ercatons – XML-based object model

Dr. Falk Langhammer

Living Pages Research GmbH
Munich, Germany
www.living-pages.de

47.Treffen des Arbeitskreises Objekttechnologie Norddeutschland, HAW Hamburg, Germany
March 22, 2004

Living Pages Research

Table of Contents

Living Pages Research

Part I

Philosophical consideration

"according to childs, objects are easy and fun to play with"

The *Ercato* Manifesto

T*he* ***exception*** *is the rule.*

§1 **O***ur world is* ***rich*** *and complex*
rather than well-structured and simple.

§2 **S***oftware must cover* ***irregular***, *changing patterns*
rather than regular patterns.

§3 **A** *software system is an* ***organic*** *being*
rather than a set of mathematical algorithms.

§4 **S***oftware components are an* ***integral part*** *of our rich world*
rather than entities at some meta level.

§5 **S***oftware engineering evolves from small to* ***large***
rather than from concrete to abstract.

Living Pages Research

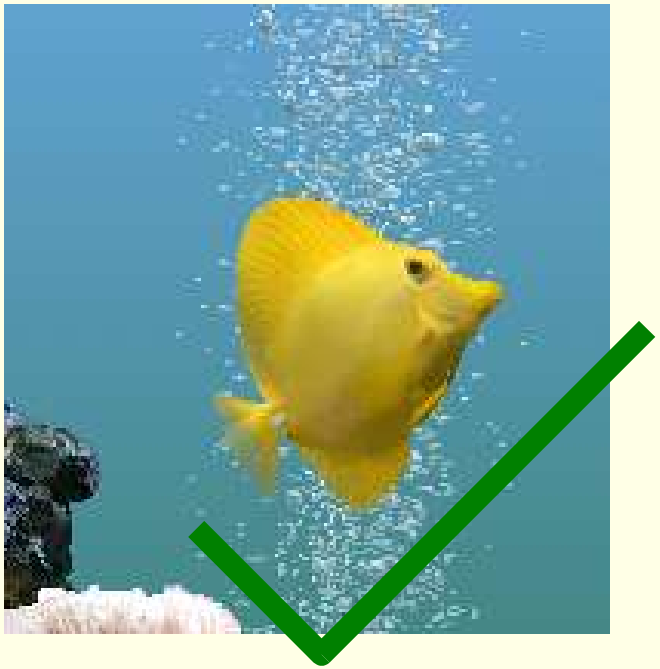
Some direct corollars

- MDA, OOP, UML, J2EE or ERP/CRM/EAI/WF do *not* deliver
  - Things must become much easier
- Directly supporting observations:
  - Frequent failures to scale large software projects
  - Ridiculously exhausted software budgets
  - The increasing interest in the “Agile Manifesto”
- Indirectly supporting observations:
  - How simply large machines/buildings are assembled from many small items in a scalable manner (Eiffel is a language name, guess why...)

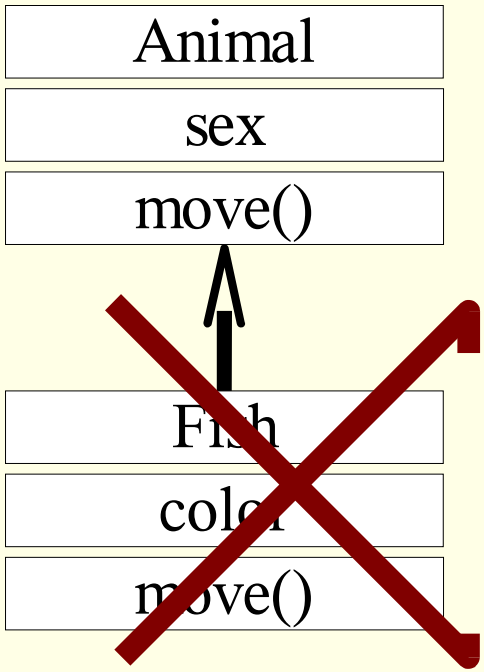
Living Pages Research

Thing-oriented programming

thing



object



- software objects are model-dependent aka *model-driven*.
- models vary
- Call a **rich instance** which needs no model: "**thing**"
- things approximate as far as p. our real-world objects
- rich instances w/o classes cf. "*prototype-based*" ('85)
- things enable the manifesto

Are the both *really* the same?

Why things may be superior to objects

what even screws have what objects don't have...

- screws are self-explanatory and re-usable
- screws can be used without factory plan
- screws have properties which are well-defined (M6) or rather informal (steel)
- screws are used in quantities
- screws can be lost but still continue to exist
- screws can be modified to fit
- screws scale to Eiffel-towers and beyond
- screws perfectly combine with other things

things are more like screws than objects

Part II

Ercatons

"everything is a thing,
by definition"

Every **ercaton** ***is*** a thing

Def.: A ***thing*** is a self-contained entity,
with **identity**, **behavior**, with inner **state** and **structure**,
with user† and model‡ **interfaces**, with **ownership** and
with self-determined **lifecycle** and **privacy**, in both *software* and *reality*.

Def.: An ***ercaton*** is a **thing**, with at least, a model interface to **XML**,
with **inheritance** and **polymorphism**, with a mutable **web** user-interface,
with **database** and **transaction** support and with **autonomous life**.

This means that an ercaton stands up for itself, e.g., it does not depend on a class, that it has a **unique name** and is persistent and protected, and that each ercaton is an individual entity where no two are equal

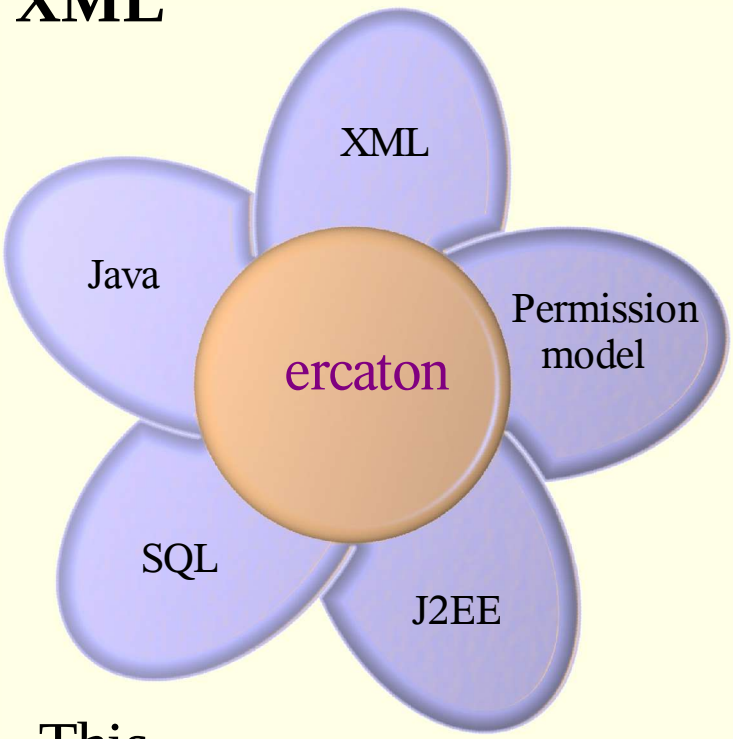
Named after elementary particle convention (electron)

