

Send data as BytesMessage

```
loop while i.hasNext()
  f = File i.next()
  parent.logger_.info( "DirectoryScanner: Found file" f )
  parent.logger_.info( "DirectoryScanner: File Size:" f.length() )
  z = Zip(f.toString()'.zip')
  z.add(f.toString())
  parent.logger_.info( "DirectoryScanner: Compressing file ...")
  z.create()
  g = File(f.toString()'.zip')
  parent.logger_.info( "DirectoryScanner: Compressed Size:" g.length() )
  in  = FileInputStream(g)
  parent.logger_.info( "DirectoryScanner: Queueing file ...")
  loop forever
    len = in.read(buf)
    if len = -1 then
      do
          len = 0
          leave
      end
    if len <> 4096 then bytesMessage_.writeBytes(buf, 0, len)
    else bytesMessage_.writeBytes(buf)
  end
  -- put it on a queue
  parent.logger_.info( "DirectoryScanner: Sending file")
sender.send(bytesMessage_)
  /* clean up resources and let the gc collect garbage */
```

Agenda

Portal

Evaluation: Barracuda

Application Framework based on XMLC compiler for strict MVC separation

Pro	Con
Strict MVC	Single Source
Built in locale support	Low takeup
	Complex build environment
	Dependent on ANT
	Could not get to run on W32 (yet)

Evaluation: Struts

Powered by Struts

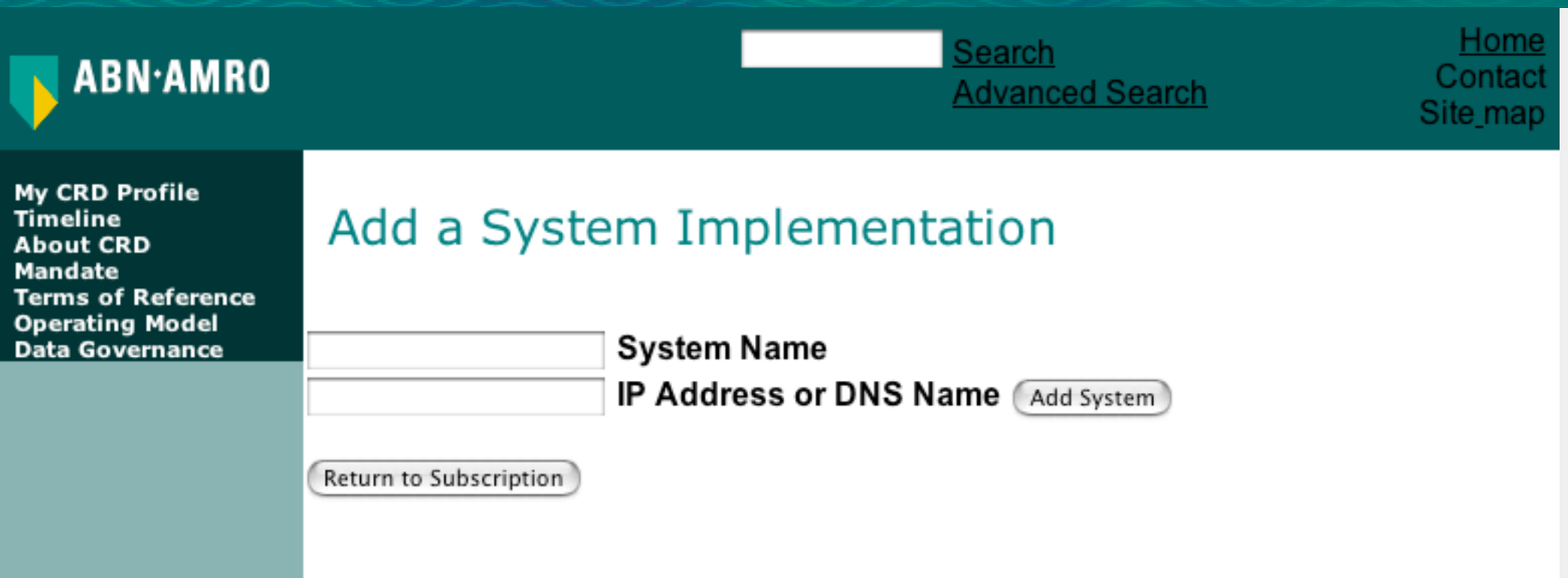
Pro	Con
MVC support	MVC support more lax than Barracuda
‘Universal acceptation’	Single source

Evaluation: JSF

JSF (Java Server Faces) is a development from the makers of Struts and Eclipse to give Java Server programs an MVC, **Event Based** GUI, along the lines of Swing

