JNoSQL (Cf) Dual Licensing Request to the Eclipse Foundation

NoSQL

- Database
- Doesn't use relationship
- BASE
- Four types

Description

The JNoSQL is a several tools to make easy an integration between the Java Application with the NoSQL. To solve this problem the project has two layers:

- **Communication** API: An API just to communicate with the database, exactly what JDBC does to SQL. This API has four specializations, one for each kind of database.

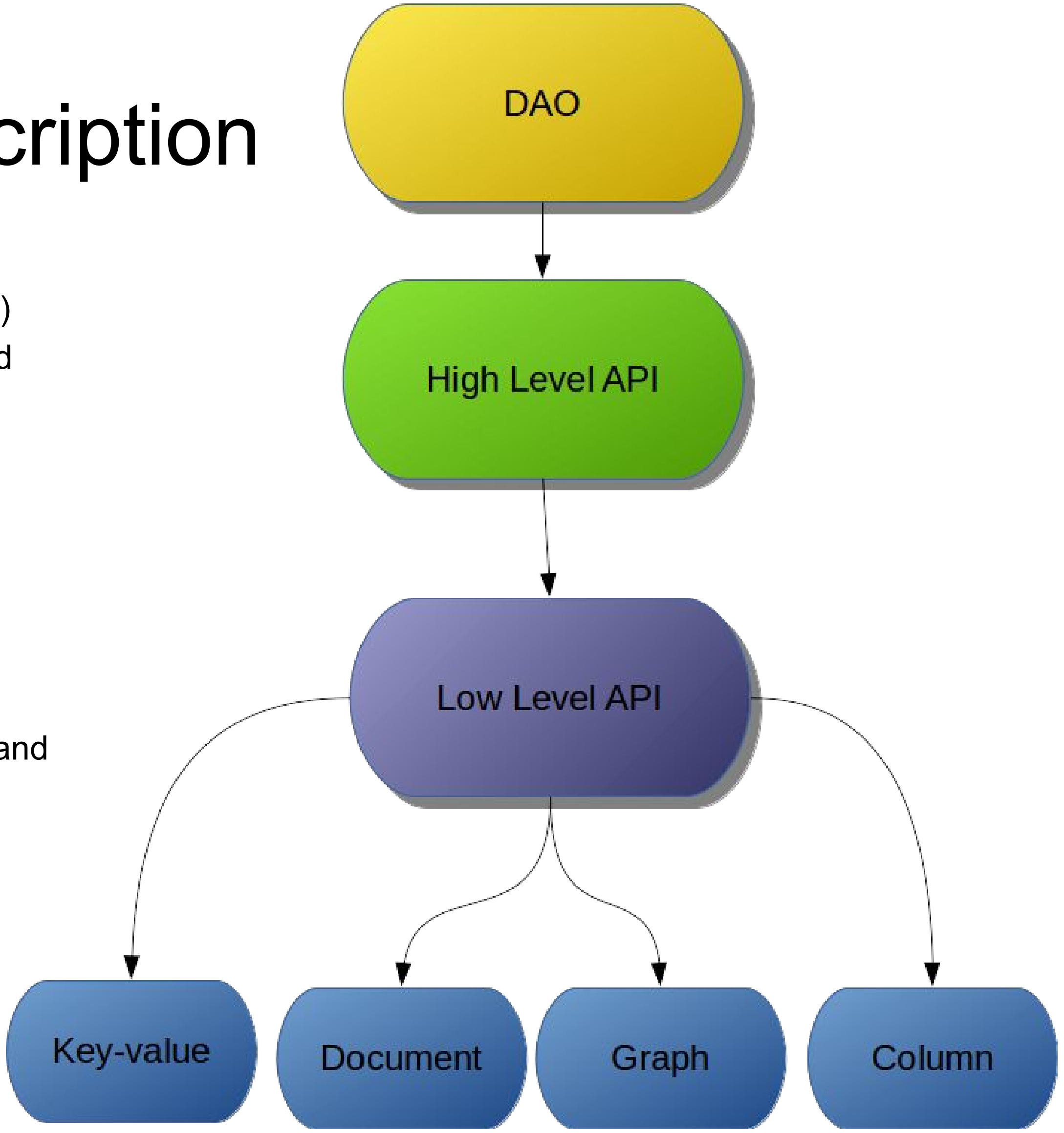
- **Abstraction** API: An API to do integration and do the best integration with the Java developer. That is annotation drive and has integration with other technologies like Bean Validation, etc. To solve it this layer is a CDI based.

