The Holmes Platform and Applications

Luxembourg, 2017

The **Vision**

Flexible
Software

Define your Processes

Keep it
Simple

Process **Automation**

Follow **Open**
Standards

Don’t re-invent, **Reuse**

The **Product**

The **Holmes** Platform

Application Development Framework designed to develop Enterprise Apps.

Productivity is enchanced thanks to a development and execution framework covering the common layers of the enterprise application’s design

data access | business logic | integration | ui

The **Holmes** Platform

¿What does **Holmes** offer?

- Reduction of the development effort.
- Reduction of the development time for simple frequent changes of information systems, such as data model extensions.
- A standard architecture and a development methodology acting as a foundation for the development of enterprise applications.
- Based on Open Source products, Holmes aims to reduce the Total Cost of Ownership (TCO) of the solution, when compared to other customizable systems.

data access | business logic | integration | ui

The **Holmes** Platform

¿How is it achieved?

- Clean separation between platform and application.

Result →

Plataform Independent from Data Model and Business Logic

- User interface **independent from data model**
  - Generic forms are generated automatically
  - Still, it is possible to design customized forms.
- User interface **independent from business logic**
  - MVC interface based on events and delegates.

The platform offers the API and runtime environment to develop and execute the resulting application.

data access | business logic | integration | ui

Data Access

Data Access

- The Data Access layer is the API to access the database.
- Whenever you change the database schema, the Data Access layer rebuilds the object model, which holds the metadata (information about tables, views, relationships). The schema changes are auto-discovered and cached locally and stored in the system tables.
- The Data Access API (OLAPI) has the following features:
  - It is independent from the database schema
  - It is “cloud-ready” which means that it is suitable for mobile applications, access through internet and firewalls.
  - It is designed with performance and scalability in mind.
  - It is multi-platform and based in open standards (JDBC, JavaBeans, XML serialization and Java EE).
  - The SQL API generates the SQL code dynamically.
- Supported configuration: MySQL / JBoss/Wildfly

data access | business logic | integration | ui

Business Logic

Business Logic

- Every user action becomes an Event that is managed through a Controller. The events are classified into Data, Business, UI and Remote categories.
- The Data events are handled by a delegate class that performs a database query to either select or update the focus object associated to the event.
- The Business events are those events that require a specific behavior dependent on the object type. A simple plug-in based on the class name is devised so that you can add behavior when an event to select or update event is processed. Moreover, the Controller can be extended to pre-process or post-process the events.
- The Remote events are only available in the 3-layer setup. These events are sent to a Web Service to be handled in the server side.
- Presentation logic is triggered by UI events.

data access | business logic | integration | ui

Integration

Integration

- The 3-layer setup is based on the Java EE architecture which is a widely used industry-standard for building enterprise applications.
- Externally, the SOAP & REST WS is also a open standard defined by the W3C, implemented by most software vendors (Microsoft, Sun, BEA, IBM…)
- E-Mail, LDAP, JTAPI integration is implemented out of the box.
- The Business Logic & Data Access API enable to build programs that can be scheduled run in the background (batches, daemons) for integration and backoffice purposes.
- The client-side can be extended to integrate with external services.
- The client is integrated with the operating system using the third-party product Java Desktop to provide file type-association features, web browser, etc.

data access | business logic | integration | ui

User Interface

User Interface

- The application client is the user interface. Two setups are possible, for small-office applications a DB client connects directly with a database server, while in the large scale setup the Enterprise client connects to a Web Service/J2EE server.
- The client obtains from the Data Access layer the necessary information to build up a default, generic forms to create, select or edit the records of your database.
- Basic customization does not require programming. The following features can be customized, through the application, without programming:
  - Menus, Lists, Multi-language labelling, Related-info tabs…
- The 3-layer architecture client enables client to connect through internet and firewalls and enables horizontal scalability.
- Multi-platform (works on Microsoft / Linux / Sun…)
- Plug-in architecture. You can override at form-level and even the internal event controller to define your own events and delegate.

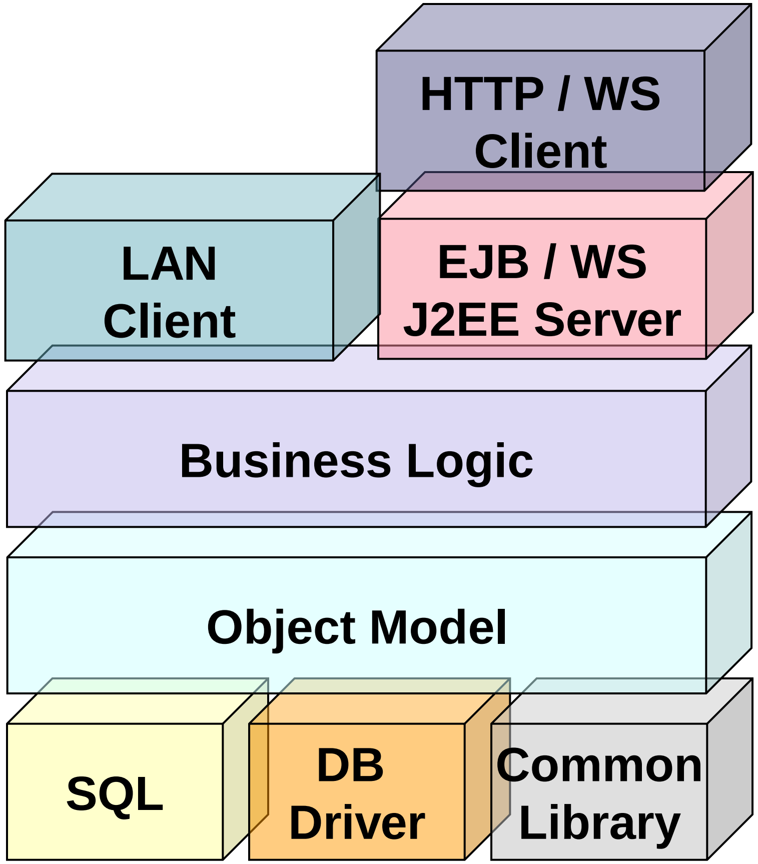
data access | business logic | integration | ui

Holmes Architecture

Auto-generated
Configurable
Extendable
User Interface

Event-driven
Business Logic

SQL Object API
No SQL embedded
nor parsing



J2EE / Web
Service Arch for
Enterprise-Class
Deployments

Data Model
Schema Discovery
High-level Data
Access API

Platform **Requirements**

- Database Server

  MySQL 5

- Application Server

  Wildfly 10

- Multi-plataform

  Java 8 / Java EE 7

  S.O. Windows, Linux, Solaris, HP-UX…

Holmes Applications

The **Holmes Applications** are application modules, customizable, covering the following functional areas of Customer Relationship Management:

- ❑ **Call Center**
- ❑ **Support**
- ❑ **Sales**
- ❑ **Product Catalogue**
- ❑ **Billing**

Thanks...