ETRACS

Intermediate Systems Training

July 17 – 21 , 2023

One Central Hotel
Cebu City Philippines

Objectives

- Design and develop a simple "**plugin**" using the Osiris3 Client Platform, SETI Framework and Osiris3 Server

Course Coverage

- Osiris3 Client Platform Programming
- Programming with SETI Framework
- Osiris3 Server Programming

Software Requirements

- NetBeans 7.2
- Java Version 1.8
- Groovy 1.6.2
- IReport-3.0.0
- SQLYog
- VSCode or SublimeText
- ETRACS Training Server 2.5.05.02

Simplified Development Process

- System Analysis and Design
  - Requirements Analysis
  - Database Design
- Server Development
  - Define **schema**
  - Create **services**
- Client Development
  - Build user interfaces
  - Integrate with server code

Violation Plugin

The **violation-plugin**

- Use to demonstrate the concepts, tools and facilities of the Osiris Client Platform and SETI Framework
- Not a full-blown plugin. Could be developed and enhanced

Specifications

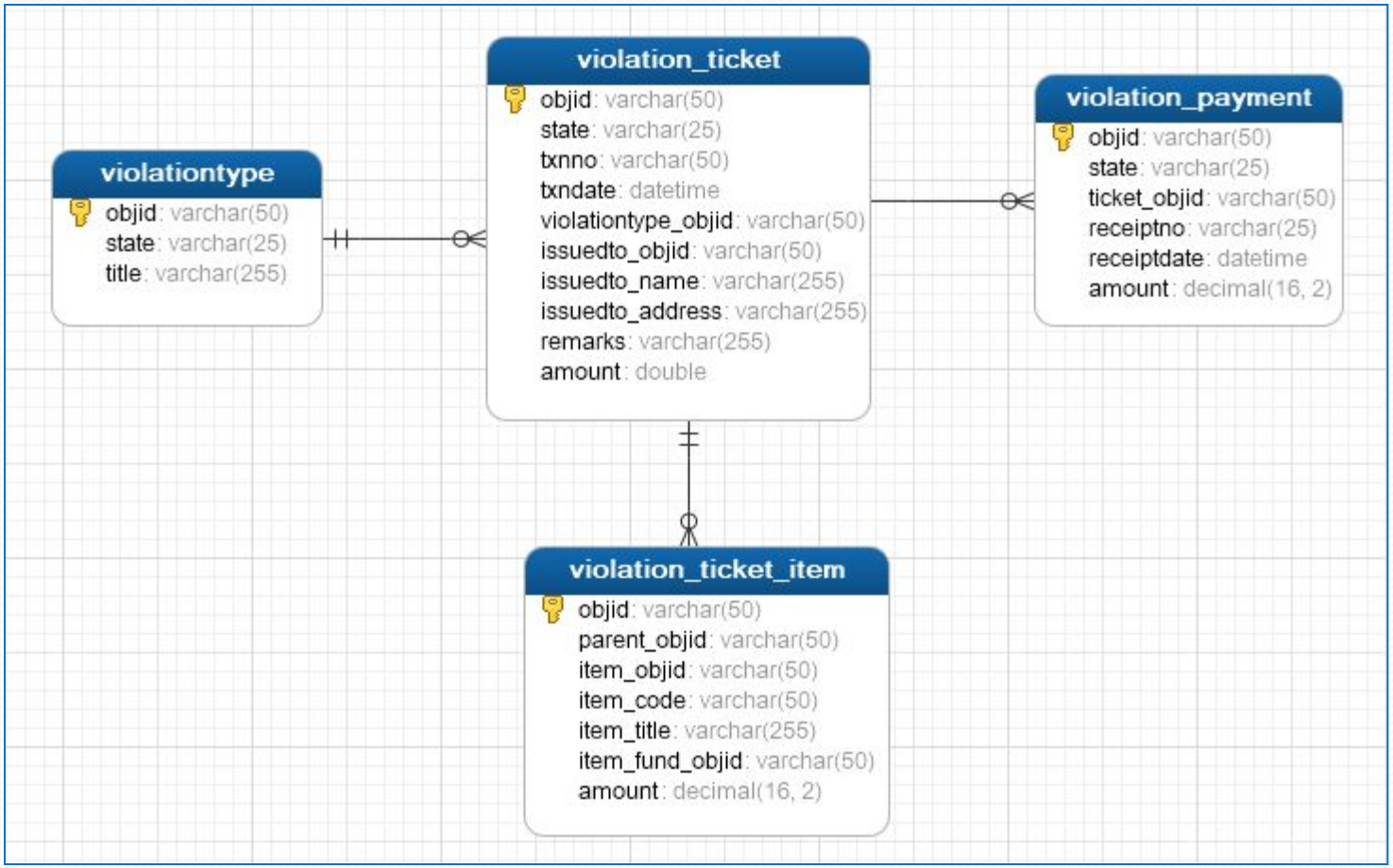
- Master Data
  - Violation Type e.g. Traffic, Ordinance, Anti-Smoking, etc