Pro	Con
MVC support	
Multiple Sources	Now one reference implementation

Easy linking of web forms to NetRexx code




```
<f:view>
<h:form id="AddSystemForm">

  <h:inputText id="PrimaryNameIn" value="#{addSystem.primaryName}" />
<b>  <h:outputText id="PrimaryNameOut" value="System Name"/></b><br>

  <h:inputText id="UriIn" value="#{addSystem.systemAddress}" />
<b>  <h:outputText id="UriOut" value="IP Address or DNS Name"/></b>

  <h:commandButton action="#{addSystem.add}" value="Add System" /><br><br>
  <h:commandButton action="#{addSystem.goback}" value="Return to Subscription" /><b
r>
</h:form>
</f:view>
```

The below JSF fragment binds this web form to the NetRexx code on the next slide

The NetRexx code to handle the business logic

AddSystem.nrx: /Volumes/Workspace/src/com/abnamro/crd/admin/AddSystem.nrx

```
method setSystemAddress(s=String)
  this.systemAddress = s

method getSystemAddress() returns String
  return this.systemAddress

method add() returns String
  transaction = this.session_.beginTransaction()
  uri_ = UniformResourceLocator()
  if this.getSystemAddress() = null then this.setSystemAddress('localhost')
  uri_.setPrimaryName("http://"this.getSystemAddress()":8080/invoker/JNDIFactory
")
    this.session_.save(uri_)
    sip = SystemImplementationService()
    if this.getPrimaryName= null then this.setPrimaryName('dummy name')
    sip.setPrimaryName(this.getPrimaryName())
    sip.setUniformResourceLocator(uri_)
    this.session_.save(sip)
    transaction.commit()
    say CRDutil.getUser() 'added a SystemImplementationService at' this.getSystemA
ddress()
    return "success"

 method goback() returns String
   say CRDutil.getUser "transferred from addSystem back to addSubscription"
   return "goback"
```

--(DOS)-- **AddSystem.nrx** Bot (50,0) SVN-6463 (Netrexx)----------

We have to tell JSF that this NetRexx Bean is managed by it

```
<managed-bean>
  <description>
    Add System Bean
  </description>
  <managed-bean-name>addSystem</managed-bean-name>
  <managed-bean-class>com.abnamro.crd.admin.AddSystem</managed-bean-class>
  <managed-bean-scope>session</managed-bean-scope>
</managed-bean>
```

There is a configuration file named faces.xml that deploys in the J2EE Container WEB-INF application directory.

All other configuration for JSF also goes here.

Very handy: all navigation between pages is in faces.xml

```
<navigation-rule>
  <from-view-id>/AddSystem.jsp</from-view-id>
  <navigation-case>
    <description>
      Any action that returns "goback" on the add System page
      skips to the page that adds a subscription
    </description>
    <from-outcome>goback</from-outcome>
    <to-view-id>/AddSubscription.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
```

The NetRexx method only needs to return a String for the JSF framework to switch upon

You could even design this beforehand ;-)

You can preload the content of dropdown lists in Maps

```
<!-- generated file containing managed classes for CRD project -->
<!-- generated on Tue 04-Jan-2005 23:45:27 -->
<!-- ==================== Initialized Static Schemes ============== -->
<managed-bean>
<managed-bean-name>Scheme</managed-bean-name>
<managed-bean-class>java.util.TreeMap</managed-bean-class>
<managed-bean-scope>application</managed-bean-scope>
<map-entries>
<value-class>java.lang.String</value-class>
  <map-entry>
    <key>Scheduling Status Event Initiation Type</key>
    <value>a38485a8-581c-11d9-80b5-000d9d9bf815</value>
  </map-entry>
  <map-entry>
    <key>Engine Type</key>
    <value>a373e3ce-581c-11d9-80b5-000d9d9bf815</value>
</map-entry>
<map-entry>
  <key>Scheduling Event Life Cycle Status Type</key>
  <value>a3658be7-581c-11d9-80b5-000d9d9bf815</value>
```

This calls the setter for this property in your NetRexx Object, so we can loop through it and fill the dropdown list

Portal

HIBE

Object-Relational Layer

- Use your Objects and have them stored and retrieved automatically
- Do not have to decompose them yourself
- Open Source

HIBERNATE

Architecture

The application can have transient objects (that you do not save), and persistent objects (that are saved in your data base)