†: we can touch, see and manipulate
‡: we can abstract, in order to think about or to code algorithms

Living Pages Research

Every business object *is* an **ercaton**

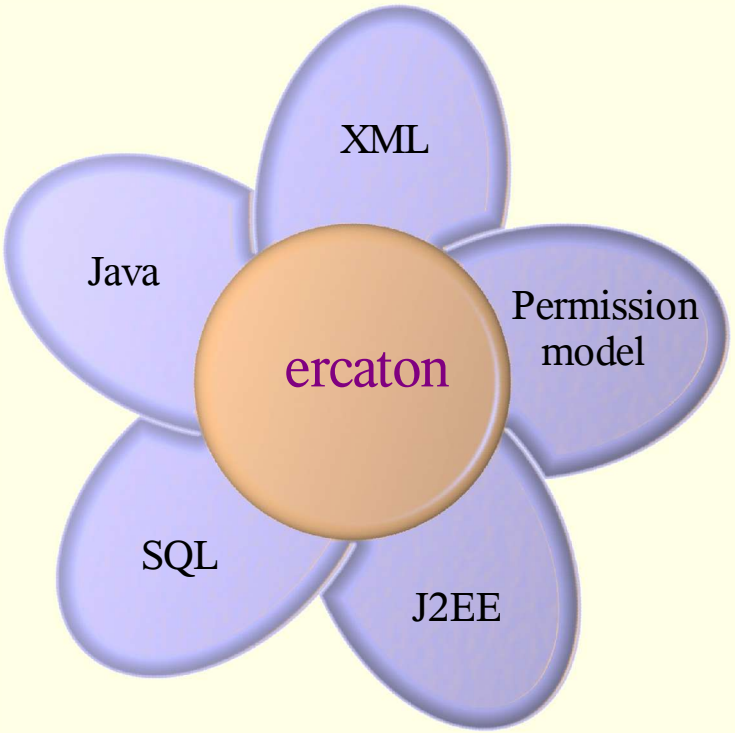
- Ercatons often encapsulate a business object or process. Business knowledge is represented in **XML** and an optional set of languages, incl. **Java**.
- This yields business objects which are fully programmable, are protected by **transactions** and **permissions** and are supported by **indexing** in a database. They are **persistent** by definition, too.
- Their dynamic **inheritance** allows extraction of common parts in the business logic.
- Ercatons express the **business model** in XML. This forms a method, not a framework.



Living Pages Research

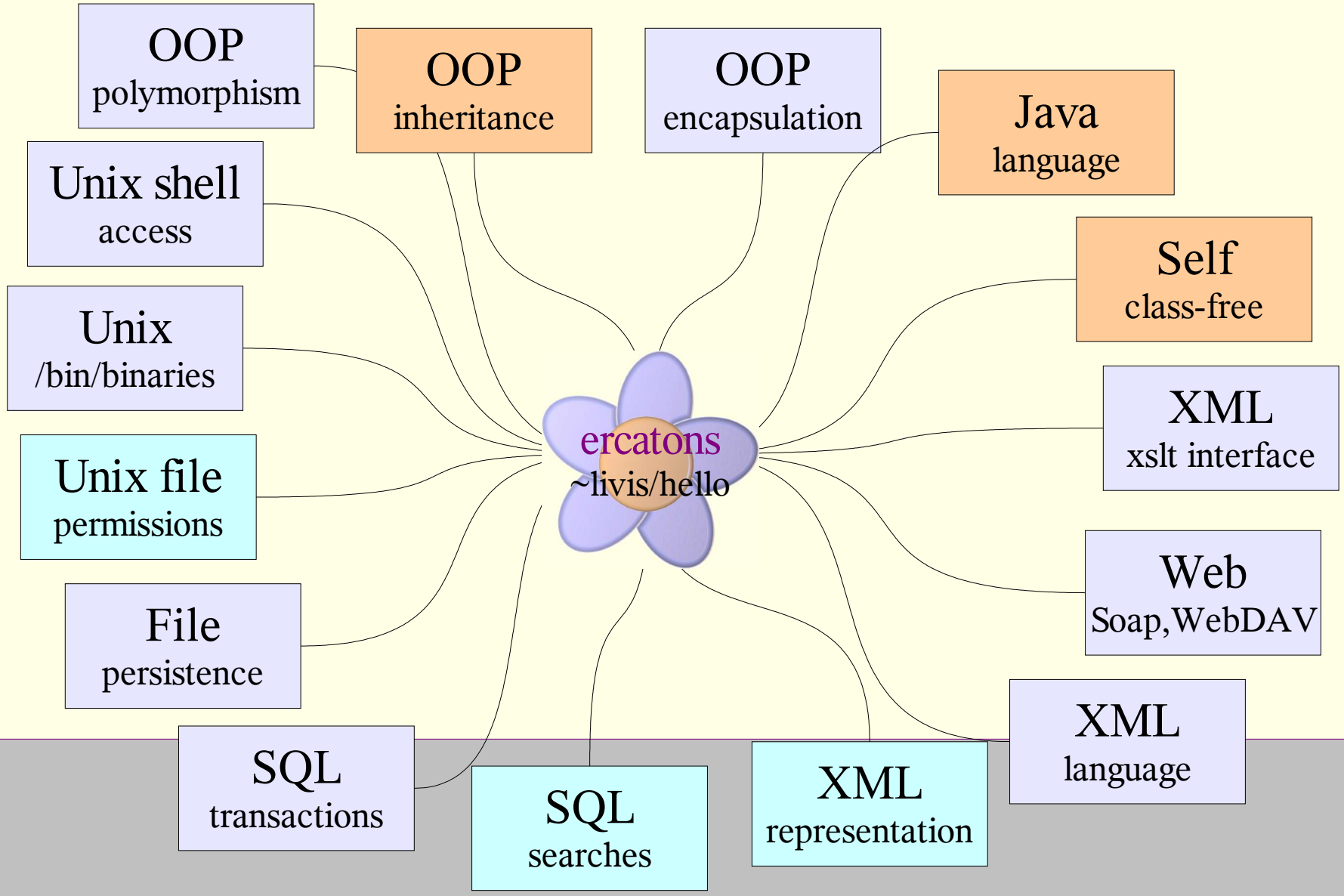
Every **ercaton** ***is*** a document, too

- **User interfaces** may be generated by style sheets depending on an output target pipe
- Non XML resources are represented as "resource" ercatons. This includes JAR files, i.e., **code may change at runtime**.
- ercatons are **versioned**
- ercatons have **owners** and may define a capability chain to **protect** their state
- **SQL**-like queries with inner and outer joins may be used to retrieve data contained in XML!