- Violation Ticket
  - Create new Violation Ticket
  - Print Violation Ticket
  - List Violation Tickets
  - Pay Violation Ticket

- Report
  - List of Violation Tickets

Database Design

ERD


violation_ticket
objid: varchar(50)
state: varchar(25)
txnno: varchar(50)
txndate: datetime
violationtype_objid: varchar(50)
issuedto_objid: varchar(50)
issuedto_name: varchar(255)
issuedto_address: varchar(255)
remarks: varchar(255)
amount: double
violationtype
objid: varchar(50)
state: varchar(25)
title: varchar(255)
violation_payment
objid: varchar(50)
state: varchar(25)
ticket_objid: varchar(50)
receiptno: varchar(25)
receiptdate: datetime
amount: decimal(16, 2)
violation_ticket_item
objid: varchar(50)
parent_objid: varchar(50)
item_objid: varchar(50)
item_code: varchar(50)
item_title: varchar(255)
item_fund_objid: varchar(50)
amount: decimal(16, 2)

The **violation** Database

1. Open SQLYog or Navicat
2. Create new **violation** database
3. Add the following tables
   - violation_type
   - violation_ticket
   - violation_ticket_item
   - violation_payment

- NOTE: Table specifications are on the next slides

The **violationtype** Table

Fields | Indexes | Foreign Keys | Triggers | Options | Comment | SQL Preview

Name	Type	Length	Decimals	Not null	
objid	varchar	50	0	☑	🔑 1
state	varchar	25	0	☑	
title	varchar	255	0	☑	

The **violation_ticket** Table

Fields | Indexes | Foreign Keys | Triggers | Options | Comment | SQL Preview

Name	Type	Length	Decimals	Not null	
objid	varchar	50	0	☑	1
state	varchar	25	0	☑	
txnno	varchar	50	0	☑	
txndate	datetime	0	0	☑	
violationtype_objid	varchar	50	0	☑	
issuedto_objid	varchar	50	0	☑	
issuedto_name	varchar	255	0	☑	
issuedto_address	varchar	255	0	☑	
remarks	varchar	255	0	☐	
amount	double	16	2	☑	

The **violation_ticket_item** Table

Fields | Indexes | Foreign Keys | Triggers | Options | Comment | SQL Preview

Name	Type	Length	Decimals	Not null	
objid	varchar	50	0	☑	1
parent_objid	varchar	50	0	☑	
item_objid	varchar	50	0	☑	
item_code	varchar	50	0	☑	
item_title	varchar	255	0	☑	
item_fund_objid	varchar	50	0	☑	
amount	decimal	16	2	☑	

