District Health Information Software 2

(DHIS 2)
Introduction

• Purpose
  – Demonstrate and provide information about DHIS 2 for project work
  – Demonstrate use of frameworks in a real-life application

• Contents
  – Domain model
  – Project structure & repository layout
  – Application design
  – Hibernate configuration
  – Testing
  – Development tips and conventions
DHIS 2

• Application for collection, validation, analysis, and presentation of aggregate / statistical data

• Tailored to health information management

• Dynamic data model and flexible user interface

• Based on open source Java development frameworks
Domain model

- **DataElement**
  - Meta-data definition of captured entities
  - Example: *BCG doses given for infant under 1 year*

<table>
<thead>
<tr>
<th>DATAELEMENT</th>
<th>Java-type</th>
<th>Db-attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>uuid</td>
<td>string</td>
<td>unique</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>alternativeName</td>
<td>string</td>
<td>unique</td>
</tr>
<tr>
<td>shortName</td>
<td>string</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>code</td>
<td>string</td>
<td>unique</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>-</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>not-null</td>
</tr>
<tr>
<td>aggregationOperator</td>
<td>string</td>
<td>not-null</td>
</tr>
</tbody>
</table>
Domain model

- **Period**
  - Time interval between start- and end-date

- **PeriodType**
  - Example: *Yearly, quarterly, monthly, weekly*

<table>
<thead>
<tr>
<th>PERIOD</th>
<th>Java-type</th>
<th>Db-attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>periodType</td>
<td>PeriodType</td>
<td>-</td>
</tr>
<tr>
<td>startDate</td>
<td>Date</td>
<td>-</td>
</tr>
<tr>
<td>endDate</td>
<td>Date</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PERIODTYPE</th>
<th>Java-type</th>
<th>Db-attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>name</td>
<td>name</td>
<td>not-null</td>
</tr>
</tbody>
</table>
Domain model

• OrganisationUnit
  – Example: Ministry of Health, Province, District, Hospital, Ward

<table>
<thead>
<tr>
<th>ORGANISATIONUNIT</th>
<th>Java-type</th>
<th>Db-attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>uuid</td>
<td>string</td>
<td>unique</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>shortName</td>
<td>string</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>organisationUnitcode</td>
<td>string</td>
<td>unique</td>
</tr>
<tr>
<td>openingDate</td>
<td>Date</td>
<td>-</td>
</tr>
<tr>
<td>closedDate</td>
<td>Date</td>
<td>-</td>
</tr>
<tr>
<td>active</td>
<td>boolean</td>
<td>-</td>
</tr>
<tr>
<td>comment</td>
<td>string</td>
<td>-</td>
</tr>
</tbody>
</table>
Domain model

- **OrganisationUnitGroup**
  - Used to organize analysis and reports / output
- **OrganisationUnitGroupSet**
  - Set of OrganisationUnitGroups
  - Compulsory and / or exclusive

<table>
<thead>
<tr>
<th>ORGANISATIONUNITGROUPSET</th>
<th>Java-type</th>
<th>Db-attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>organisationUnitGroups</td>
<td>Set&lt;OrganisationUnitGroup&gt;</td>
<td>-</td>
</tr>
<tr>
<td>compulsory</td>
<td>boolean</td>
<td>-</td>
</tr>
<tr>
<td>exclusive</td>
<td>boolean</td>
<td>-</td>
</tr>
</tbody>
</table>
Domain model

• Source
  – Generic in order to allow any type of Sources
  – Current source in DHIS 2 is OrganisationUnit
  – Abstract class - OrganisationUnit extends Source

<table>
<thead>
<tr>
<th>SOURCE</th>
<th>Java-type</th>
<th>Db-attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>unique, not-null</td>
</tr>
</tbody>
</table>
## Domain model

- **DataValue**
  - Captured data for a combination of DataElement, Period, and Source

