# An Eclipse Toolchain for Rapid Java Development Using Transparent Persistence

**by Christian Ernst and Markus Kuppe,**
**Engineers at Versant GmbH**

VERSANT

## Agenda

- Introduction
- Persistence
- Java Data Objects (JDO)
- Toolchain
- Demo
- In the future
- Questions and answers (QA)

## Company Introduction

- Versant and the Versant Object Database (VOD)
  - Founded: 1988
  - In 2004 merged with Poet (FastObjects)
  - Offices in Redwood City (USA), Hamburg, and India (Pune)
    - Hamburg: Center for R&D
  - Current release 7.0.1.3
    - APIs available for C, C++, .NET, Java (and originally even Smalltalk)

## Personal Introduction

- Christian Ernst
  - Graduated in Software Engineering 2002, FH Hamburg
  - Started working with Java persistence in 1999
  - For Diploma thesis implemented JDOQL
  - Since 2005 at Versant R&D in Hamburg
  - Official JDO expert for Versant since 2006
- Markus Kuppe
  - B.Sc. in CS from Hamburg University of Applied Sciences
  - With Versant since 2006
  - Started working with Eclipse right from the start in 2001
    - Committer on SoC, ECF and former Dali JPA

## Persistence Basics

- Transparent persistence
- Persistence by reachability
- Lazy-Loading
- Automatic change tracking
- Uniquing

## Java Data Objects (JDO)

- JDO API based on Interfaces
  - PersistenceManagerFactory, PersistenceManager, PersistenceCapable, Query, etc.
- JDO Datamodel aligned with Java datamodel
- JDO Object Lifecycle
- JDO Metadata
- JDO Binary Contract
  - Java Bytecode Enhancement

## JDO Metadata

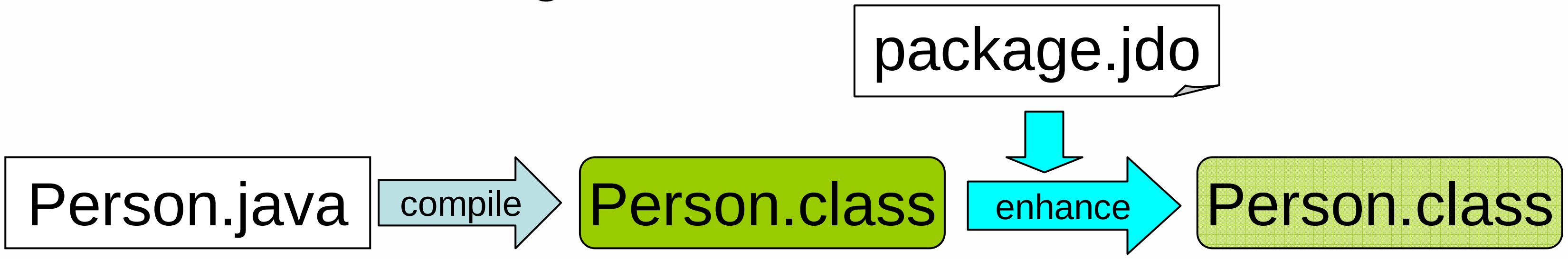
- Uses XML
- Defines persistent Classes, Fields, etc.
- Detailed load and deploy process

```
...
<jdo>
    <package name="model">
        <class name="Address"/>
        <class name="Person"/>
        <class name="Employee" persistence-capable-superclass="model.Person"/>
        <class name="Company">
            <field name="employees" embedded="true" persistence-modifier="persistent">
                <collection element-type="model.Employee"/>
            </field>
        </class>
…
```

package.jdo

## JDO bytecode enhancement

- Adds *PersistenceCapable* Interface
- Adds Field Access instrumentation
- Adds State Management, etc.