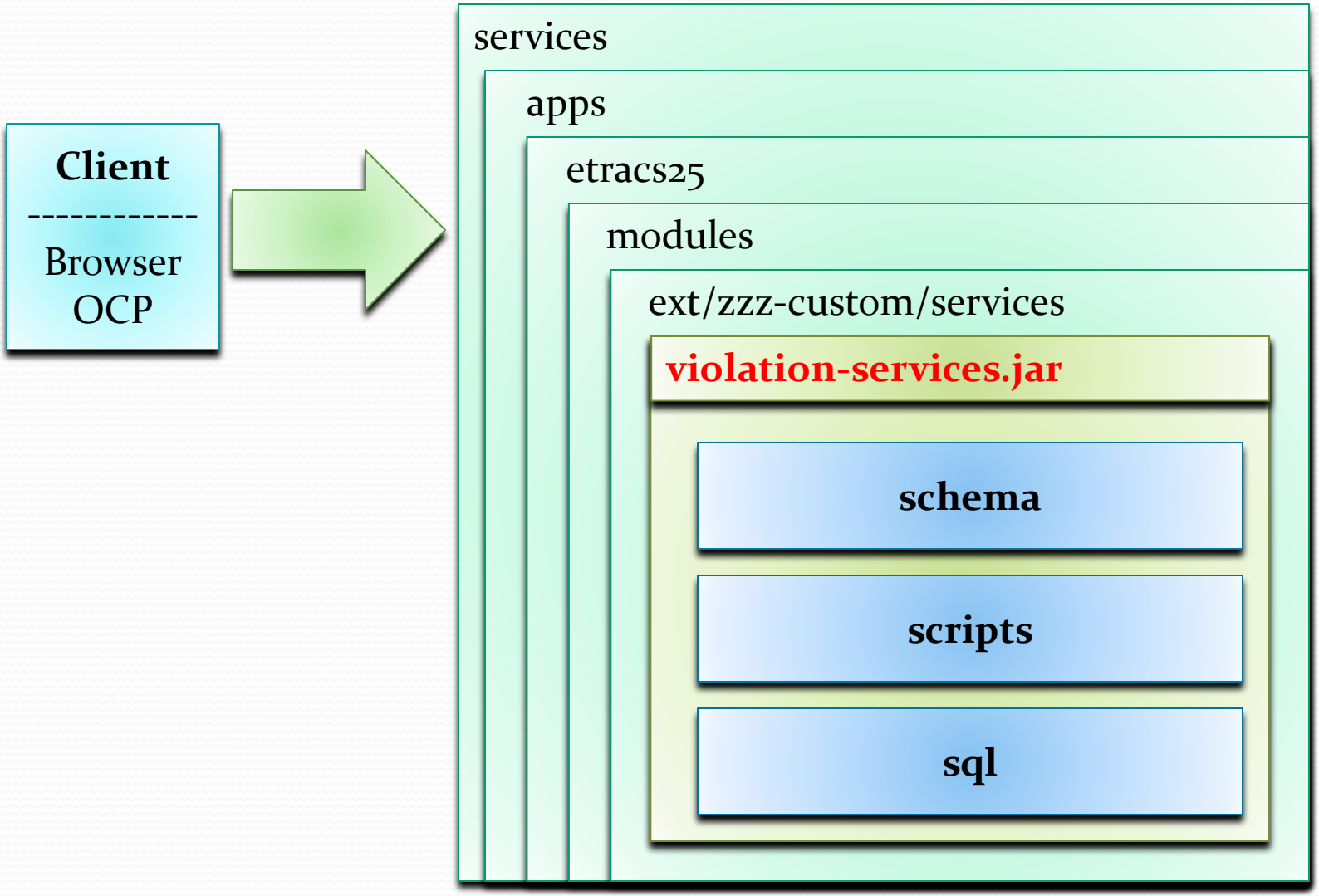
The **violation_payment** Table

Fields | Indexes | Foreign Keys | Triggers | Options | Comment | SQL Preview

Name	Type	Length	Decimals	Not null	
objid	varchar	50	0	☑	1
state	varchar	25	0	☑	
ticket_objid	varchar	50	0	☑	
receiptno	varchar	25	0	☑	
receiptdate	datetime	0	0	☑	
amount	decimal	16	2	☑	

