VERSANT

e-business to the power of Versant

E-Business Plattform
Versant enJin

Java Forum 2000

Stuttgart, 28.06.2000

Gerhard Klein, Manager Professional Services CE

gerhard.klein@versant.de

http://www.versant.de

e-Business Plattform

- **Komponenten:**
  - **Anwendungen**
    - CRM, Supply Chain Management andere e-Commerce Anwendungen
  - **e-Commerce Funktionalität:**
    - Personalisierung, Auktion, Billing, Portal
  - **Integrationsfunktionalität**
    - XML, Datenintegration, Anwendungsintegration
  - **Applikationsserver**
    - Komponenten, Persistenz, Messaging, Security, Load Balancing
- **Nächste Generation**
  - **Integration und Intelligenz**

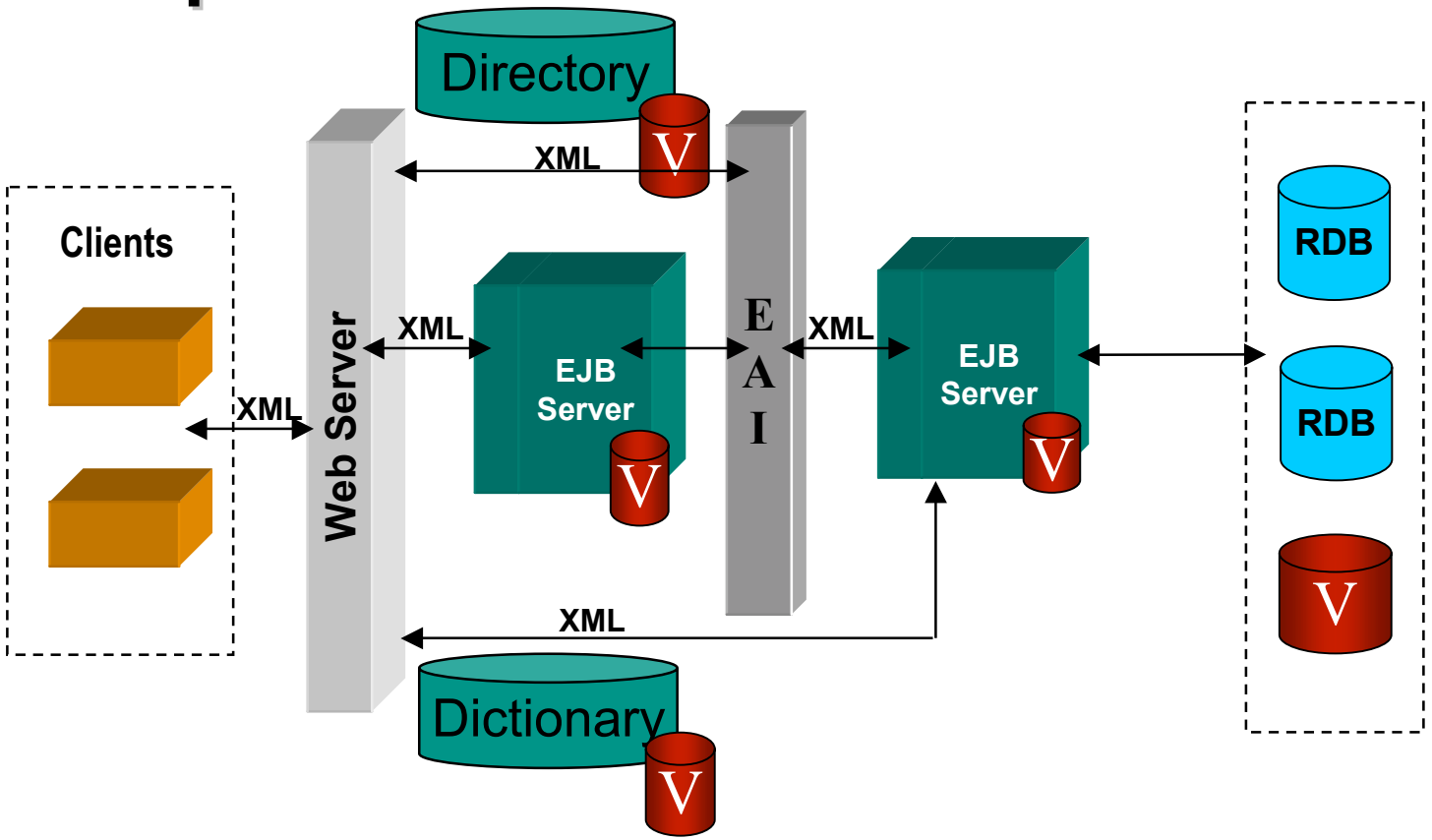
Anforderungen

- **Skalierbarkeit und Performanz**
  - Hierbei sind häufig die existierende Systeme der Engpaß
- **Integration im Mittel-Tier**
- **Komplexes Modell und viele navigierende Zugriffe**
- **Time-to-Market Zeiträume werden immer kürzer**
  - Produktivität der Entwicklungswerkzeuge und Flexibilität
- **Zukunftsweisende Architektur**
  - Standards: J2EE, EJB, JDO, XML

"Many people are under the misconception that scalability is a function of hardware."
- Anne Thomas Manes, Patricia Seybold Group

Source: Patricia Seybold Group, 12/99

Beispiel: France Telecom



Jean-Paul Maury, CIO, France Telecom