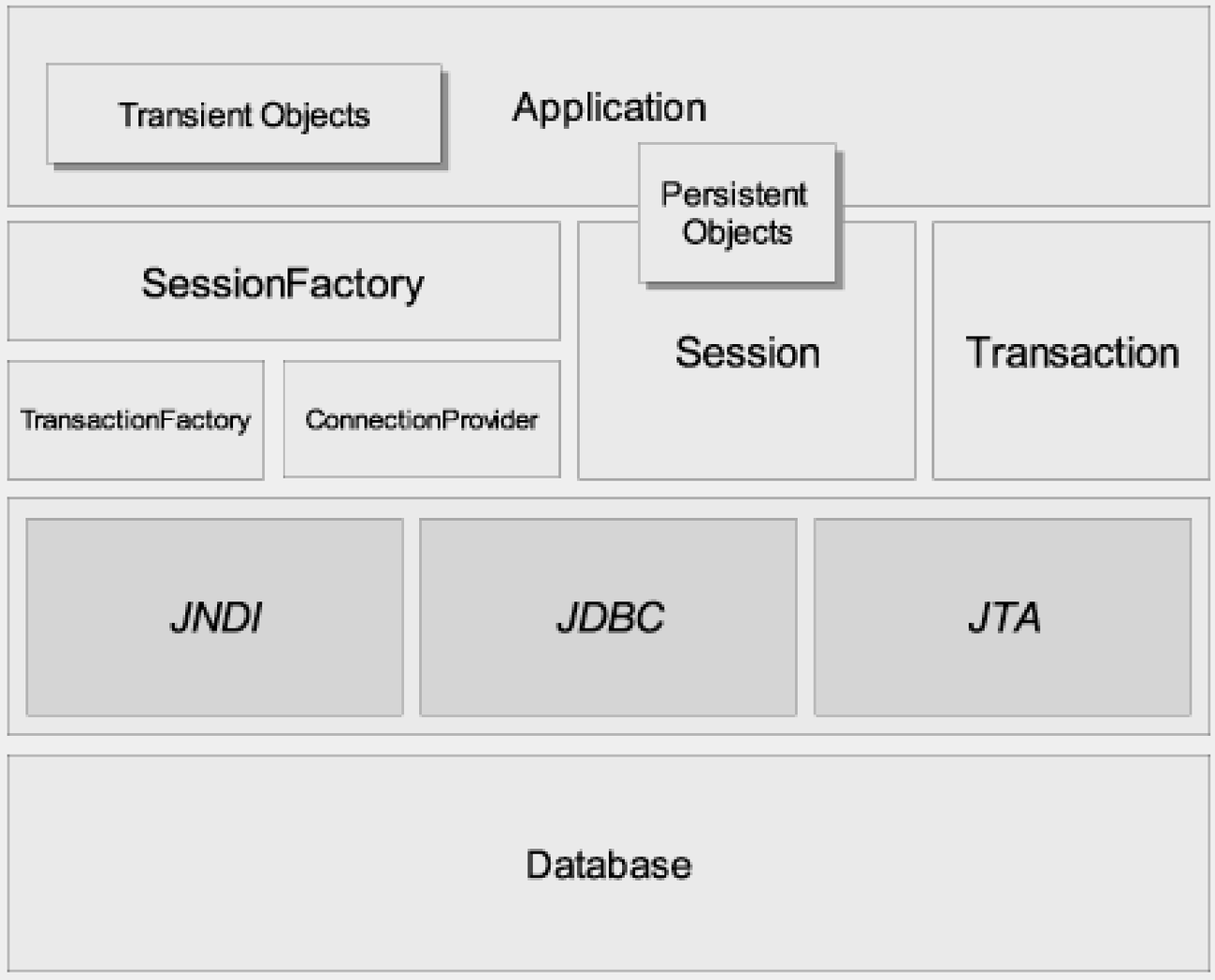
The general form is Session.save(object)

It is possible to have transactions

Hibernate works with nearly all RDBMS engines

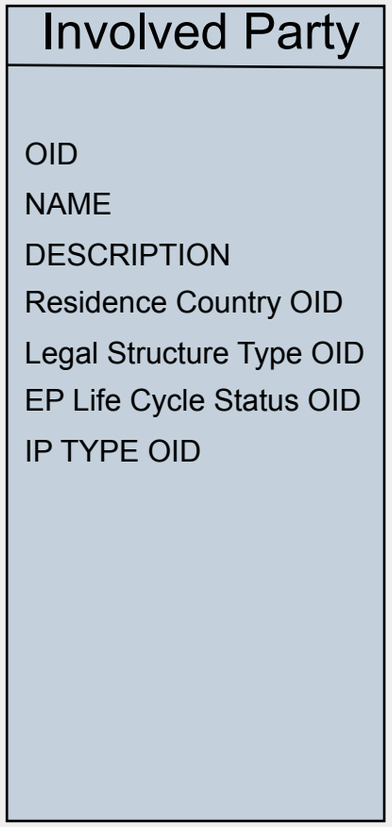
There are several strategies to generate classes from specs, or DDL from classes