This way, the NoSQL vendors just need to implement the communicate API without warning about the another API.

So, the project has two core project that is Diana and Artemis project.

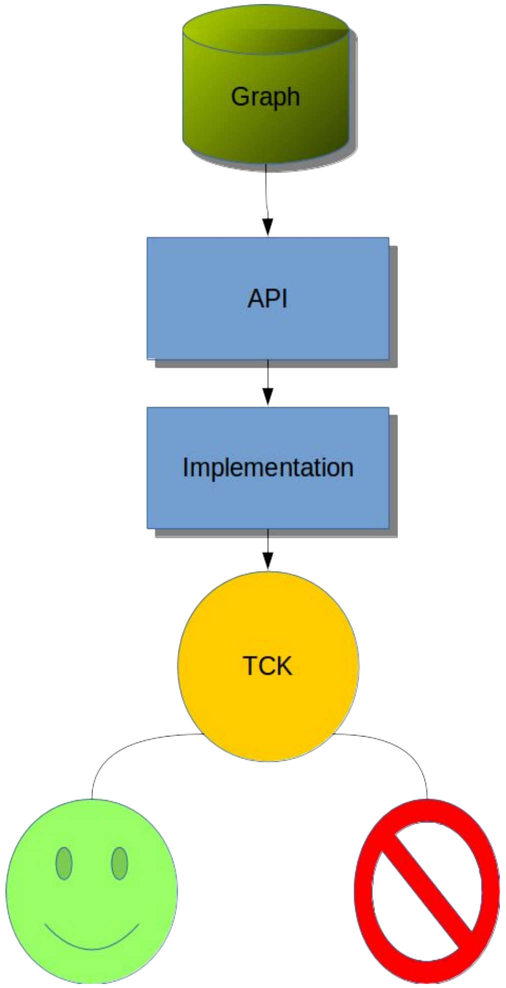
Description

- **Artemis** as abstraction API:
  - Communicate with the Data Access Object (DAO)
  - Integrate with other technologies such as CDI and Bean Validation
  - Focus on integration to make the developer life easier
  - What the JPA does to SQL databases
- **Diana** as Communication API:
  - communicate with the database
  - Manage Connection
  - Focus on communication between the database and the application
  - What the JDBC does to SQL databases