"Our rigorous testing proved to us that standardizing on Versant ODBMS, as one of our key architectural components, will allow us to continue to lead the industry in customer service."

Daten im Middle Tier

- **Bei den Daten kann man unterscheiden zwischen**
  - **Business Daten:** z.B. Business Transaktionen
  - **Intermediate Daten:** alle anderen Daten, die auch im Middle Tier gehalten werden
    - **Business Intelligence Daten**: z.B. für personalisiertes Marketing
    - **Meta Daten:** beschreiben z.B., wie mit Back End System und Legacy Datenbanken interagiert werden soll.
    - **Workflow Daten:** Business Regeln und aktueller Bearbeitungszustand
    - **Session Daten:** für aktuelle Einstellungen und Aktivitäten
- **Wie werden sie verwaltet?**
  - **Als persistente Objekt im Middle Tier**
  - **Business Daten werden weitergereicht**

Synchronisation mit anderen Datenquellen

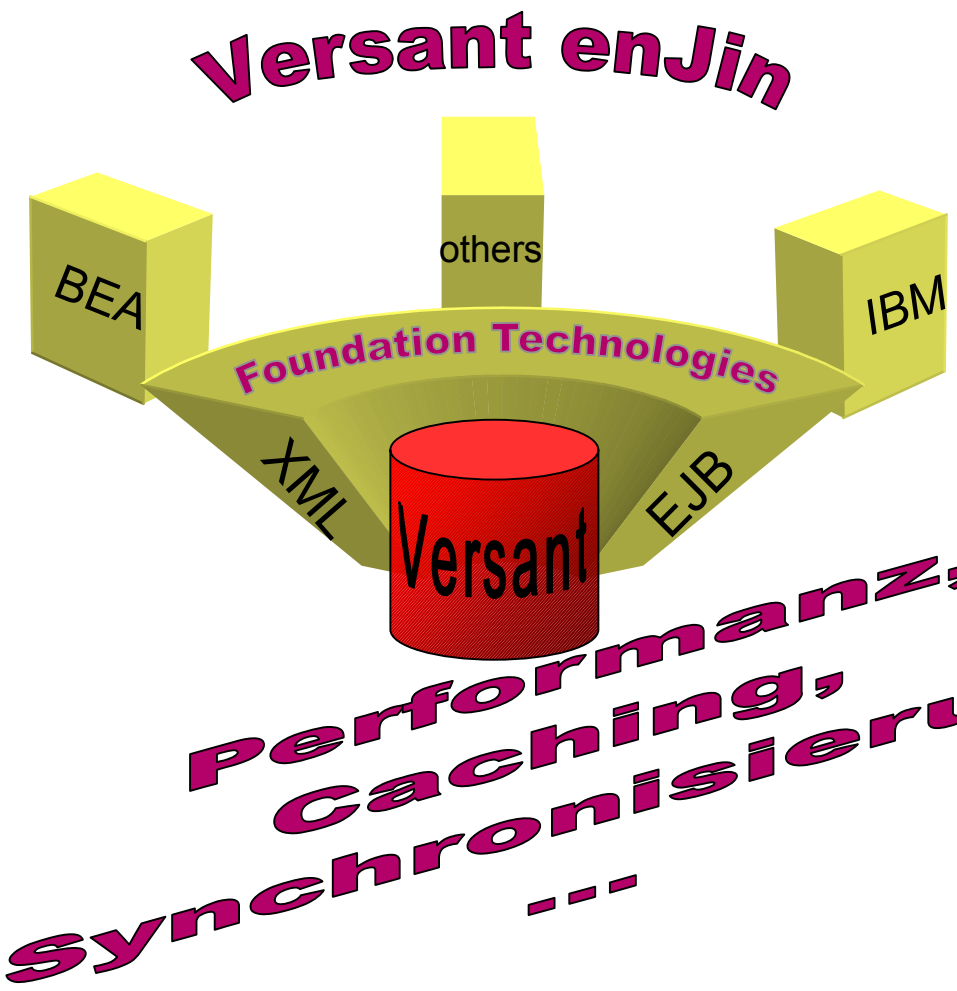
- **Wann wird aus Versant geschrieben?**
  - An Transaktionsgrenzen („write-thru“, JTA)
  - Am Tagesende („intra day“)
  - Workflow Ereignis
    - Neue Order wurde plaziert
    - Einkauf wurde abgeschlossen
  - Niemals (Metadaten, Neue Daten)
- **Wann wird Versant aktualisiert?**
  - Niemals, da „write thru“ synchron ist
  - Nach jedem Schreiben
  - Periodisches Pollen („Änderungstabelle vom Host“)
  - Workflow Ereignis (Neues Produkt wurde zum Katalog im Legacy System hinzugefügt)

