ORM:

Problem:
Impedance Mismatch (i.e., different languages for data and programming, need casting between types, makes analysis difficult)

Solution:
Object-Relation-Mapping middleware (provide an persistence abstraction for objects, and takes care of transformation from/to the DB world)
"Everyone who is somebody has one! Either standard (e.g., hibernate) or ad-hoc."

The idea is to provide:
- pre-canned mapping between OO classes/fields and table/columns
- manually defined mappings
- provides object persistency without looking at the DB

Good:
- abstraction
- ease of debug

Bad:
- performance

Example: Hibernate

Example of Hibernate Mapping

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
Example of Hibernate Usage (many details are hidden)

```java
Honey honey = new Honey();
honey.setName("forest honey");
honey.setTaste("very sweet");
...
    tx = session.beginTransaction();
    session.save(honey);
    tx.commit();
    ...
    tx = session.beginTransaction();
    session.update(honey);
    tx.commit();
    ...
    tx = session.beginTransaction();
    List honeys = session.createQuery("select h from Honey as h").list();
    tx.commit();
```

---

Next we talk about DriadLINQ… it provides similar features but adds much more in particular:
- LINQ language integration
- Batch-oriented
- Cluster-oriented
- More than SQL