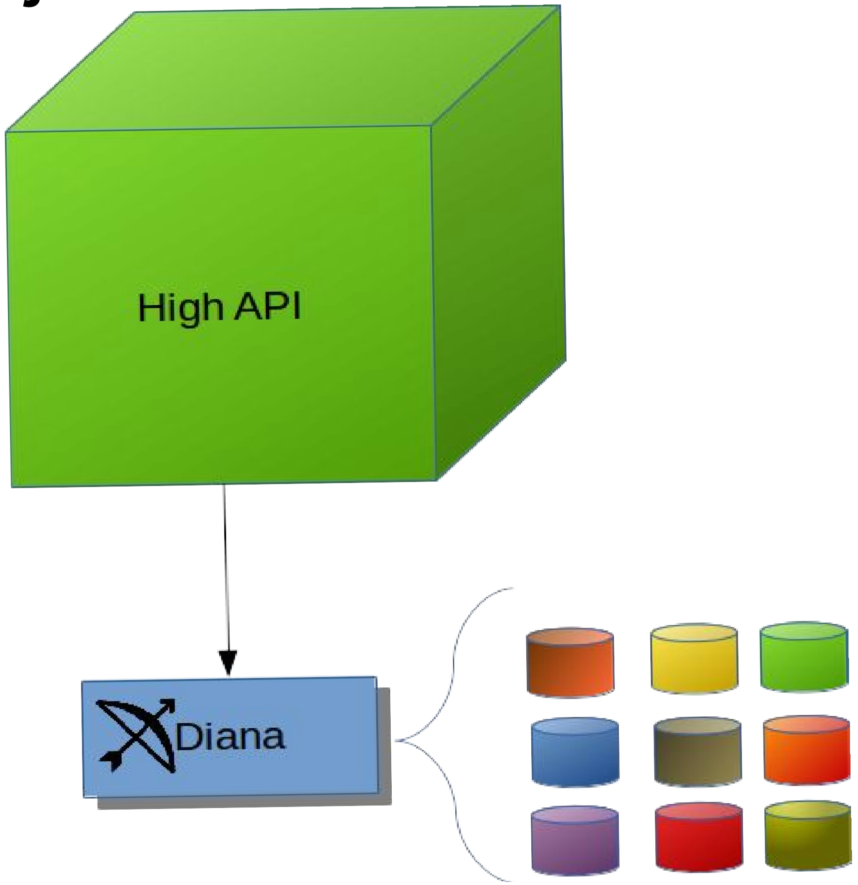
Diana Project

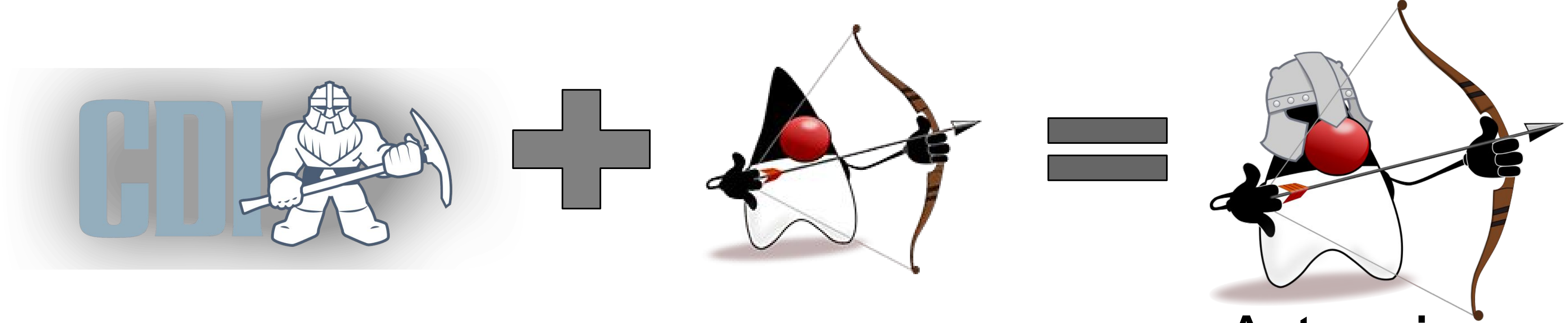
- Diana has four API, one to each NoSQL type, and its respective TCK, Technology Compatibility Kit. So, if a NoSQL vendor wants to support Diana project just need to implement an API, from his specific database, and then run the TCK, if all tests had passed this database supports Diana API. If they want is possible to add supports to specific behavior that just its database has.



Diana Project

Just with Diana API base, is possible to connect a several kind of NoSQL database.