Versant enJin

- „Provide innovative Software solutions to enable the development of e-Business Applications“
  - Verteilte Umgebungen, die Softwarelösung für Internet Skalierbarkeit benötigen
  - Performanz, Skalierbarkeit und Verfügbarkeit
  - Einfache Entwicklung, Time-to-market
- Versant enJin Produkt Suite
  - OODBMS Versant: implementiert den kommenden Java Data Objects (JDO) Standard
  - Nahtlose Integration mit den führenden EJB App. Servern
  - enJin eXPress: Lightweight Objekt-Relationales Mapping
  - Synchrone und asynchrone Replikation
  - XML Toolkit

Versant enJin

e-business **Objekt Management für e-Business Anwendungen**



Versant Komponenten

- ◆Versant Enterprise Container
  - Integration mit Application Servern von BEA WebLogic und IBM WebSphere
  - "Portabiltätsgarantie" über *Container Managed Persistence*
- ◆Versant Express
  - Leightwieght OR-Mapping Toolkit
- ◆Asynchrone Replikation
  - Abgleich mit Legacysystemen
- ◆Versant Java Developer Suite
  - Hochperformante Persistenzlösung für Java Anwendungen
  - Reduziert die Entwicklungszeit um 40%
  - Implementiert JDO
  - Fehlertoleranter Server als Option
- ◆XML Toolkit
  - XML Import und Export
  - Transformation von Java Objekten