Living Pages Research

Roots of **ercato** programming



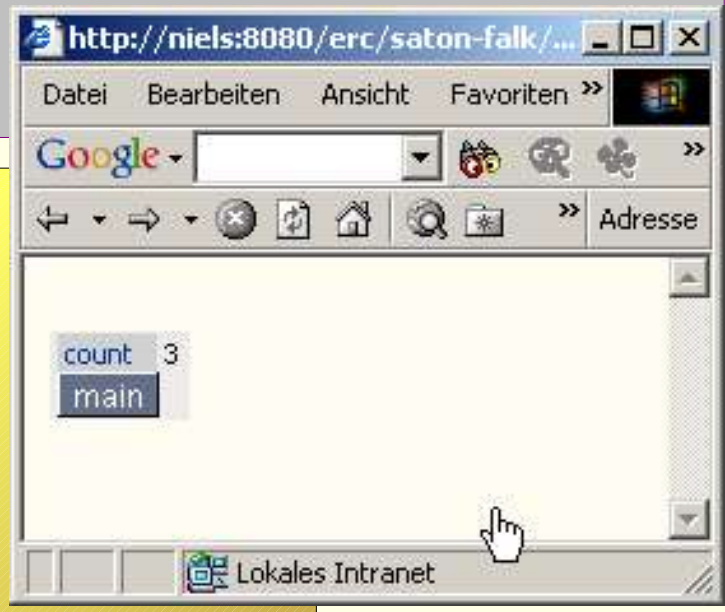
Living Pages Research

Part III

Samples

“a line of code says more than thousand images”

An **ercaton** named ~livis/count




```
<?xml version="1.0" encoding="ISO-8859-1"?>
<counter xmlns:erc="http://ercato.com/xmlns/ErcatoCore">
  <count>0</count>
  <erc:id>~livis/count</erc:id>
  <erc:object lang="Java">
    <erc:class>com.ercato.lib.shared.Counter</erc:class>
    <erc:archive>~livis/lib/shared.jar</erc:archive>
  </erc:object>
  <erc:action name="main"> !increment </erc:action>
  <erc:action name="increment">
    <erc:permission role="~livis/roles/friend">bx</erc:permission>
    <erc:arg name="amount">1</erc:arg>
    <erc:native lang="Java">
      <erc:method>increment</erc:method>
      <erc:parameter name="amount"  type="int"/>
      <erc:returns type="int"/>
    </erc:native>
  </erc:action>
</counter>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<counter xmlns:erc="http://ercato.com/xmlns/ErcatoCore">
  <erc:clone>~livis/count</erc:clone>
  <count>19</count>
  <erc:id>~livis/count2</erc:id>
</counter>
```

Living Pages Research

A more interesting example

Task: Design & implementation of an "**Address Manager**" application

Detail: Should be network-ready and extensible

Budget: ...15 person minutes

here we go...

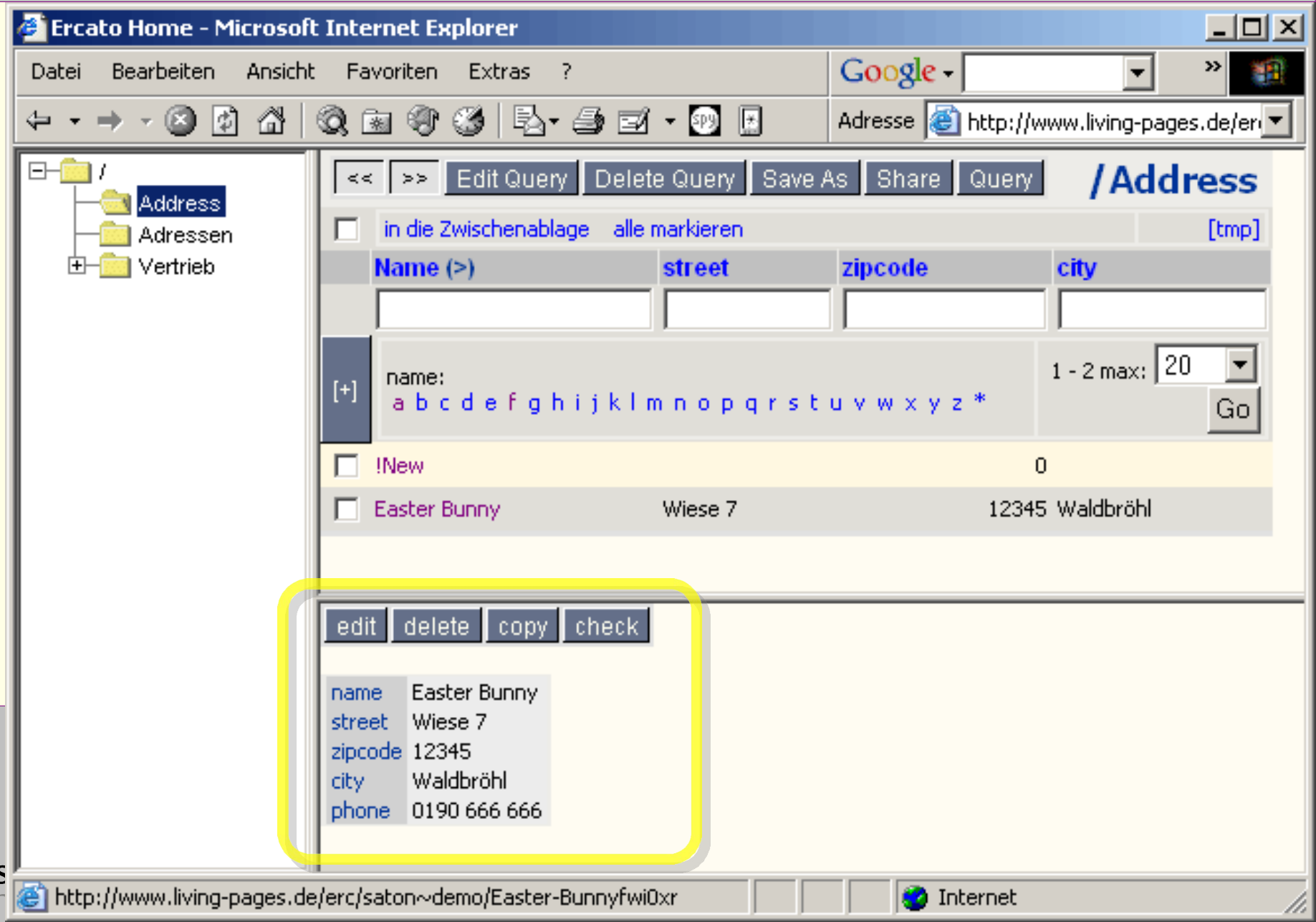
~livis/adr/bunny – That's it

```
<?xml version="1.0" encoding="iso-8859-1"?>
<address xmlns:erc="http://ercato.com/xmlns/ErcatoCore"
         xmlns:erx="http://ercato.com/xmlns/ErcatoExtensions">
      <erc:id>~livis/adr/bunny</erc:id>
      <erc:catalog category="/Address" id-ref="~livis/catalog"/>

      <name    erx:field-ref="string" erc:index="~livis/catalog">Easter Bunny</name>
      <street  erx:field-ref="string" erc:index="~livis/catalog">Wiese 7</street>
      <zipcode erx:field-ref="string" erc:index="~livis/catalog">12345</zipcode>
      <city    erx:field-ref="int"    erc:index="~livis/catalog">Waldbröhl</city>
      <phone   erx:field-ref="string">0190 666 666</phone>

      <erc:action name="edit">   /bin/edit       </erc:action>
      <erc:action name="delete"> /bin/rm$wizard  </erc:action>
      <erc:action name="copy">   /bin/cp$forEdit </erc:action>
      <erc:action name="check">  ~livis/check.xsl
          <erc:arg name="default">Buxtehude</erc:arg>
      </erc:action>
      <erc:trigger name="on-change">$check</erc:trigger>
</address>
```