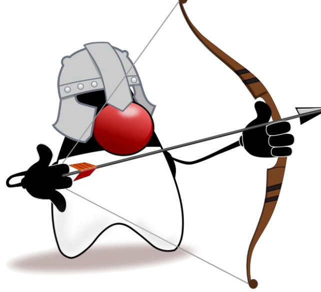




Artemis

- CDI Based
- Diana Based
- Annotation Based
- Events to insert, delete, update
- Supports to Bean Validation
- Configurable and extensible

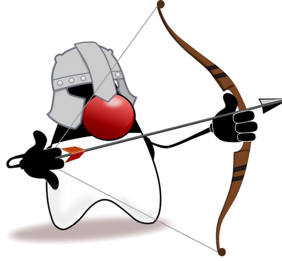
Annotations

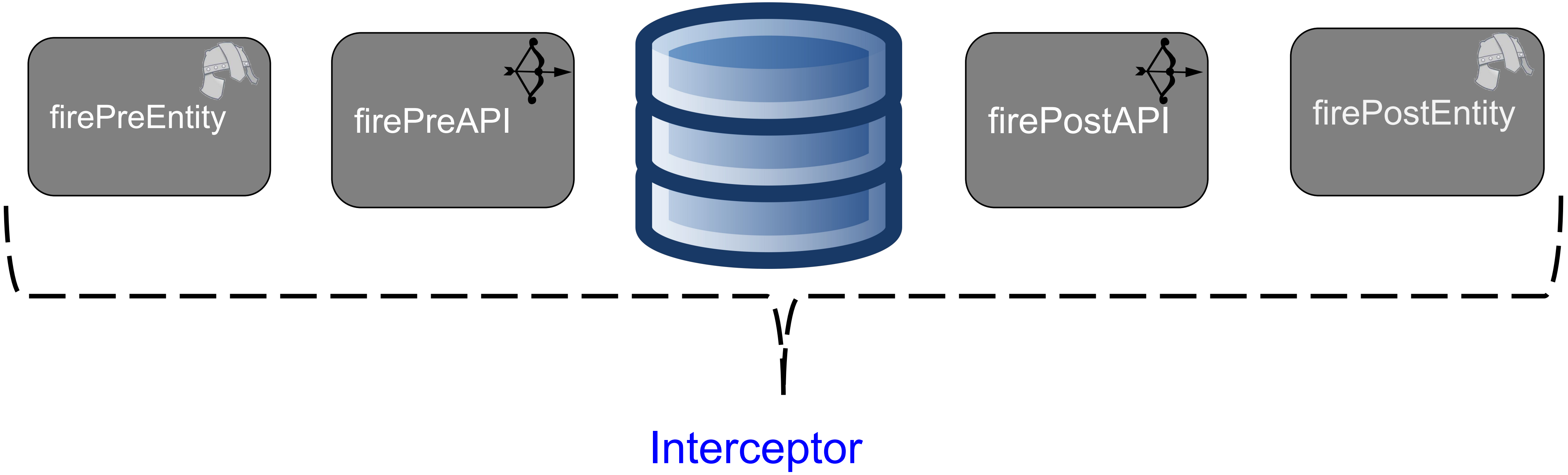


- MappedSuperclass
- Entity
- Column

```
@Entity("movie")
public class Movie {

  @Column
  private String name;

  @Column
  private long year;

  @Column
  private Set<String> actors;
```

Events




Movie
firePreEntity
firePreAPI
firePostAPI
firePostEntity
Interceptor

Scope

- Standardize the NoSQL databases, but don't forget the diversity that there is on that.
- A simple API to support Column NoSQL Database
- A simple API to support Key-value NoSQL Database
- A simple API to support Graph NoSQL Database
- A simple API to support Document Database
- Convention over configuration
- Support for asynchronous queries
- Support for asynchronous writes operations
- An easy API to implement, so that NoSQL vendors can comply with it and test by themselves.

Initial Developers

- Anatole Tresch
- Werner Keil
- Otavio Goncalves de Santana
- Luca Garulli
- Oliver B. Fischer
- Gerald Sangudi
- Prasad Varakur
- Christoph Engelbert
- Johan Larson
- Daniel Cunha

Initial Developers

- Spring Data http://projects.spring.io/spring-data/
- Hibernate ogm: http://hibernate.org/ogm/
- Eclipselink: http://www.eclipse.org/eclipselink/
- Jdbc-json: https://github.com/jdbc-json/jdbc-cb
- Simba: http://www.simba.com/drivers/
- Tinkerpop: http://tinkerpop.apache.org/