Vorteile einer OODBMS im Middle Tier

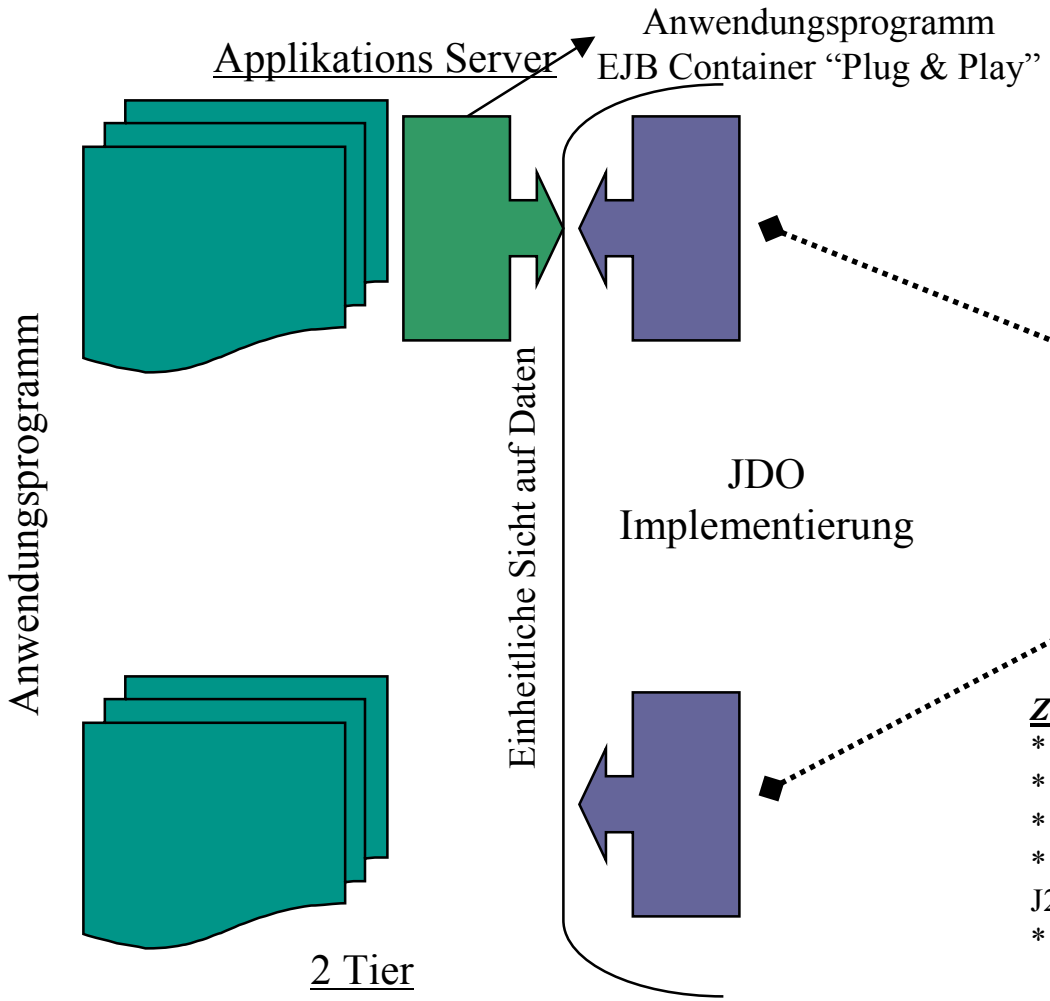
- **"Time to Market" reduzieren**
  - Klares, natürliches Design
  - Nur ein Modell für Anwendung und Datenbank
  - Keine spezielle Zugriffsschicht (mapping code)
- **Bis zu 40% weniger Code zu erstellen / zu warten**
- **Exzellente Performanz der Anwendung**
  - Navigation ist schneller, intuitiver als Joins
  - unterstützt transparente Verteilung
- **Senkt den Aufwand für Wartung und Administration**
- **On-line schema Evolution**
  - Einfachere Handhabung durch weniger Altlasten

Java Data Objects (JDO)

Definiert einen Kontrakt zwischen Programmierer und JDO Anbieter

- **Ziel: Standard API für transparenten Datenbankzugriff**
  - **Nahtlose Integration von der Programmiersprache Java mit Datenbank**
  - **Fokus liegt auf**
    - Integration mit DBMS und Applikationsservern
    - direkter programmatische Zugriff auf Unternehmensdaten
- **JDO und JDBC**
  - **Mit JDBC muß der Programmierer**
    - explizit die Attributwerte verwalten und auf Tabellen mappen
    - die Spezifika unterschiedlicher Datenbanksysteme kennen
  - **JDO ersetzt nicht JDBC**
    - Ergänzende Technologien
    - Standardisierter Zugriff auf Objektdatenbanken und Objektrelationale Mappings
    - JDO Programmier kann sich auf Businesslogik konzentrieren