<table>
<thead>
<tr>
<th>DATAVALUE</th>
<th>Java-type</th>
<th>Db-attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataElement</td>
<td>DataElement</td>
<td>-</td>
</tr>
<tr>
<td>period</td>
<td>Period</td>
<td>-</td>
</tr>
<tr>
<td>source</td>
<td>Source</td>
<td>-</td>
</tr>
<tr>
<td>value</td>
<td>String</td>
<td>-</td>
</tr>
<tr>
<td>storedBy</td>
<td>String</td>
<td>-</td>
</tr>
<tr>
<td>timeStamp</td>
<td>Date</td>
<td>-</td>
</tr>
<tr>
<td>comment</td>
<td>String</td>
<td>-</td>
</tr>
</tbody>
</table>
Domain model

• DataValue associations
Domain model

• Indicator
  – Formula based on DataElements
  – Used to improve data analysis
  – Example: BCG under 1 year coverage = BCG doses under 1 year / Total population under 1 year
  – Multiplied with a factor (IndicatorType)

• Indicator associations
## Domain model

<table>
<thead>
<tr>
<th>INDICATOR</th>
<th>Java-type</th>
<th>Db-attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>uuid</td>
<td>string</td>
<td>unique</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>alternativeName</td>
<td>string</td>
<td>unique</td>
</tr>
<tr>
<td>shortName</td>
<td>string</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>code</td>
<td>string</td>
<td>unique</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>indicatorType</td>
<td>IndicatorType</td>
<td>-</td>
</tr>
<tr>
<td>numerator</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>numeratorDescription</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>numeratorAggregationOperator</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>denominator</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>denominatorDescription</td>
<td>string</td>
<td>-</td>
</tr>
<tr>
<td>denominatorAggregationOperator</td>
<td>string</td>
<td>-</td>
</tr>
</tbody>
</table>
Domain model

- **IndicatorType**
  - Example: *Rate (1), Percent (100), Per thousand (1000)*

- **IndicatorGroup**
  - Used to organize analysis and reports / output

<table>
<thead>
<tr>
<th>INDICATORTYPE</th>
<th>Java-type</th>
<th>Db-attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>factor</td>
<td>int</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>INDICATORGROUP</th>
<th>Java-type</th>
<th>Db-attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>uuid</td>
<td>string</td>
<td>unique</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>members</td>
<td>Set&lt;Indicator&gt;</td>
<td>-</td>
</tr>
</tbody>
</table>
Domain model

- **DataSet**
  - Set of data elements
  - Used to organize data entry / input

<table>
<thead>
<tr>
<th>DATASET</th>
<th>Java-type</th>
<th>Db-attr</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>unique, not-null</td>
</tr>
<tr>
<td>periodType</td>
<td>PeriodType</td>
<td>-</td>
</tr>
<tr>
<td>dataElements</td>
<td>Collection&lt;DataElement&gt;</td>
<td>-</td>
</tr>
</tbody>
</table>
Domain model

- DataSet associations
**Project Structure Core**

- **Service POM.**
  - Root as parent,
  - contains project aggregation for service modules

- **Support POM.**
  - Root as parent,
  - contains project aggregation for support modules

- **Root POM.**
  - Contains dependency management and project aggregation.

Building root POM will build all projects except dhis-web

All projects follow the Maven standard directory layout
Project Structure Web

Web maintenance POM. Web root as parent. Contains project aggregation.

Web root POM. Root as parent. Contains dependency management and project aggregation.