~livis/adr/bunny and (4) friends



Living Pages Research

~livis/adr/bunny reloaded

```
<?xml version="1.0" encoding="iso-8859-1"?>
<address xmlns:erc="http://ercato.com/xmlns/ErcatoCore">
      <erc:id>~livis/adr/bunny</erc:id>
      <erc:clone>~livis/adr/base</erc:clone>
      <name    >Easter Bunny</name>
      <street  >Wiese 7</street>
      <zipcode >12345</zipcode>
</address>
```

The above is an **equivalent** ercaton using inheritance. It addresses:

- Normalization of data etc. (avoids unwanted duplications etc.)!
- Ercatons provide measures to solve those issues:
  - <erc:clone>
  - <erc:id-ref>
  - erc:expand

~livis/adr/base : class or template?

```
<?xml version="1.0" encoding="iso-8859-1"?>
<address xmlns:erc="http://ercato.com/xmlns/ErcatoCore"
         xmlns:erx="http://ercato.com/xmlns/ErcatoExtensions">
      <erc:id>~livis/adr/base</erc:id>
      <erc:type>prototype</erc:type>
      <erc:catalog category="/Address" id-ref="~livis/catalog"/>

      <name    erx:field-ref="string" erc:index="~livis/catalog"/>
      <street  erx:field-ref="string" erc:index="~livis/catalog"/>
      <zipcode erx:field-ref="string" erc:index="~livis/catalog"/>
      <city    erx:field-ref="int"    erc:index="~livis/catalog"/>
      <phone   erx:field-ref="string"/>

      <erc:action name="edit">   /bin/edit       </erc:action>
      <erc:action name="delete"> /bin/rm$wizard  </erc:action>
      <erc:action name="copy">   /bin/cp$forEdit </erc:action>
      <erc:action name="check">  ~livis/check.xsl
          <erc:arg name="default">Buxtehude</erc:arg>
      </erc:action>
      <erc:trigger name="on-change">$check</erc:trigger>
</address>
```

Model interface: an algorithm in Java

- Ercaton actions are language-independent
- E.g., the ***~livis/count!increment*** action is implemented here:

```
// this is an ercaton's action implemented in Java
package com.ercato.lib.shared;
import com.ercato.core.*;
import org.w3c.dom.*;

public class Counter extends ErcatonObject implements Action {
  Text counter;
  protected void evaluateElement (EvaluationContext ec, String tag, String nsuri) {
    if ("count".equals (tag)) counter = ec.getTextNode (false);
  }
  public int increment (int amount) {
    int count = Integer.parseInt (counter.getData ());
    counter.setData (String.valueOf (count += amount));
    touch ();
    return count;
  }
}
```

Living Pages Research

Part IV

ercatoJ: an ercato engine

“all that glistens is not gold”

Living Pages Research

ercatoJ engine summary

In the first place, ercatons only need a certain amount of '<'-characters to exist.

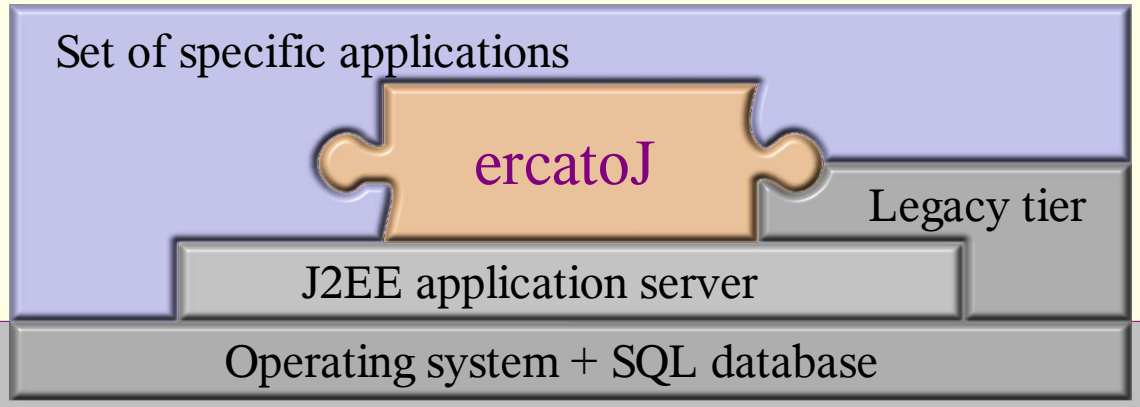
For the more subtle features, a **virtual environment** is needed.

Our only such environment is **ercatoJ**, the **J2EE**-based implementation.

Product v1.0 is in mission-critical use. All basic ideas implemented.

ercatoJ is a set of **EJB's** which need **no redeployment** when ercatons change. These EJB's integrate well into an existing J2EE application.

Every ercaton introduced
absorbs from a J2EE application's complexity.



Living Pages Research

ercatoJ engine properties

(1) Ercatons are mapped onto **Enterprise JavaBeans** (EJBs).

(2) Powerful **algebra for XML** trees implements inheritance.