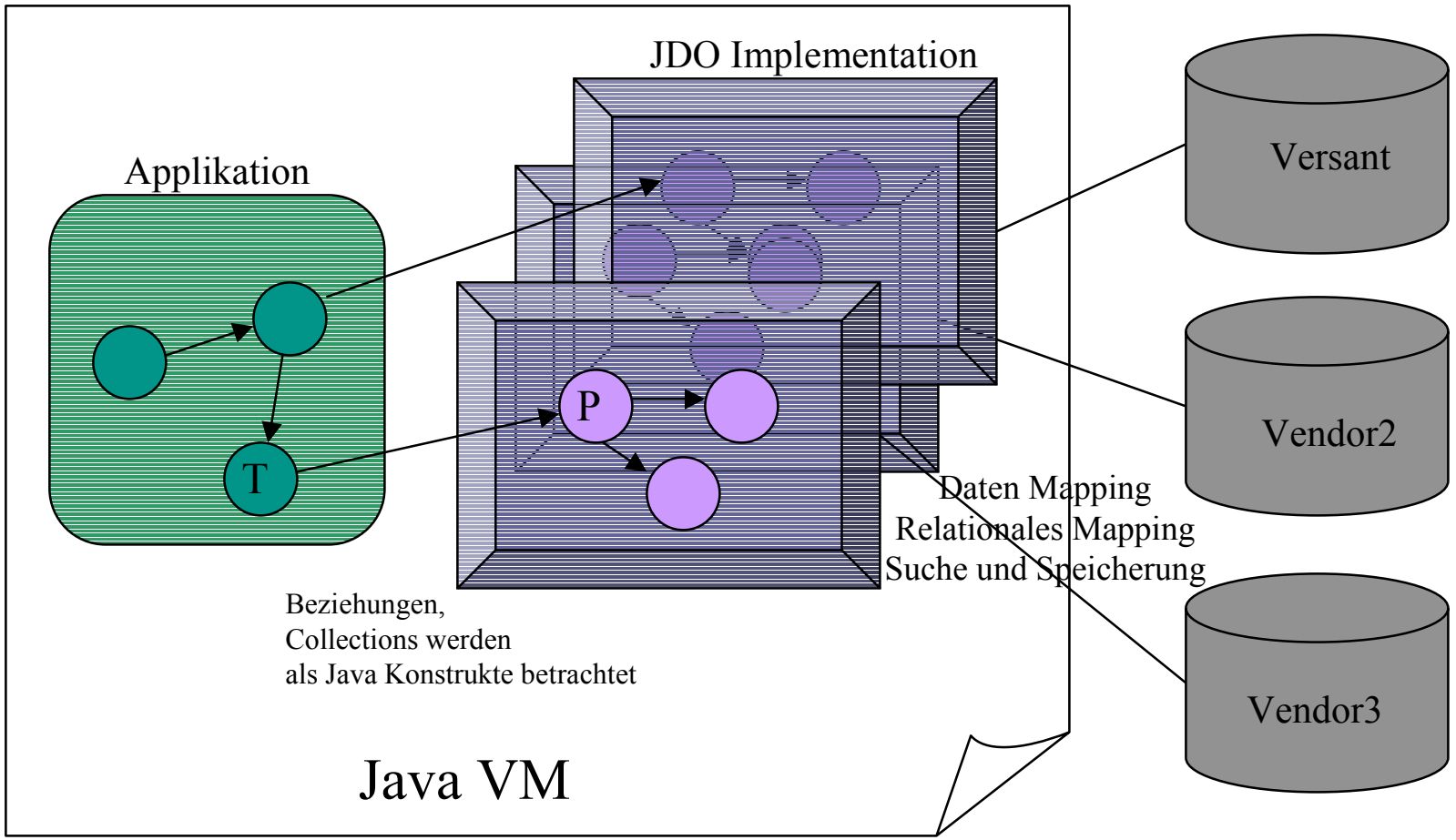
Basis JDO Architektur



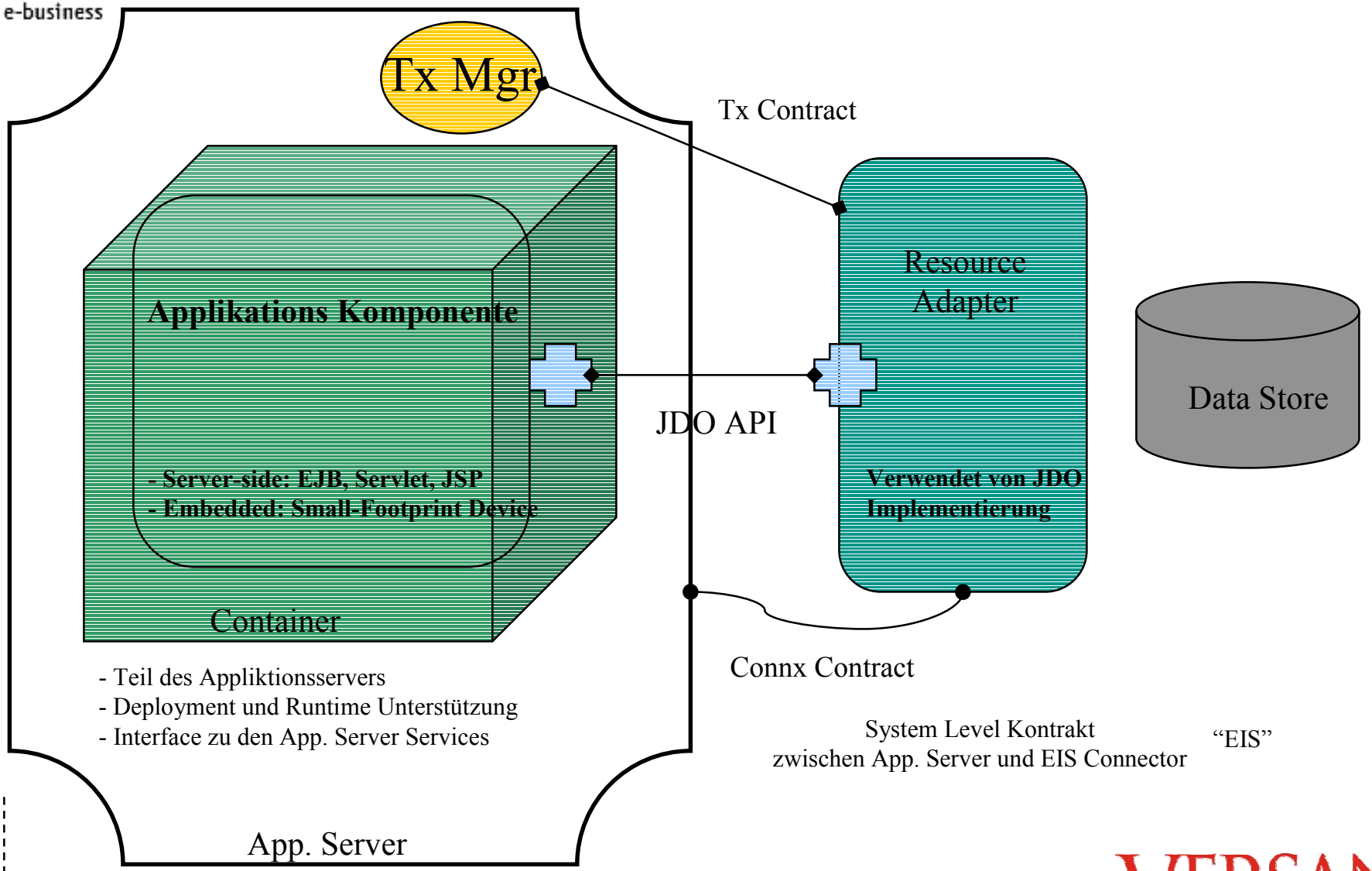
Ziele

* Transparentes Interface ohne neue Datenzugriffssprache
* Skalierbare und transaktionelle Implementierung
* "Small Footprint" für Applikationsserver
* Verwendet die Connector Architektur zur Integration mit J2EE Compliant App. Servern
* Kontrakte und Verantwortlichkeiten wurden definiert