We chose to generate *.hbm.xml files, and generate classes and DDL from these

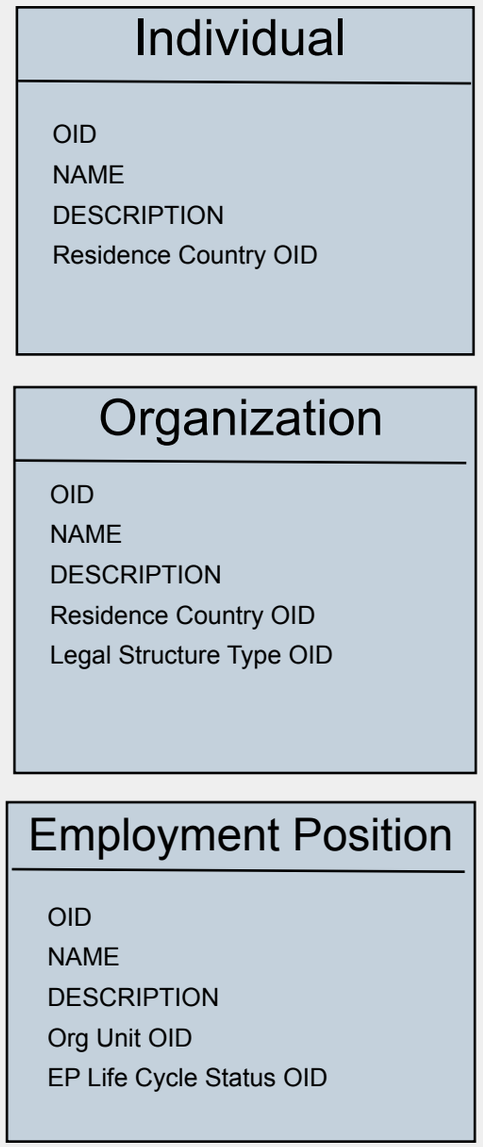


Choose the forward engineering strategy

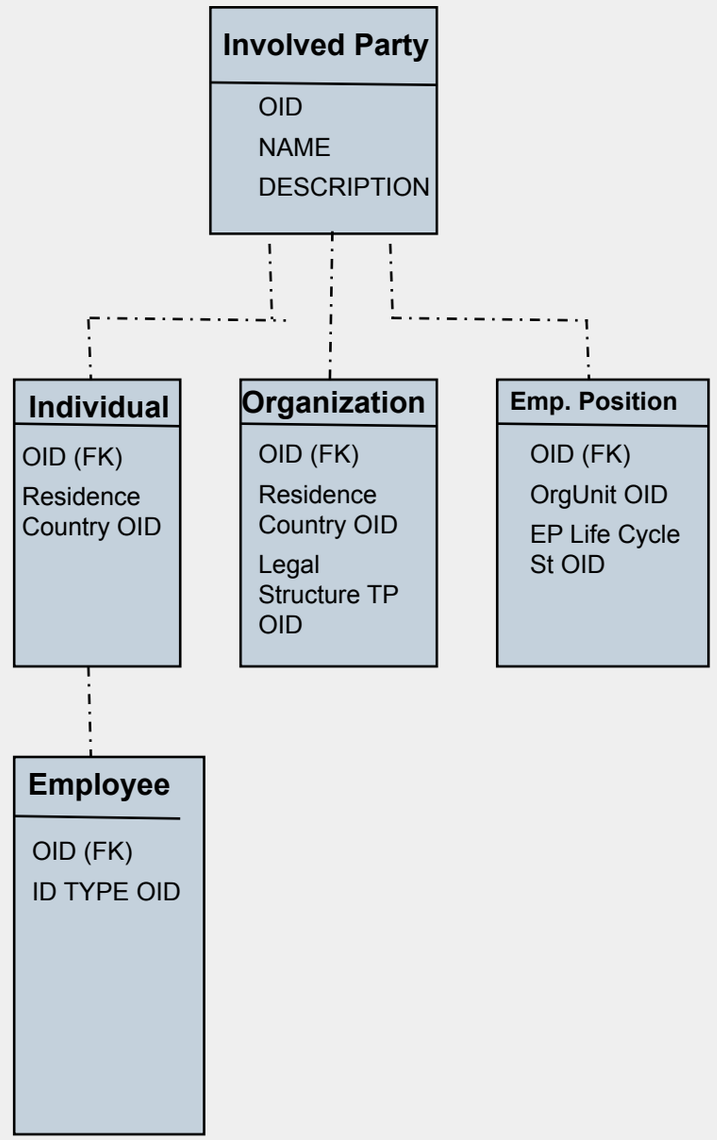
One Table per Hierarchy



One table per concrete class



One table per class



Object Identifiers

47141864-D8E7-11D8-96A5-000393123340

Identity is a property that differentiates an object from all other objects. This term is often confused with addressability, equality of attributes, or a unique name. In relational databases, names or other descriptors are used as keys. Usually, these keys are then also employed to create relationships between objects. However, those descriptors are only partly usable as identifiers, because the identity of an object (for example a car) does not usually depend on its attributes (for example its number plate).

Nothing endures like change, and identifying objects by names in a business environment is generally a bad idea. For this purpose an identifier strategy is called for.