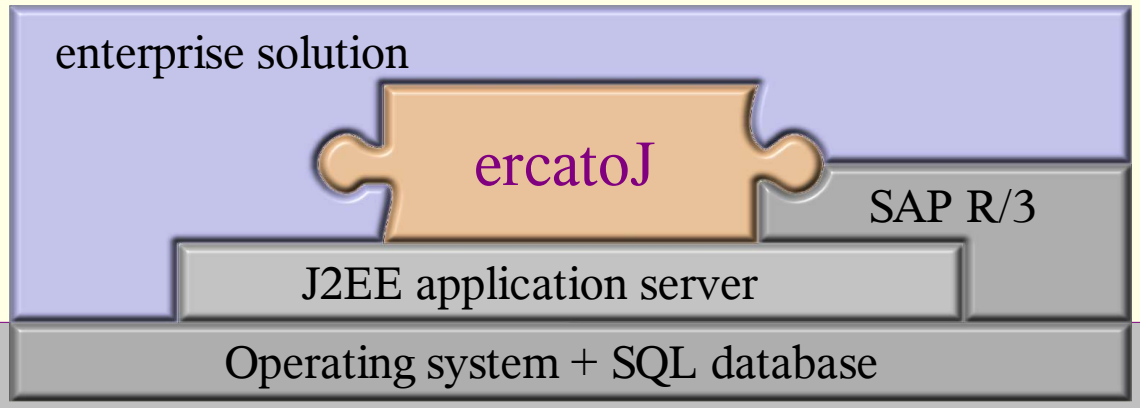
(3) Behaviour of ercatons expressed in both Java and/or XSLT.

(4) WebServices and plain XML exchange available, e.g. for SAP/R3.

(5) User interface is web-based (or via shell), not (yet) Swing etc.

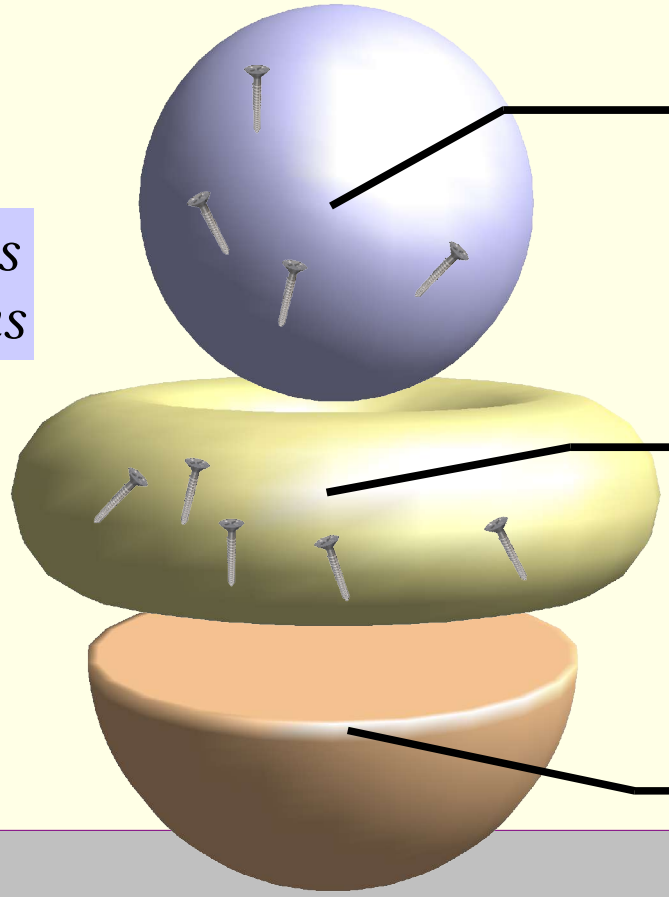
(6) Everything may be an ercaton, incl. images and binary code.

(7) Complex database schemes are generated and kept synchronized.



Living Pages Research

ercatoJ engine hierarchy



business ercatons

service ercatons

EJBs

business processes:
customers
recipes
status propagation, workflows
trigger external systems
check business validity
etc.

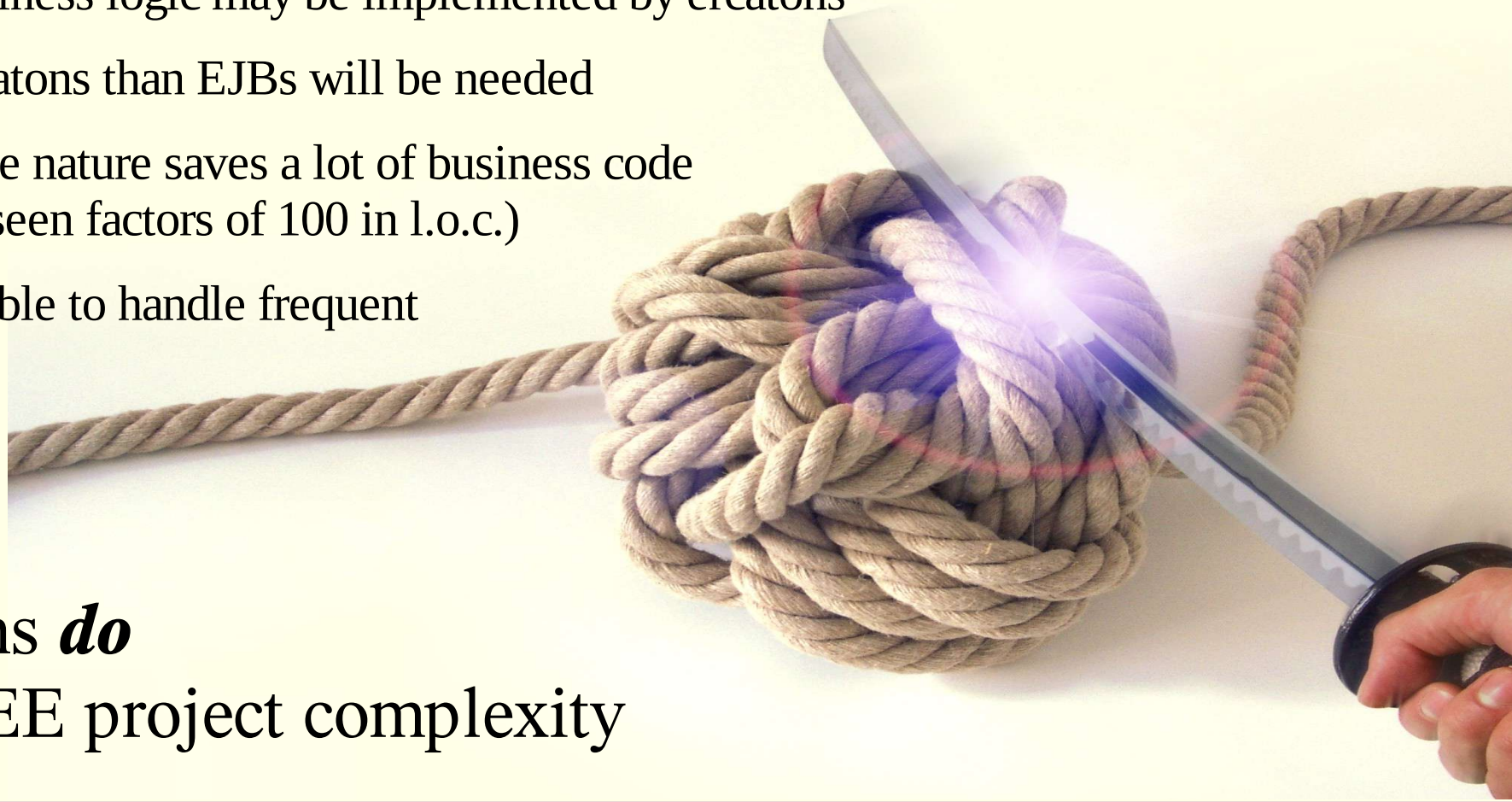
ercatoX standard extensions: (erx)
user interface management
standard services (cp, check, edit, etc.)
administration tools (backup etc.)
etc.

