ETH Eidgenössische Technische Hochschule Zürich Swiss Federal Institute of Technology Zurich
inf Informatik Computer Science
IKS Information and Communication Systems Research Group

Making Applications Persistent at Run-time

Angela Nicoara, Gustavo Alonso / ETH Zurich, Switzerland

Cluster4

http://prose.ethz.ch

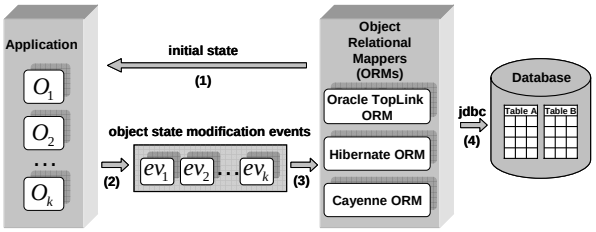
- Persistence has become a common requirement in many applications
- Persistence is an orthogonal concern typically implemented by the underlying middleware infrastructure and not by the application developer
- In existing systems, persistence is added to an application either at compile or deployment time by using a variety of mechanisms

- **Orthogonal Persistence** – implemented as:
  - Language extensions
  - A development time property (e.g., ORMs)
  - A deployment time property (e.g., container)
  - A compile time property

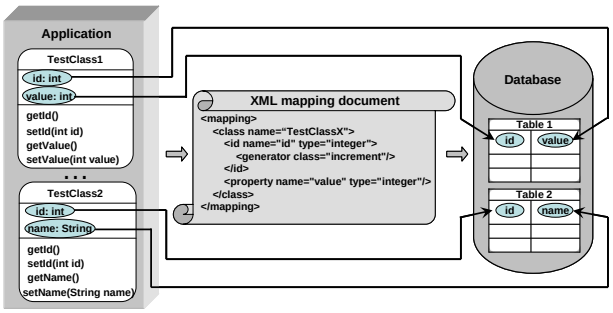
- **Contributions – Dynamic Persistence:**
  - A Java-based dynamic AOP infrastructure where persistence functionality can be dynamically added or removed from running applications
  - Persistence as a run-time property
  - Persistence behavior can be added to existing applications without requiring to modify their original code
  - Making the application independent of the container
  - Application can be made persistent at run-time without having to stop or redeploy the code

Dynamic Persistence

Architecture for object persistence



Mapping application objects to tables



Persistence as an aspect

```
public class PersistenceAspect extends DefaultAspect {
    public SessionFactory sf;
    public Crosscut makePersistent = new SetCut() {
        // advice method: runs before field modifications of type "Integer"
        public void SET_ARGS(Object obj, Integer f) {
            Session ses = sf.openSession();  // session handling
            ses.saveOrUpdate(obj);           // object writes or updates
            ses.flush();
            ses.connection().commit();  // commit the changes into the database
            ses.close();
        }
        protected PointCutter pointCutter() {
            return (   (Fields.declaredInClass("Service.*"))  .AND
                       (Fields.named("field.*")) );
        }
    }
}
```

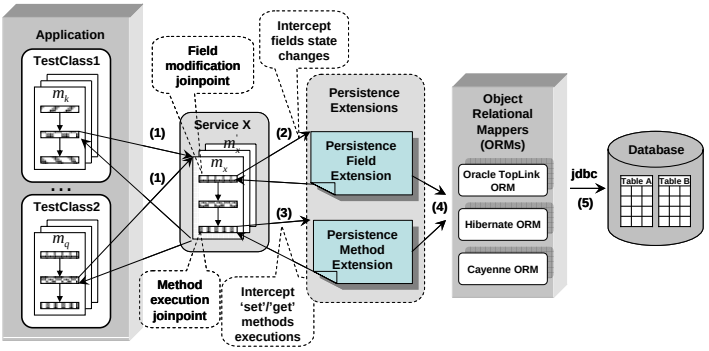
Integer *.*

advice method

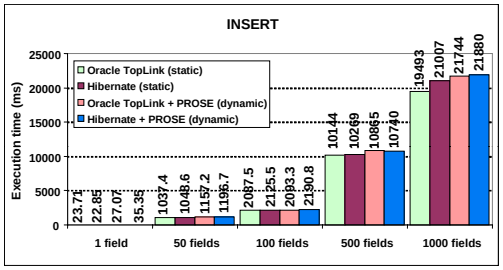
before field modification: declaredInClass(Service.*) && named(field.*)

pointcut method

Adapting running applications

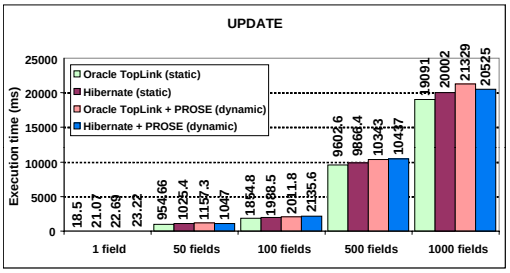


Performance evaluation



INSERT measurements with Oracle TopLink, Hibernate and PROSE with Sun JVM

- Performance penalty: the AOP support leads to an average overhead of 8.92 % in case of Oracle TopLink



UPDATE measurements with Oracle TopLink, Hibernate and PROSE with Sun JVM

- Performance penalty: the AOP support leads to an average overhead of 14.35 % in case of Oracle TopLink

FNSNF Fonds National Suisse Schweizerischer Nationalfonds Fondo Nazionale Svizzero Swiss National Science Foundation
EPFL École Polytechnique Fédérale de Lausanne
UNIL Université de Lausanne
ETH Eidgenössische Technische Hochschule Zürich Swiss Federal Institute of Technology Zurich
UNIVERSITÄT BERN
csem
UNI BASEL