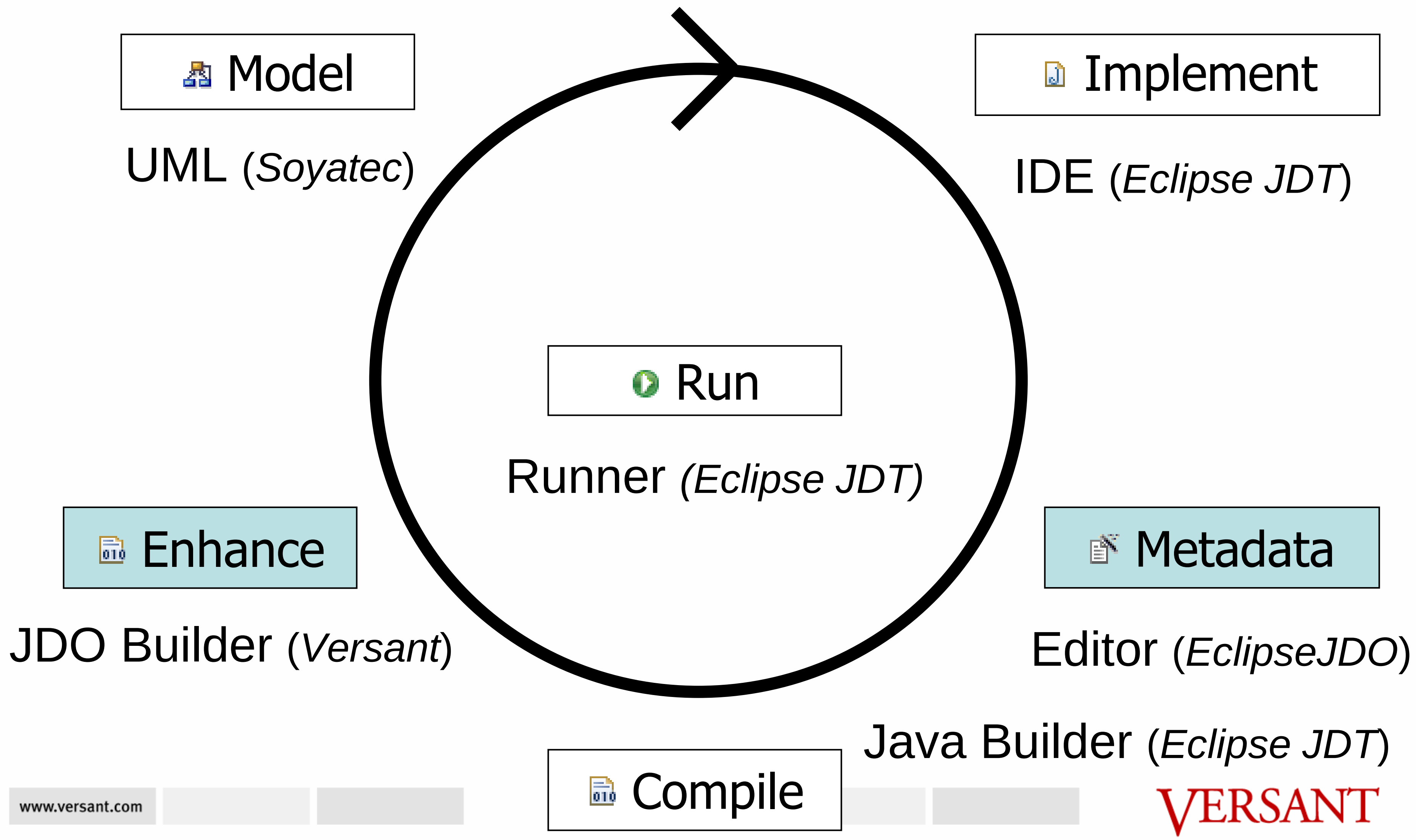
```
public class Person implements PersistenceCapable {
    protected String name;

    public String getName() {
        return jdoGetname(this);
    }
     protected static String jdoGetname(Person x)  {
    …..
```

## Applying transparent persistence to the development environment leads to transparent development?

## What does transparency mean?

- Seamless integration into the (Java) development environment (IDE)
  - Just one tool and one tool only
- Do not create additional steps
  - Or hide them under the covers
  - Pays off with each iteration