ercatoJ engine: (erc)
ercatons
permissions, transactions
actions, clones, versions
xml interchange
http (browser) access
etc.

Living Pages Research

ercatoJ v1.0 to overcome limitations of J2EE

- Entire business logic may be implemented by ercatons
- Fewer ercatons than EJBs will be needed
- Declarative nature saves a lot of business code (we have seen factors of 100 in l.o.c.)
- Very suitable to handle frequent changes



ercatons ***do***
cut J2EE project complexity

Living Pages Research

when in J2EE project, did you hear, too that...

...development takes 42 percent longer than the worst estimate?

...progress has slowed down?

...hours of delay between coding and testing?

...builds are a nightmare?

...multiday transition to live systems?

...your architecture becomes obfuscated?

...business logic moves into JSPs?

...yes? and it is all true!

E.g., if You follow the J2EE blueprint approach:

- → with one Enterprise JavaBean per Business Object (EJB/BO)
- \+ maybe, using CMP for persistence
- \+ maybe, using JSP for the view
- \+ having more than 30 business objects (or database tables)
- = *You're going to fail really fast.*

J2EE has some intrinsic properties which prevent J2EE projects from scaling or using the “Agile method”. *However*:

Deployed J2EE software is robust, transactional, scalable and uses standardized middleware: **J2EE is a must**!

Other considerations to overcome limitations of J2EE

MDA, UML2.0 and generators:

- Not mature and candidate for yet another CASE hype
- Requires too complex models (too expensive)
- Not extensible/open enough (Skynamics, JDragon, Phaidros)
- Does not solve the real problem of too complex EARs

Frameworks:

- Most of them are not transactionally safe
- Limited scope (Struts, Cocoon, Castor, Turbine, etc.)

XML may help, but:

- XML middleware is not standardized and/or not transactional
- XML retrieval is slower than anyone would normally guess

Part V

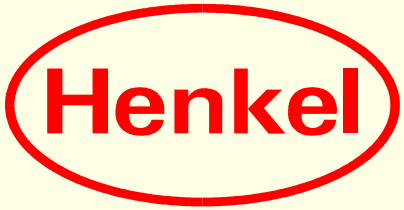
References

“try and success”

Selected ercato community members

- Clavis berater sozietät; HB, BN. (ercato project)
- Docutec AG; AC. (ercatoLT research project)
- Ekato GmbH; LÖ. (ercato project pending)
- Foxray AG; HH. (ercatoLT)
- Ganmi consulting group; E, M. (ercato consulting)
- GI e.V. (organic computing initiative)
- Henkel KGaA; D, KR. (ercato in mission-critical use)
- Hypovereinsbank; M. (J2EE)
- Living Pages Research GmbH; M (ercatoJ); owned by founders, profitable, start-up.
- Lufthansa Revenue Systems; HH. (ercatoLT study)
- Co-created JavaDays, JIT and Net.ObjectDays ('98-'04). (ercato dissimination)
- Sektor GmbH; D. (ercato project)