Building web root POM will build all web projects
Application design

- Struts 2
- Logic
- Hibernate
- Relational database

Presentation layer → Service layer → Store layer

Method invocation

ANSI SQL
Store layer

• One *store* per domain object
• Performs CRUD operations
• Hibernate implementations of each store
• All stores have dependencies on the *LocalSessionFactoryBean* (Spring)
Store layer overview

- Core module
  - DataElementStore
  - DataSetStore
  - SourceStore
  - DataValueStore
  - ExpressionStore
  - PeriodStore
  - IndicatorStore

- OrgUnit module
  - OrgUnitStore
  - OrgUnitGroupStore
  - DataMartStore
  - DataMartStore
  - UserStore
  - ReportTableStore

- Reporting module
  - User module
Hibernate configuration

- ConfigurationProvider looks for `hibernate.properties`:
  - Inside dhis-support-hibernate (hibernate-default.properties)
  - On the classpath (src/main/resources) in all projects
  - In DHIS2_HOME directory (env variable)
- The last file gets precedence
- Separate properties file for testing
  - hibernate-test.properties
- PostgreSQL configuration:
  ```
  hibernate.dialect = org.hibernate.dialect.PostgreSQLDialect
  hibernate.connection.driver_class = org.postgresql.Driver
  hibernate.connection.url = jdbc:postgresql:dhis2
  hibernate.connection.username = dhis
  hibernate.connection.password = dhis
  ```

Create file hibernate.properties and put in DHIS2_HOME folder for configuration to take effect

Database must be created manually
Service layer

- Service functionality:
  - Administration
  - Settings
  - Datamart
  - Import and export
  - GIS / mapping
  - Reporting
  - User management
  - Validation
Service layer overview

- DataElementService
- DataSetService
- IndicatorService
- DataValueService
- ExpressionService
- PeriodService
- OrgUnitService
- OrgUnitGroupService
- OrgUnit module
- CurrentUserService
- UserSettingService
- User module
- DataMartService
- DataMart module
- AggregationService
- Aggregation module
Testing

Provides supportive methods for object creation, dates etc.

Starts and caches a Spring ApplicationContext and a Hibernate session between tests + Empties database with SQL

Starts and caches a Spring Application context and a Hibernate session + transaction between tests

(extends)

DhisConvenienceTest

(extends)

DhisSpringTest

(extends)

DhisTest

DataSetServiceTest

DataMartServiceTest
Presentation layer

- *Struts 2* used as web framework
- Web modules can run independently or in the *web portal*
- The web portal assembles all web modules (WAR-files) into a complete application
- Common web functionality located in *dhis-web-commons*
- Common web resources located in *dhis-web-commons-resources* (css, javascript, templates)
Support projects

• **DHIS Support External**
  - Configuration manager
    • Write and read configuration objects
  - Location manager
    • Get file or stream for reading
    • Get file or stream for writing
    • Relative to the DHIS2_HOME environment variable

• **DHIS Support System**
  - Support functionality for processes, object deletion, startup routines and more
  - Util classes for codecs, dates, math, PDF, text and more
Installing DHIS 2

• Live package (all-in-one)
  – Embedded servlet container (Jetty)
  – Embedded database (H2)
  – Unzip and invoke exe file / sh script

• Manual installation
  – Install DBMS (PostgreSQL, MySQL)
  – Configuration setup
  – Install servlet container (Tomcat, Jetty)
  – Deploy DHIS 2 WAR file
Collaboration tools

• DHIS 2 on Launchpad
  – launchpad.net/dhis2
  – Check out with Bazaar with lp:dhis2

• Developer mailing list
  – Subscribe: launchpad.net/~dhis2-devs
  – Post: dhis2-devs@lists.launchpad.net

• DHIS 2 site
  – dhis2.org
  – Documentation, development, downloads, features, demo
Development conventions

• Follow the code style

• Do not break the layered architecture
  – Stores for persistence
  – Services for business logic
  – Web modules for GUI

• Write informative commit messages
Development tips

• Create *working sets* in Eclipse to keep track of projects

• Use Maven Jetty plugin during web development
  – Invoke with: $ mvn jetty:run-war on WAR projects

• Speed up building with Maven:
  – Without tests: $ mvn install –DskipTests=true
  – One test among others: $ mvn test –Dtest=NameOfTestClass*