# Development cycle using Eclipse



# DEMO

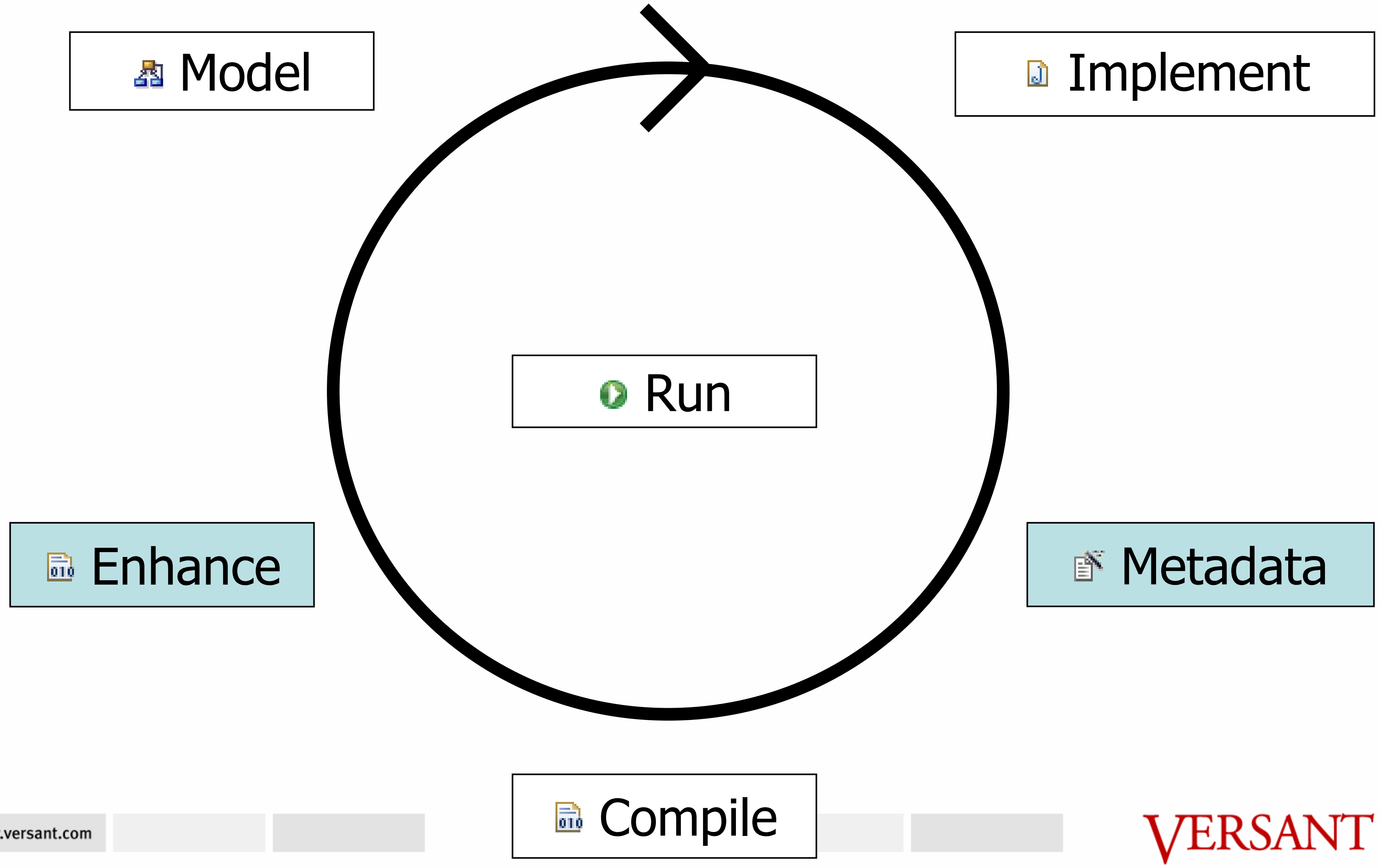
## In the future

- Annotation for Metadata (JDO 2.1 & Java Persistence API (JPA) 1.0)

```
@Persistent
public class Person {
…..
}
```

- Runtime Enhancement (JDK 1.5 „–javaagent“)
  - Enhancement during application startup

## Development cycle using Eclipse



VERSANT

## Questions?

## QA

- Who uses Eclipse?
- Who is still using JDK 1.4, or even older?
- Who uses JDO?
- Who plans on using JDO in the future?
- Who uses JPA?
- Who plans on using JPA in the future?
- Who uses Hibernate?

THINK OUTSIDE THE GRID.
NON-SQUARE DATA MANAGEMENT.

www.versant.com

VERSANT