Osiris3 Server Programming

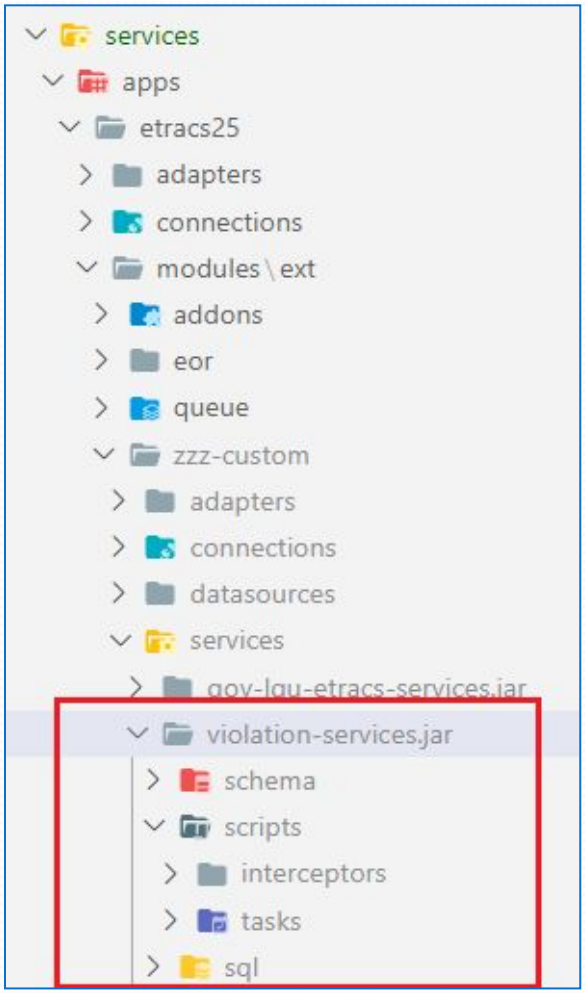
Osiris3 Service Structure



Service Structure

- The service folder must end with "**.jar**" extension
- It can contain the following folders:
  - **schema** – defines data/table definition
  - **scripts** – contain business logic codes
    - **gdx** – handles remote request
    - **interceptors** –code that intercepts a service method
    - **messaging** –handles messaging information
    - **tasks** –codes that are executed based on schedule
  - **sql** – contains SQL statements

The **violation-services.jar**



The **violation-services.jar**

- Create the **violation-services.jar** folder structure

1. Open TRAINING_SERVER in VSCode
2. Navicate to **services/apps/etracs25/modules/ext/zzz-custom /services** folder
3. Right-click **services**, select New Folder…, enter **violation-services.jar** as folder name and press Enter to commit
4. Add **schema**, **scripts** and **sql** folders under violation_services.jar

Data Sources

- Represent a source of data such as database, file etc
- One configuration setting per data source

Register **violationds**

1. Open **SERVER_DIR/services/datasources** folder
2. Select **etracsds** and press CTRL+C to copy.
3. Select **SERVER_DIR/services/apps/etracs25/modules/ext/zzz-custom/datasources** folder
4. Press CTRL + V to paste the etracsds file.
5. Right-click **etracsds** file, select Rename… and change name to **violationds**
6. Edit **v**

```
dialect=mysql
driverClass=com.mysql.jdbc.Driver
url=jdbc:mysql://${dbserver_host}/${db_violation}
user=${dbserver_user}
pwd=${dbserver_pass}
```

7. Save and close file.
8. Edit **SERVER_DIR/bin/env.conf** add **db_violation** option and then Save

```
db_epayment=eor
db_queue=queue
db_violation=violation
```