Selected ercato project reference

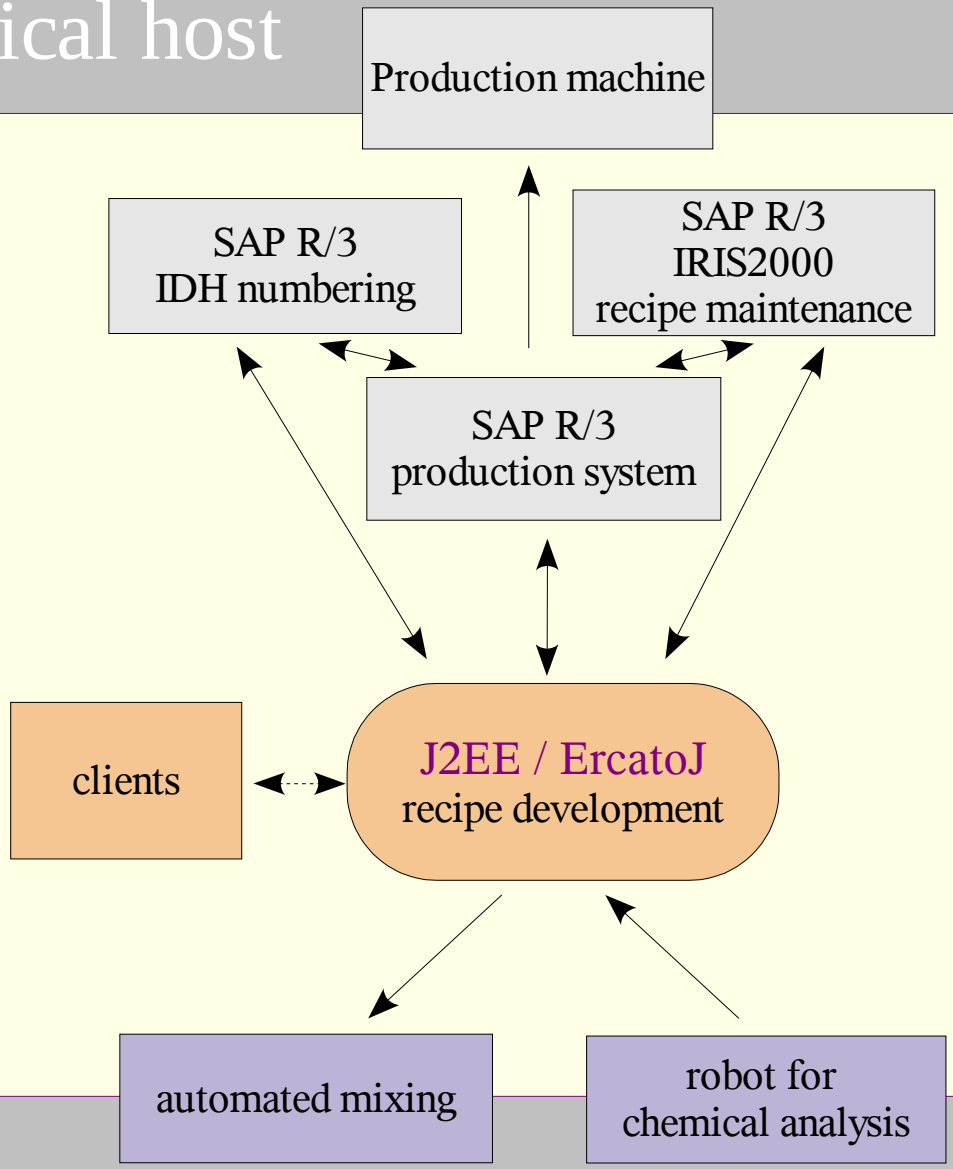
The  project

"even hosts are mortal"

Living Pages Research

The Henkel project:
Substitute mission-critical host

- Substitute mission-critical **host** by an **EJB**-based system
- Migrate existing base of corporate data
- Weekend switch from old to new
- Interface
  - to SAP production systems
  - to specialized machines for mixing and chemical analysis
- at Henkel KGaA, Düsseldorf



Living Pages Research

The Henkel challenge:
available, secure, complex, creative

Availability

- 99.99% at working hours
- Assured synchronization with SAP/R3 systems

Security

- Recipes are *the* core business asset
- Meet legal requirements for perfume recipes (perfume secret, laws)

Complexity

- ~1 million lines of code to be replaced, in short time and fixed budget
- Business processes involve *all* departments at Henkel Fragrance Center
- Configurable data mining

Creativity

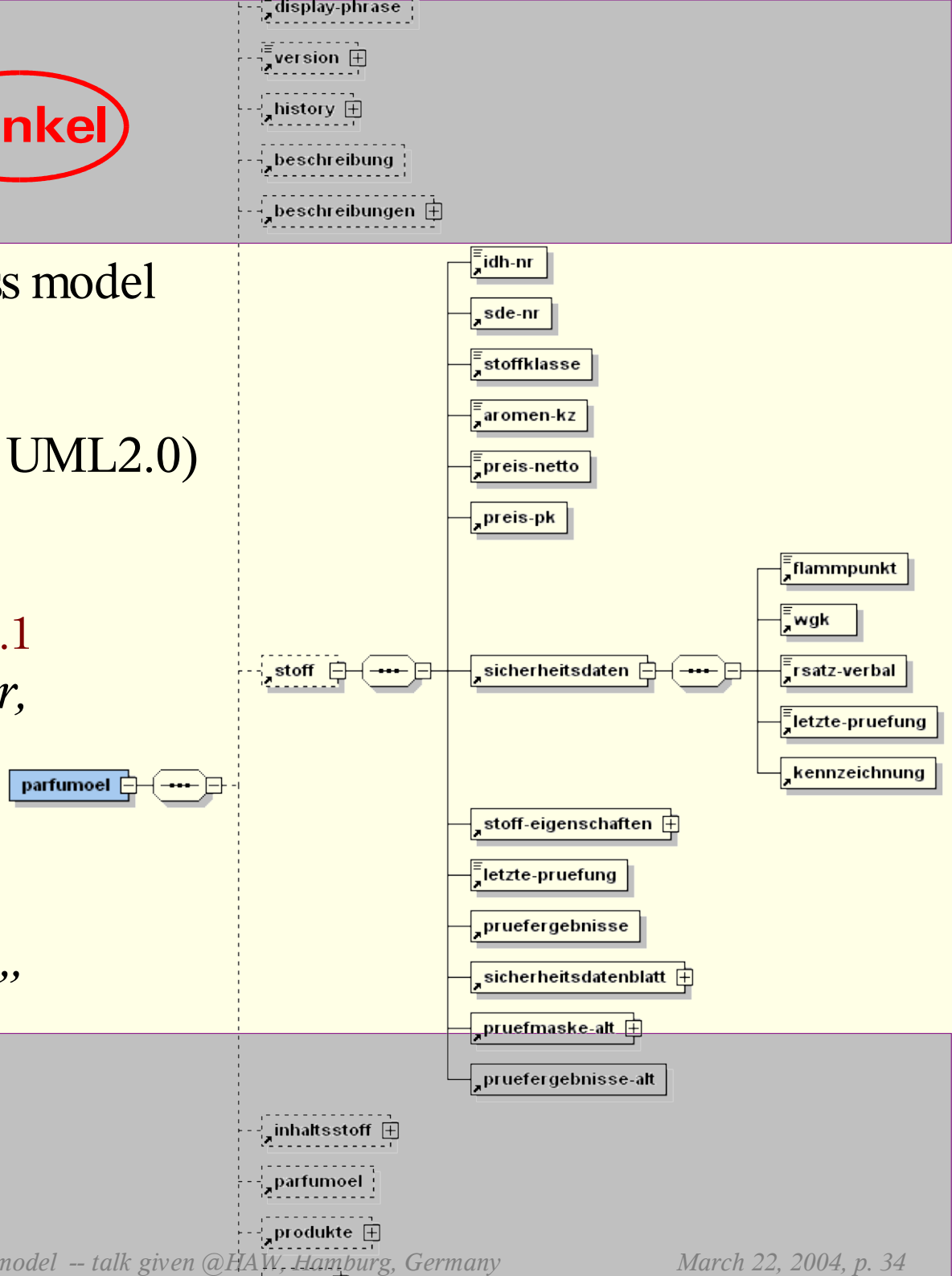
- Perfumers need *very* ergonomic tools to be productive