JDO 2-Tier Überblick



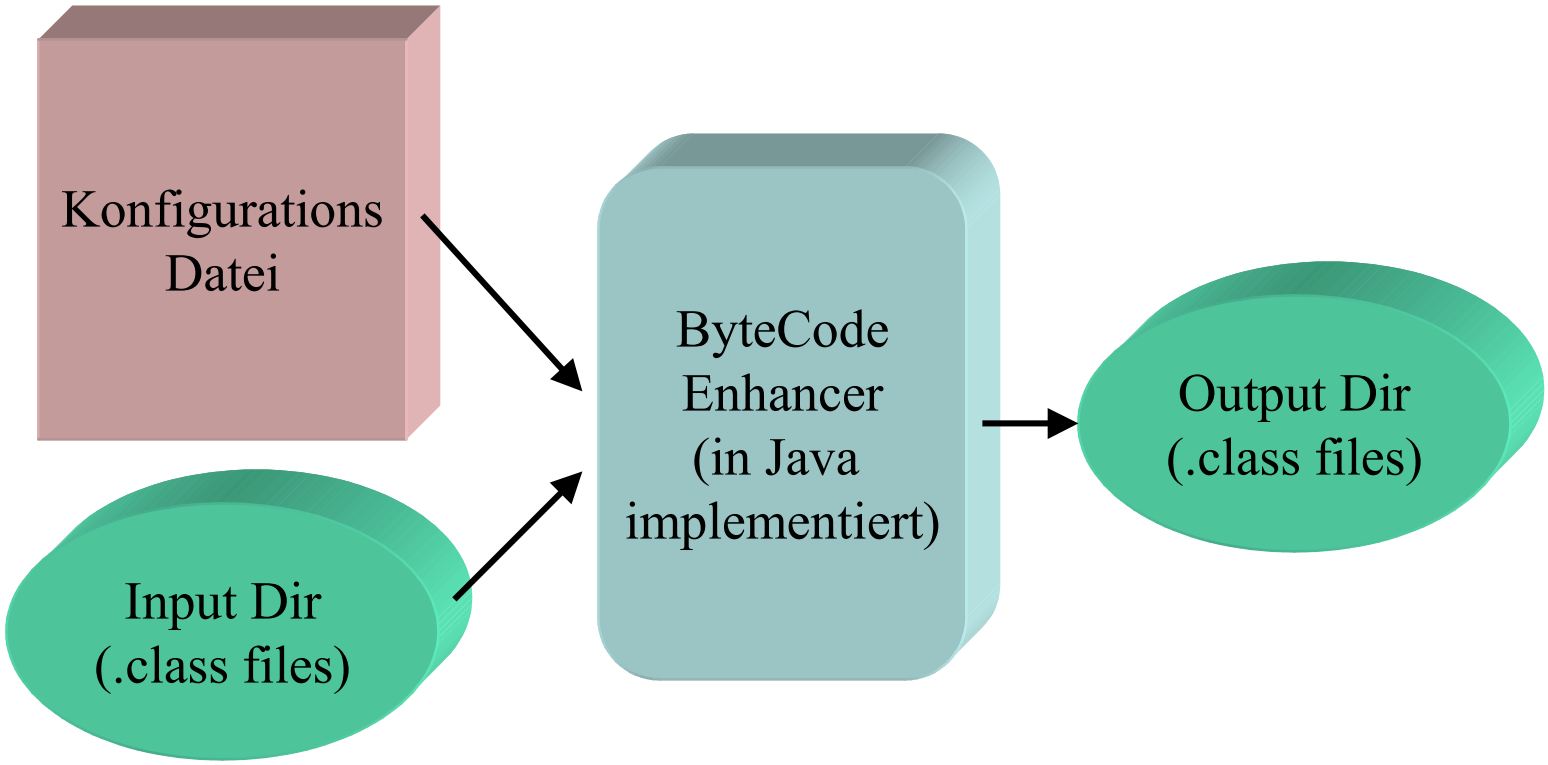
JDO mit Applikationsserver



Entwicklungsschritte

- **Entwicklung mit JDO :**
  - **Entwicklung eines Objektmodells. Entscheiden welche Klassen persistent gemacht werden sollen**
    - Dies kann mit Tools wie z.B. Together, Rose, ArcStyler erfolgen
  - **Implementierung der Java Klassen**
    - Für die persistenten Java Klassen wird die gleiche Semantik wie für die transienten verwendet
    - Die Vererbungshierachie und Attributtypen werden entsprechend in der Datenbank abgebildet
    - Collection Klassen können verwendet werden
  - **Implementierung der Transaktionen in den Anwendungsprogramm**