Adapters

- Represents a **connection** to the datasource
- Should be defined per plugin

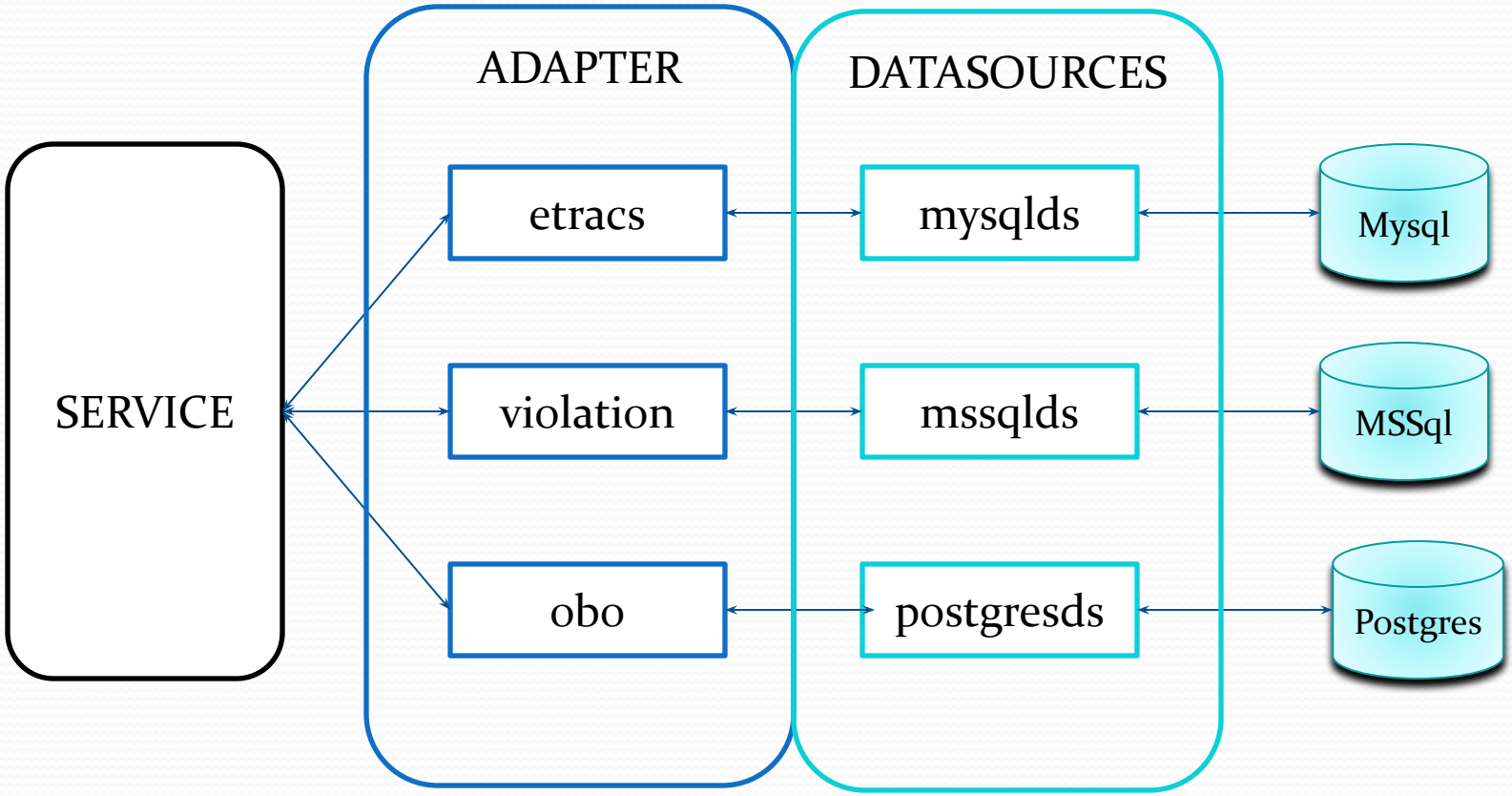
Add **violation** Adapter

1. Open **SERVER_DIR/services/apps/etracs25/modules/ext/zzz-custom/adapters** folder
2. Create a new text file **violation** with no extension
3. Edit **violation** file and add **dsname** option

```
dsname=violationds
```