Model detail Henkel

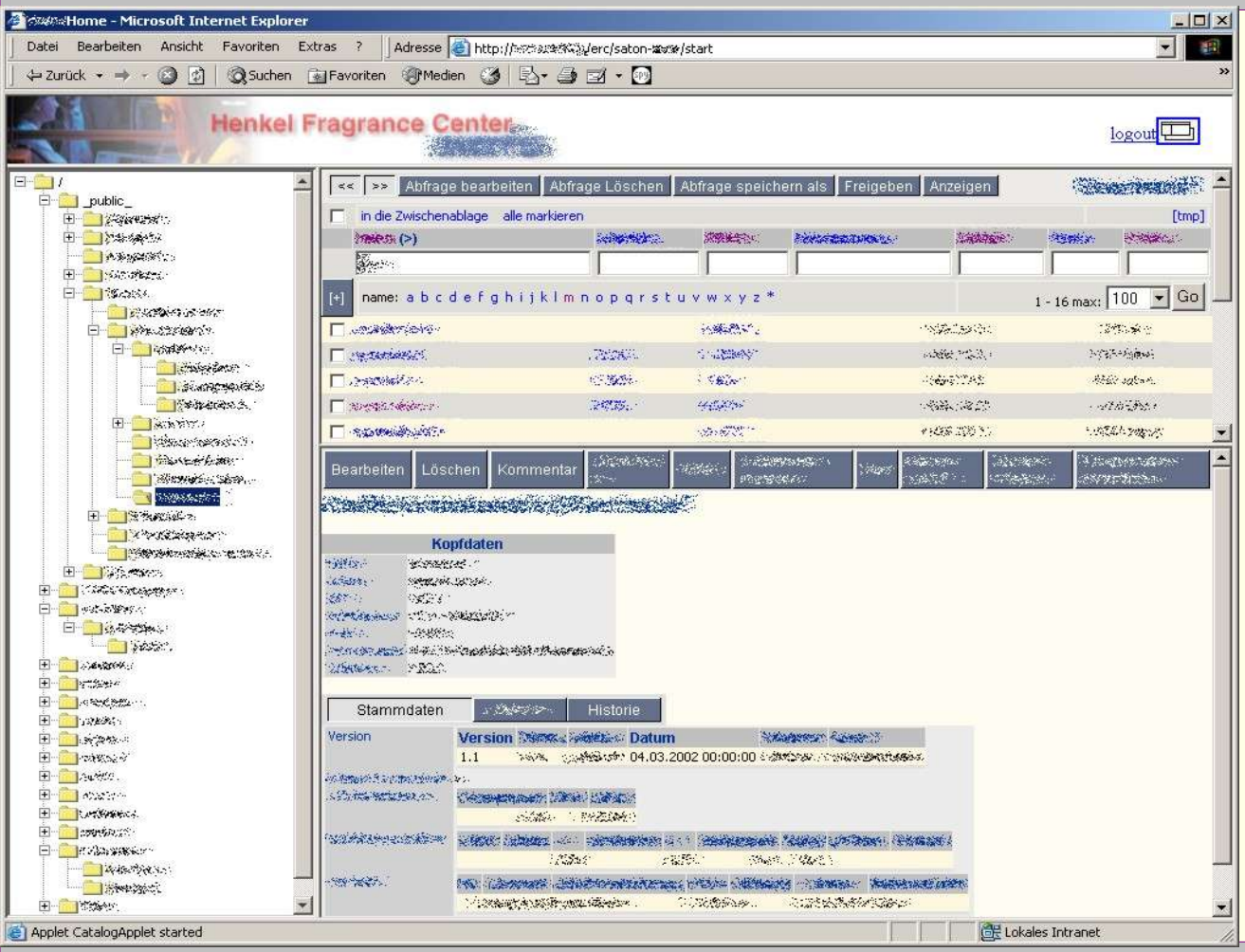
This is an excerpt from the business model

It is drawn as an XML schema
(sometimes more appropriate than UML2.0)

What you see, is, e.g., ercaton
~naris/subs/perfoils/lavendel-6,v2.1
from the live production server,
schema extracted &
visualized by XmlSpy.
...
because
*"implementation **is** the model"*



Henkel The deployed solution...



(screenshot: courtesy of Henkel Fragrance Center GmbH)

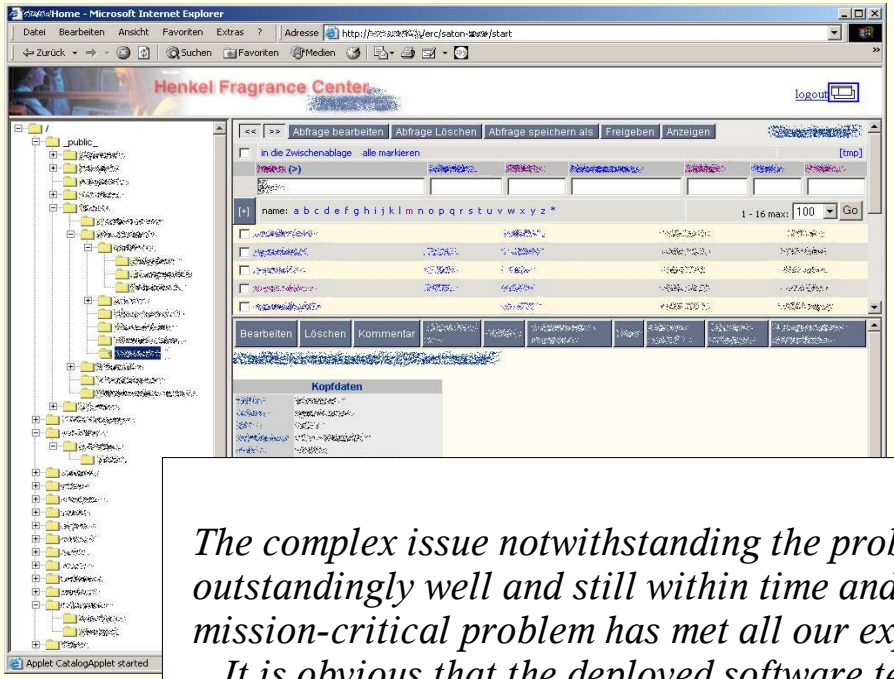
- **Browser based intranet solution**
- **J2EE server application**

Features

- Document-centric (versioned)
- Efficient configurable data mining
- Generated user interface
- "Better than html" controls
- Comprehensible implementation of workflow

Living Pages Research

...satisfies ~*livis/customers/henkel*



- Using innovative J2EE architecture
- Was 1 million lines in host-based system
- One calendar year for development
- Went productive on schedule
- 10k+ lines of XML plus 10k+ lines of Java now make up the business logic

The complex issue notwithstanding the problem has been solved outstandingly well and still within time and budget. The solution of our mission-critical problem has met all our expectations.

It is obvious that the deployed software technology is well suited to address problems of a more complex nature, too.

Dr. Alexander Boeck
Geschäftsführer [Managing Director]
Henkel Fragrance Center GmbH

Henkel

Living Pages Research

Conclusion

The great idea behind object-oriented programming never got implemented.

OO languages actually describe algorithms, not objects.

Some“thing” really close to the initial idea now exists: ercatons.

Ercatons allow to *build* rather than *model* and once you're done, you got an implementation, a model, time left and spare money...

Can rescue J2EE projects in trouble.

It works.

Ercatons were easy to use and breathtakingly efficient. Once you get the idea you wonder how you ever worked without it.

Dr. Ralf Marsula
Senior Consultant
Clavis berater sozietät GmbH, Bremen

Living Pages Research


“Ercatons boldly go where no EJB has gone before”