Byte-Code Enhancement



java com.versant.Enhance -config config_file
-in in_dir -out out_dir

Unterstützt In-Place Enhancement

VERSANT
e-business to the power of Versant

JDO: Beispiel

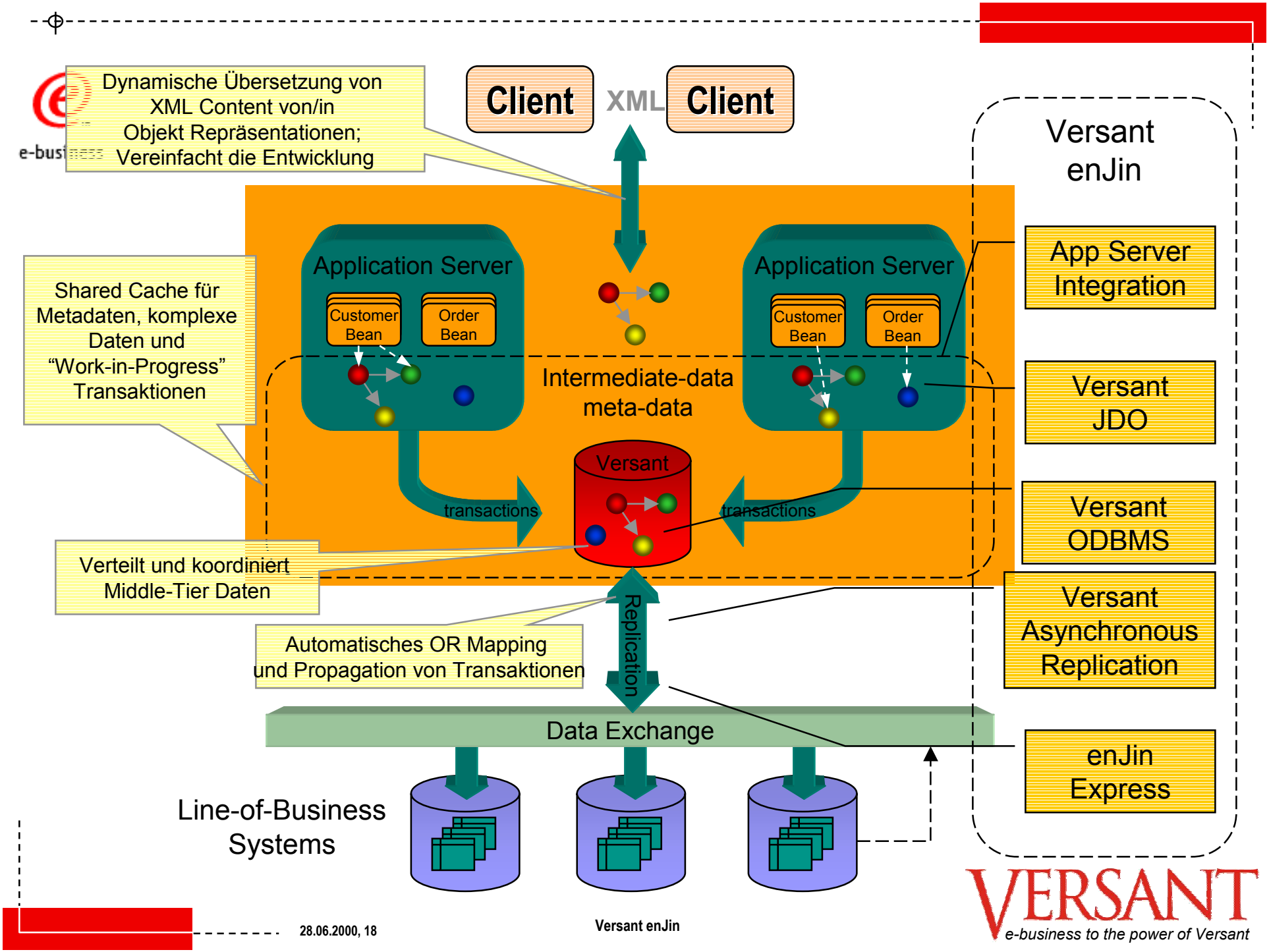
```
public class Person{
   private String name;
   public Person (String n)  { name = n;}
  public void setName(String n) { name = n;}
}
public class PersonMain{
    public static void main (String args[])  {
        PersistenceManagerFactory factory =
            new VersantPersistenceManagerFactory ();
        PersistenceManager pm = factory.getPersistenceManager (args[0]);
        Transaction tx = pm.currentTransaction ();
        Person newPerson;
        tx.begin ();
        for (int i = 0; i < args.length; ++i) {
               newPerson = new Person(args[i]);
               pm.makePersistent (newPerson);
        }
        tx.commit(); tx.begin();
        newPerson.setName(„Mr. X“);    // now change the last person
        tx.commit();
        pm.close ();
    }
}
```

Versant enJin Express

e-business

- **Objekt-relationales Mapping für e-Business Apps**
  - **Nahtloses OR-Mapping für Java Objekte, EJB**
  - **Verwaltet komplexe Beziehungen zwischen Objekten**
    - 1-1, 1-M Assoziationen, vererbung, Pfad Queries und Pre-Post Hooks
  - **Nutzt Datenbank Features**
    - Stored Procedures, Views, BLOBs
  - **Dynamisches Mapping anhand von Meta-Data (XML Beschreibung)**
  - **Lightweight – Internet Ready (Pure-Java < 300k)**
  - **Performanz**
  - **Integration mit führenden Applikationsservern (EJB, ...)**

Dynamisches Objekt-Mapping ...



VERSANT
e-business to the power of Versant

Zusammenfassung

$$\sum_{EJB}^{XML} = e^{\text{VERSANT}}$$

e - business to the power of Versant

scalable persistence for Java

VERSANT