4. Save and close file.

Datasource and Adapter Relationship

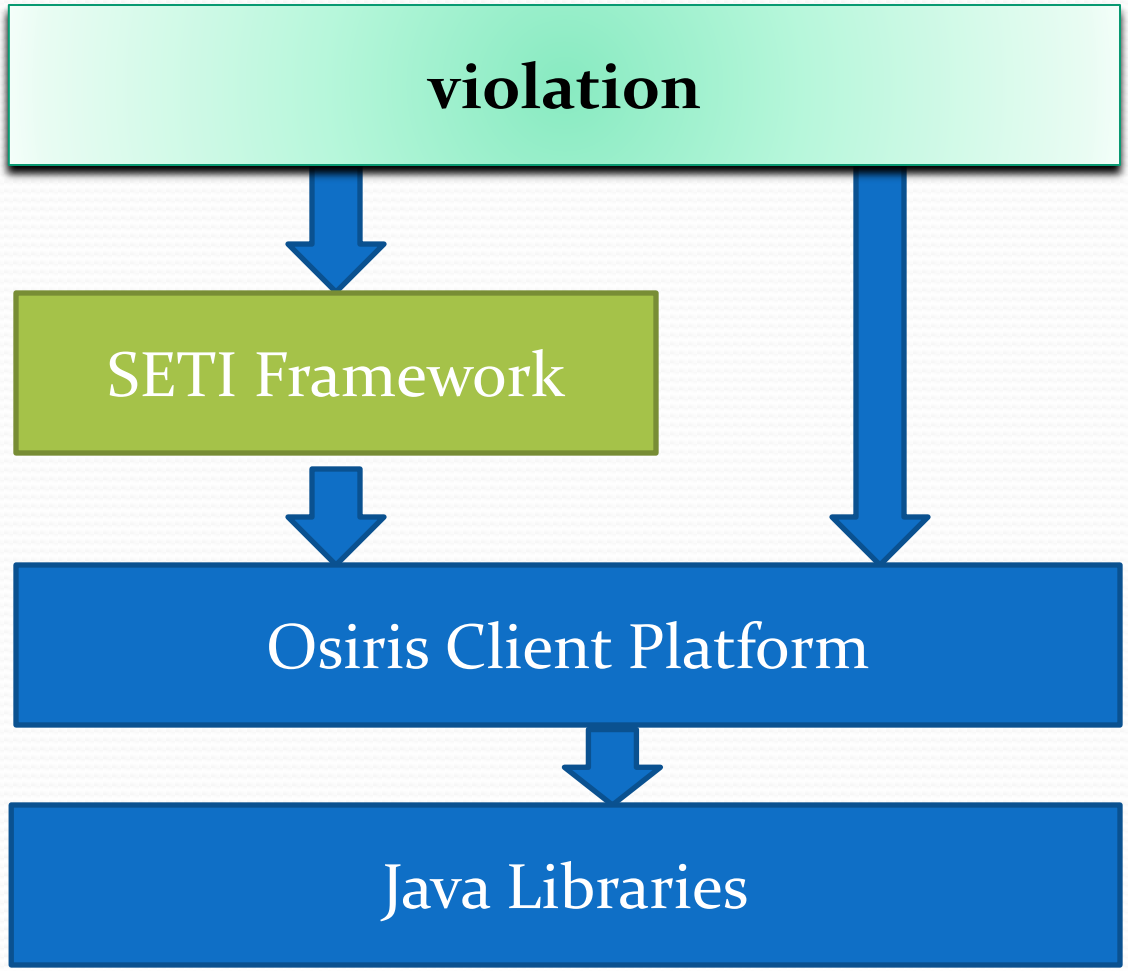


Rameses SETI

Rameses SETI

- Framework to simplify persistence and retrieval of database information
- Designed to be database agnostic
- Encourages Non-SQL coding to access databases
- Schema-based
- Tight integration between server and client for rapid application development
- Designed to handle CRUD documents and listing

Osiris Client Platform



Schema Definition

Schema

- Schema must be per table
- There should be **one primary** field
- Currently supported types:
  - string, decimal, integer, boolean (true/false)
  - date, timestamp
- Structure
  - schema
    - element
      - field
      - complex
        - key
- Additional Attributes (used by tables)
  - caption, searchable, indexed

Example Schema

```
1   <schema>
2     <element tablename="tablename">
3       <field name="pkfield" primary="true"/>
4       <field name="field1" required="true" />
5       <field name="field2" type="decimal" required="true"/>
6       <field name="field3" type="integer" required="true"/>
7       <field name="field4" type="boolean" />
8       <field name="field5" type="date" />
9       <complex name="field6" ref="refschema" jointype="many-to-one">
10        <key field="field1" target="pkfield" />
11      </complex>
12    </element>
13  </schema>
```

Sublime Text Shortcuts

- In SQLYog
- 1. desc tablename
- 2. copy the column names

- In Sublime
- 3. paste the copied column names
- 4. ctrl A -> to select all
- 5. shift crtl L -> to activate multi-cursor
- 6. press home -> to move cursor to home position
- 7. press ctrl + right arrow -> move cursor end of fieldname
- 8. press shift + end and delete extra text
- 9. press home

The **violationtype** schema

- On **violation-services.jar/schema**, add a new file **violationtype.xml** and enter the code below

```
1  <schema adapter="violation">
2    <element tablename="violationtype">
3      <field name="objid" primary="true" caption="Code" />
4      <field name="state" caption="State" searchable="true" />
5      <field name="title" caption="Title" searchable="true" />
6    </element>
7  </schema>
```

The **violation_ticket** schema

- On **violation-services.jar/schema**, add a new file **violation_ticket.xml** and enter the code below

```xml
<schema adapter="violation">
  <element tablename="violation_ticket">
    <field name="objid" primary="true" prefix="VT"/>
    <field name="state" caption="State" required="true" searchable="true" />
    <field name="txnno" caption="Txn No." required="true" searchable="true" />
    <field name="txndate" caption="Txn Date" type="data" required="true"  />
    <field name="violationtype_objid" required="true" />
    <field name="issuedto_objid" required="true" />
    <field name="issuedto_name" caption="Issued To" required="true" searchable="true" />
    <field name="issuedto_address" required="true" />
    <field name="remarks" required="true" />
    <field name="amount" caption="Amount" type="decimal" required="true" />
    <complex name="violationtype" ref="violationtype" jointype="many-to-one">
      <key field="violationtype_objid" target="objid" />
    </complex>
  </element>
</schema>
```

The **violation_ticket_item** schema

- On **violation-services.jar/schema**, add a new file **violation_ticket_item.xml** and enter the code

```xml
<schema adapter="violation">
  <element tablename="violation_ticket_item">
    <field name="objid" primary="true" prefix="VTI" />
    <field name="parent_objid" required="true" />
    <field name="item_objid" required="true" />
    <field name="item_code" required="true" />
    <field name="item_title" required="true" />
    <field name="item_fund_objid" required="true" />
    <field name="amount" required="true" type="decimal" />
  </element>
</schema>
```