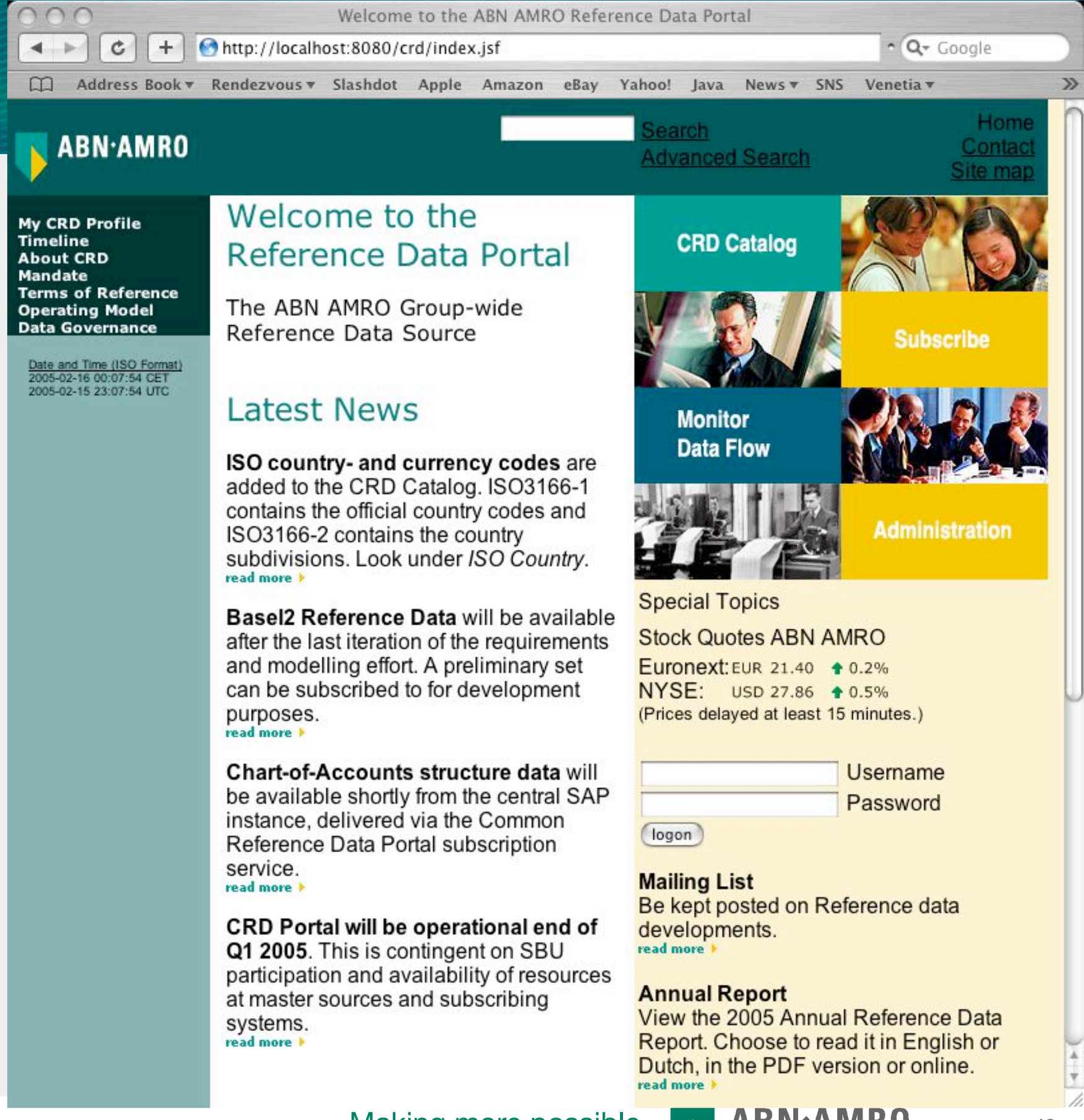
An ID should be unique, at least within a type hierarchy.

Most useful is a strategy that combines a globally unique ID with a low-latency distribution-to-central mechanism.

The ID range should not run out of values.