The **violation_payment** schema

- On **violation-services.jar/schema**, add a new file **violation_payment.xml** and enter the code

```xml
<schema adapter="violation">
  <element tablename="violation_payment">
    <field name="objid" primary="true" prefix="VP" />
    <field name="state" caption="state" required="true" />
    <field name="ticket_objid" required="true" />
    <field name="receiptno" caption="Receipt No." required="true" searchable="true" />
    <field name="receiptdate" caption="Receipt Date" required="true" />
    <field name="amount" caption="Amount" type="decimal" required="true" />
    <complex name="ticket" ref="violation_ticket" jointype="many-to-one">
      <key field="ticket_objid" target="objid" />
    </complex>
  </element>
</schema>
```

The **underscore-dot** Relation

- The **underscore**
  - normally used in database field names such as barangay_code and barangay_name
  - when read by the platform, it will be converted to an "**embedded**" map object
  - when save by the platform, the "**embedded**" map object keys will be converted to **underscore**
- The **dot**
  - use to access or traverse map object keys

- Example

Field Name	**Map Object**
idno	entity.idno
lastname	entity.lastname
barangay_code	entity.barangay.code
barangay_name	entity.barangay.name

Rameses SETI Client

Rameses SETI Client

- **Templates**
  - Generic UI design for specific purpose and can be reused by providing specific behaviours
  - Package: com/rameses/seti2/templates/
  - Commonly used templates
    - CrudList.xml – use to list records
    - CrudNodeList.xml – use to list records with node options
    - CrudForm.xml – basic crud form support
    - CrudLookup.xml – basic lookup support
    - CrudReport.xml – basic report support

Rameses SETI Client

- **Models**
  - Generate model code to support specific templates and can be extended for specific purpose
  - Package: com/rameses/seti2/models/
  - Supported models
    - CrudListModel – support class for list templates
    - CrudFormModel – support class for crud forms
    - CrudLookupModel – support class for crud lookup forms
    - CrudReportModel – support class for crud report forms

The CrudListModel

- **Attributes**
  - schemaName - target schema
  - entitySchemaName – in case schemaName is a custom schema
  - cols – columns to display on the list
  - hiddenCols –included cols by not displayed on the list
  - orderBy – define order by clause
  - surroundSearch – surround searchtex with ‘%’
  - multiSelect – allow multiple record selection
  - windowTitle – the window title
  - title – the form title
  - allowCreate – must be tue to allow create
  - allowFilter – must be tue to allow filter
  - allowPrint – must be tue to allow print
  - allowOpen – must be tue to allow open
  - allowEdit – must be tue to allow edit
  - allowDelete – must be tue to allow delete

The CrudListModel

- **Methods**
  - def getPersistenceService()
  - def getQueryService()
  - void beforeQuery(map)
  - void beforeFetchNodes(map)
  - def getCustomFilter()
  - void beforeInit()
  - void afterInit()
  - def fetchNodeList(map)