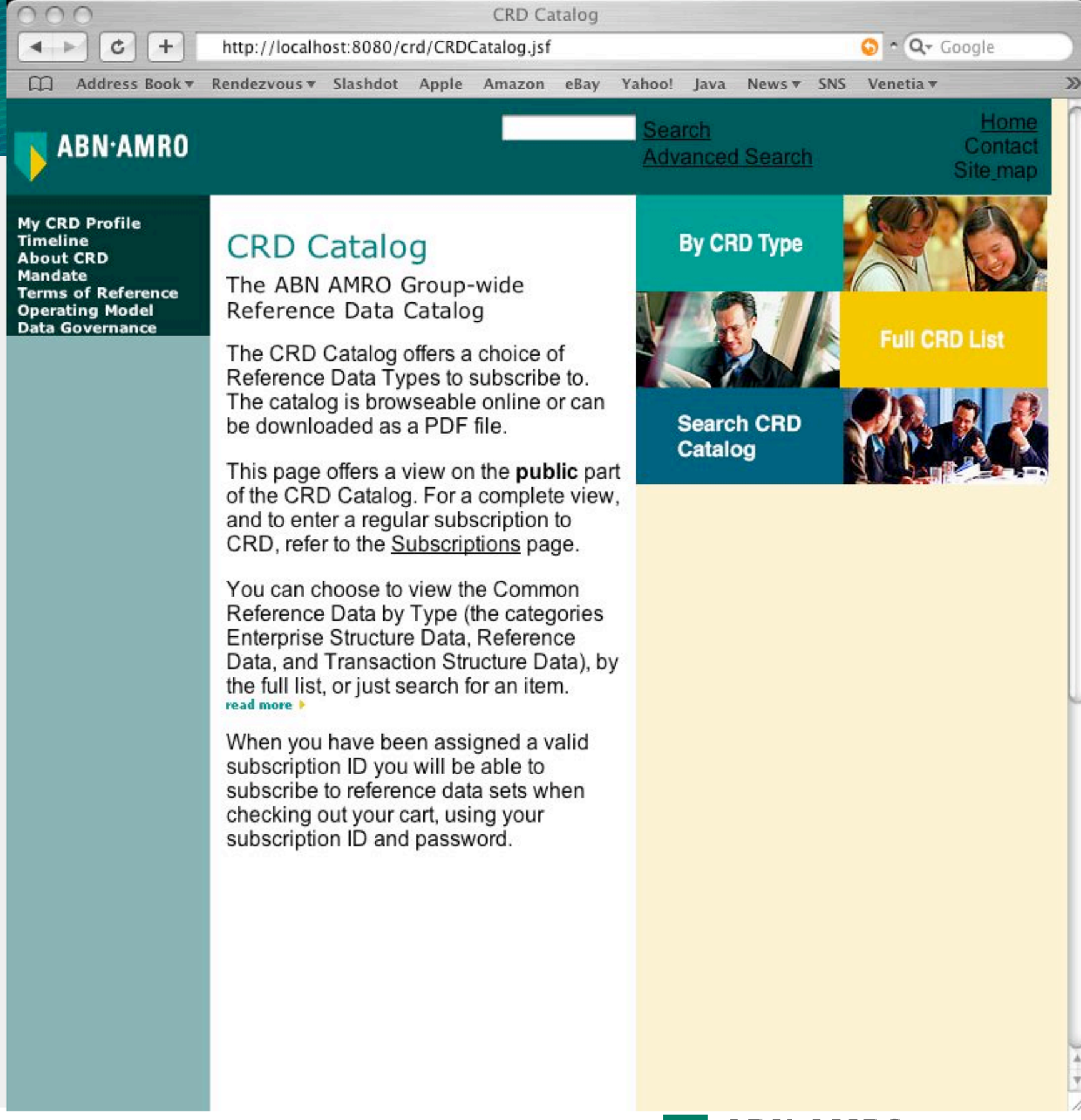
Screenshots

Home Page



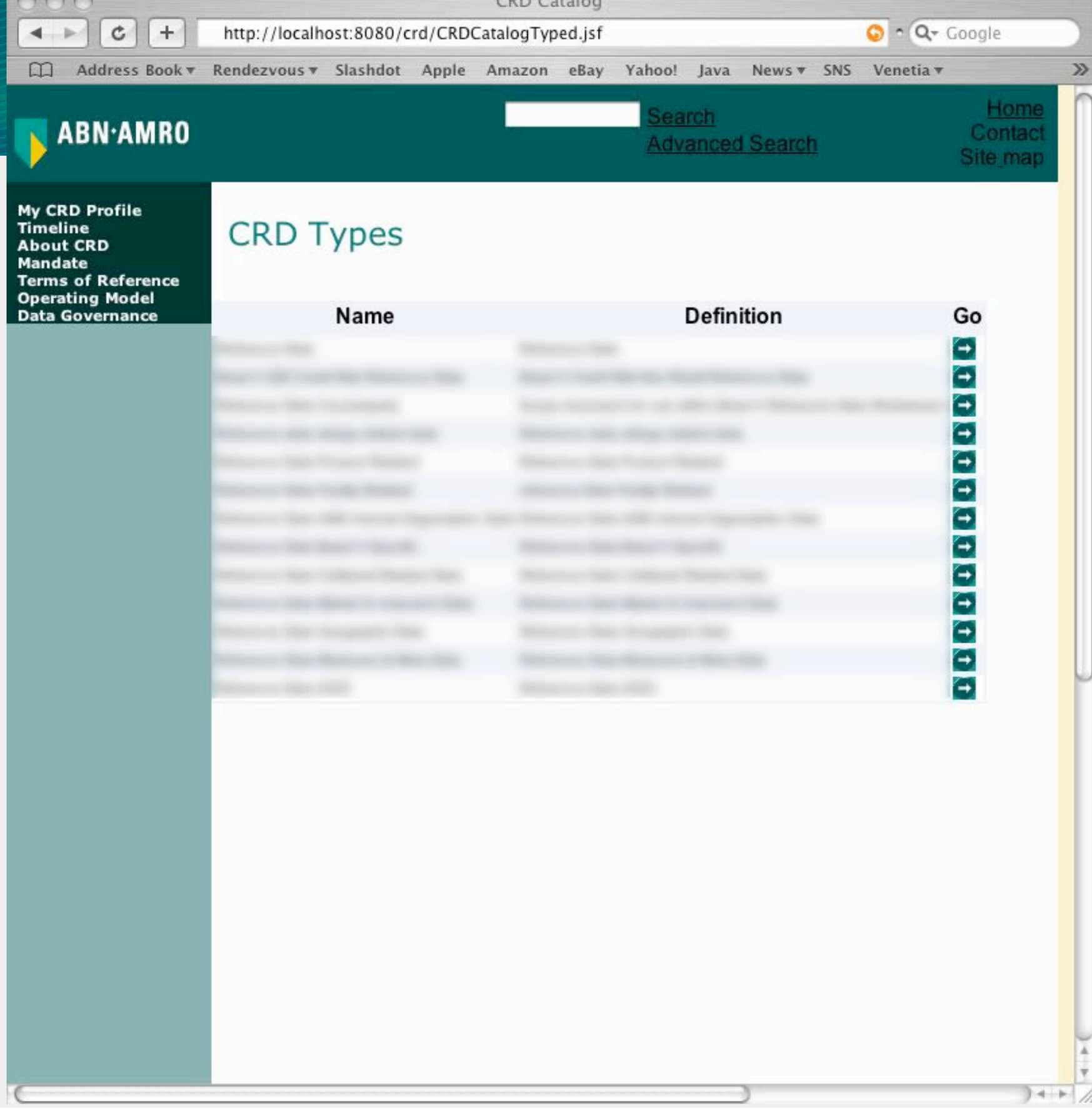
Screenshots

Catalog Menu



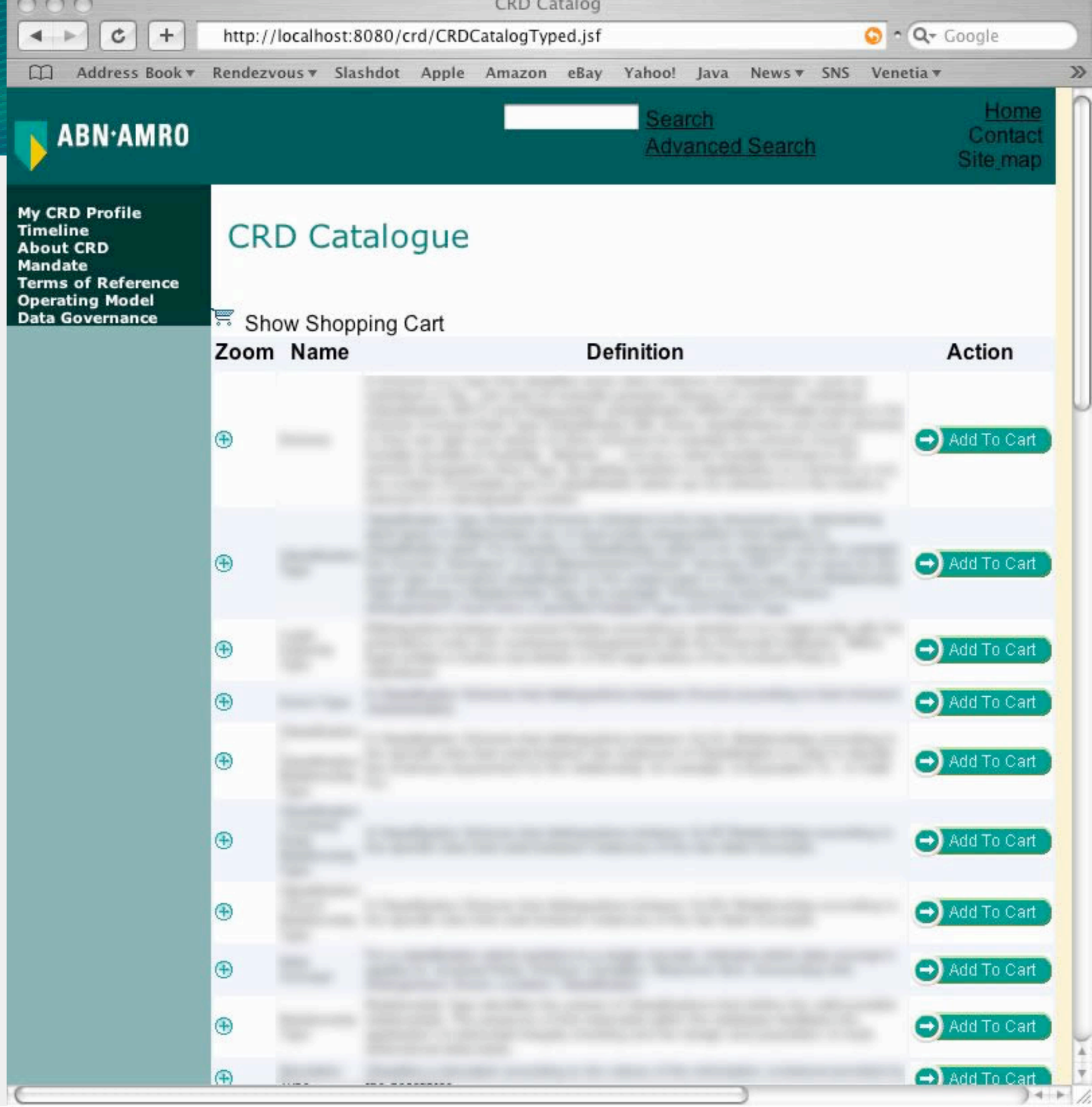
Screenshots

CRD Types



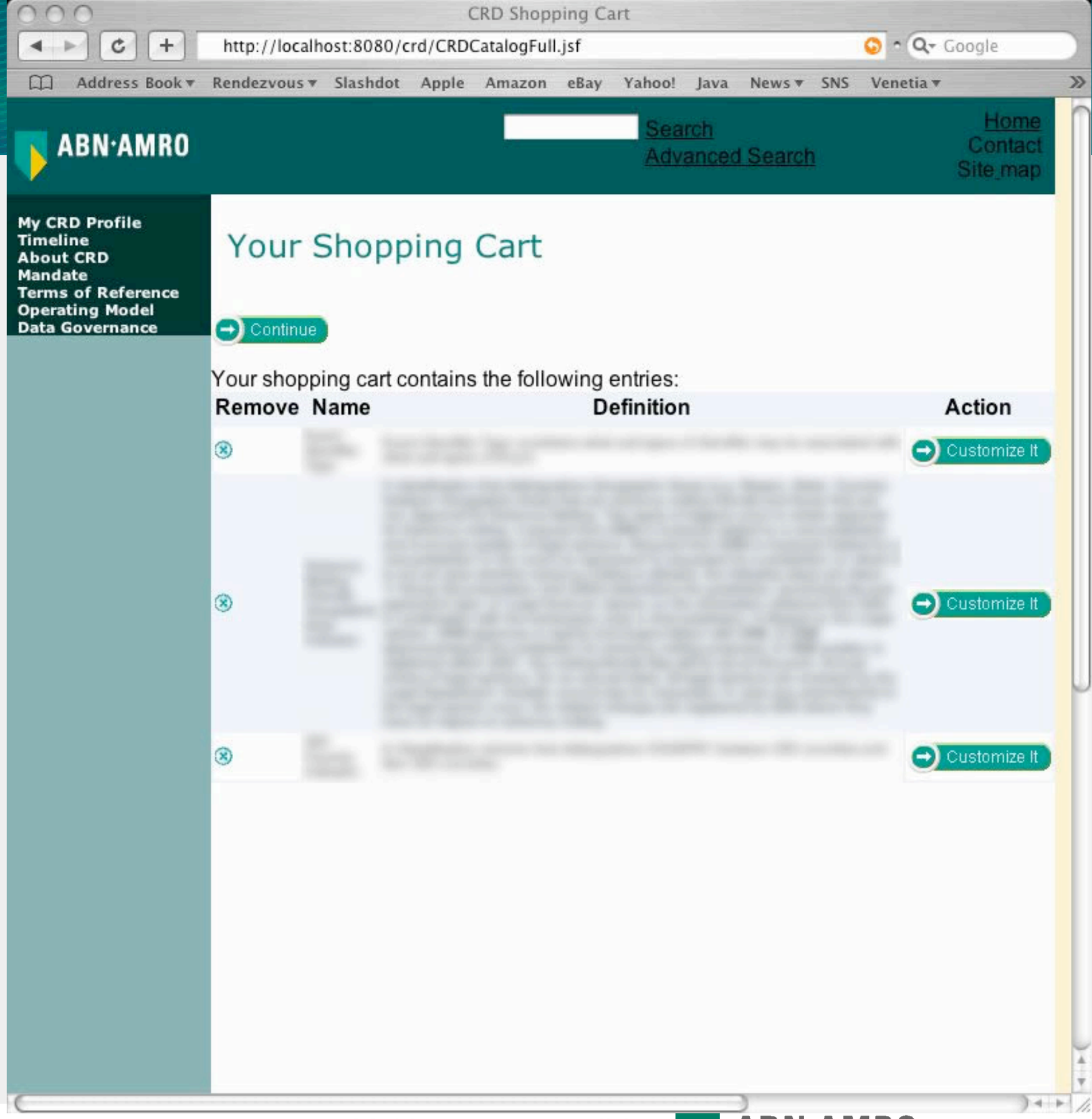
Screenshots

Add to Cart



Screenshots

Your Shopping Cart



Conclusions

- NetRexx easily integrates into the J2EE world
  - Because integration ‘glue’ was always Rexx’s strong point
  - Blurs the distinction between scripting and building a system
- There is a qualitative difference that enables model driven development decisively
-

That’s all for now

Are there any questions?

Thank you very much for your attention.

rvjansen@xs4all.nl

Rene.Vincent.Jansen@nl.abnamro.com