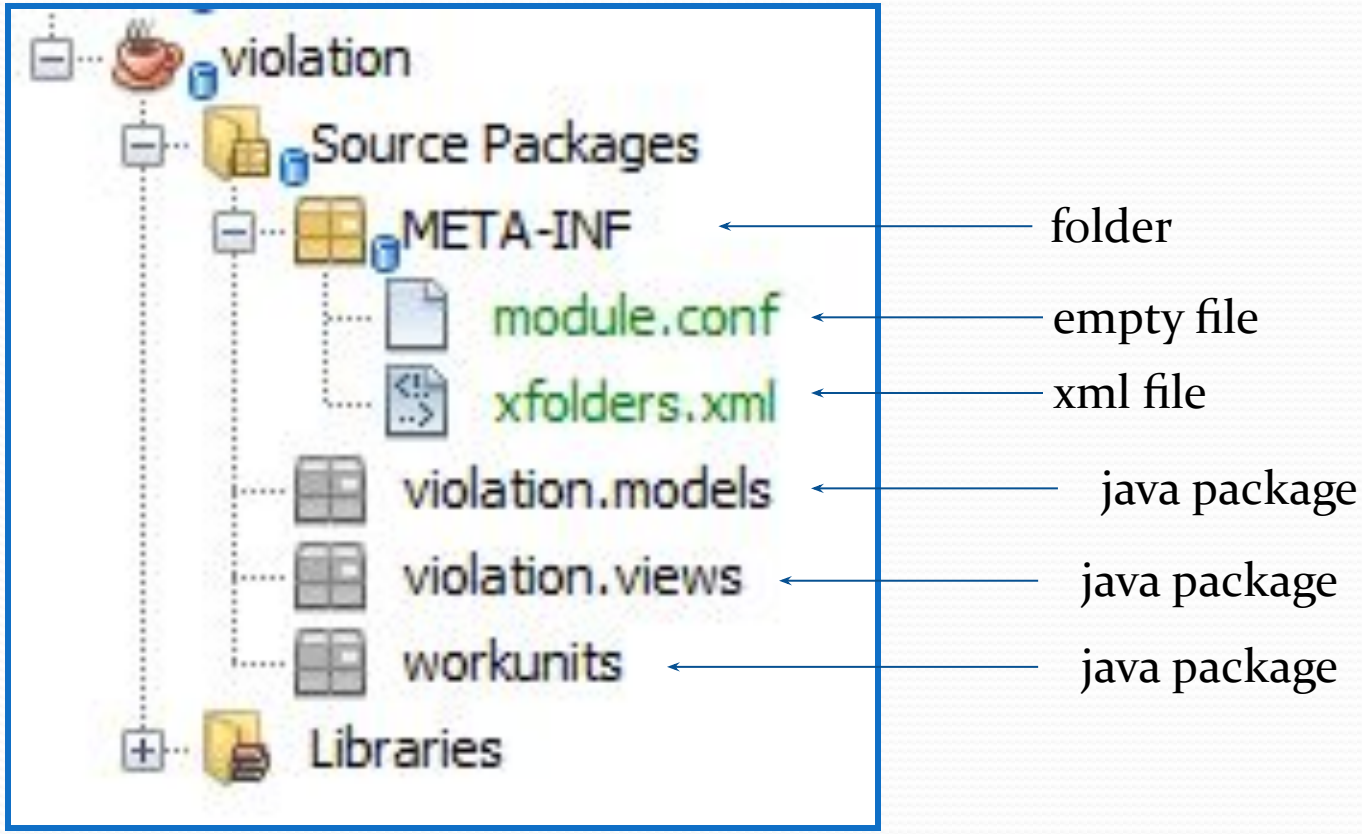
The CrudFormModel

- **Methods**
  - public void afterInit()
  - public void afterCreate()
  - public void beforeOpen()
  - public void afterOpen()
  - public void afterEdit()
  - public void beforeSave(def mode)
  - public void afterSave()
  - public void afterCreateData(String name, def data)
  - public boolean name, item, colName, newItem)
  - public void afterColumnUpdate(name, item, colName)
  - public void beforeAddItem(name, item)
  - public void afterAddItem(name, item)
  - boolean isColumnEditable(name, item, columnName)
  - boolean beforeRemoveItem(name, item)

Violation Plugin Development

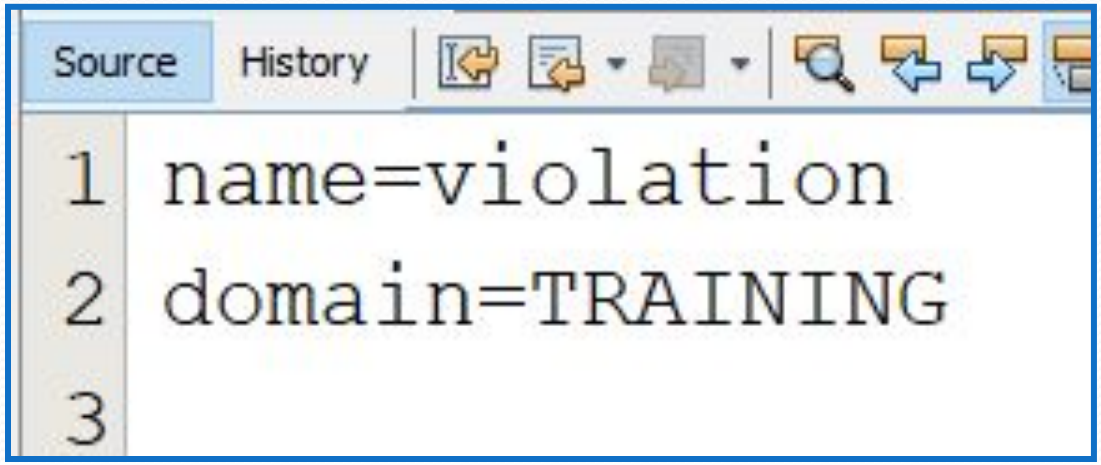
The **violation** project

- Create the **violation** java project.



The module.conf

- Edit **module.conf** and add the following information



The xfolders.xml

- Edit **xfolders.xml** and add the following information

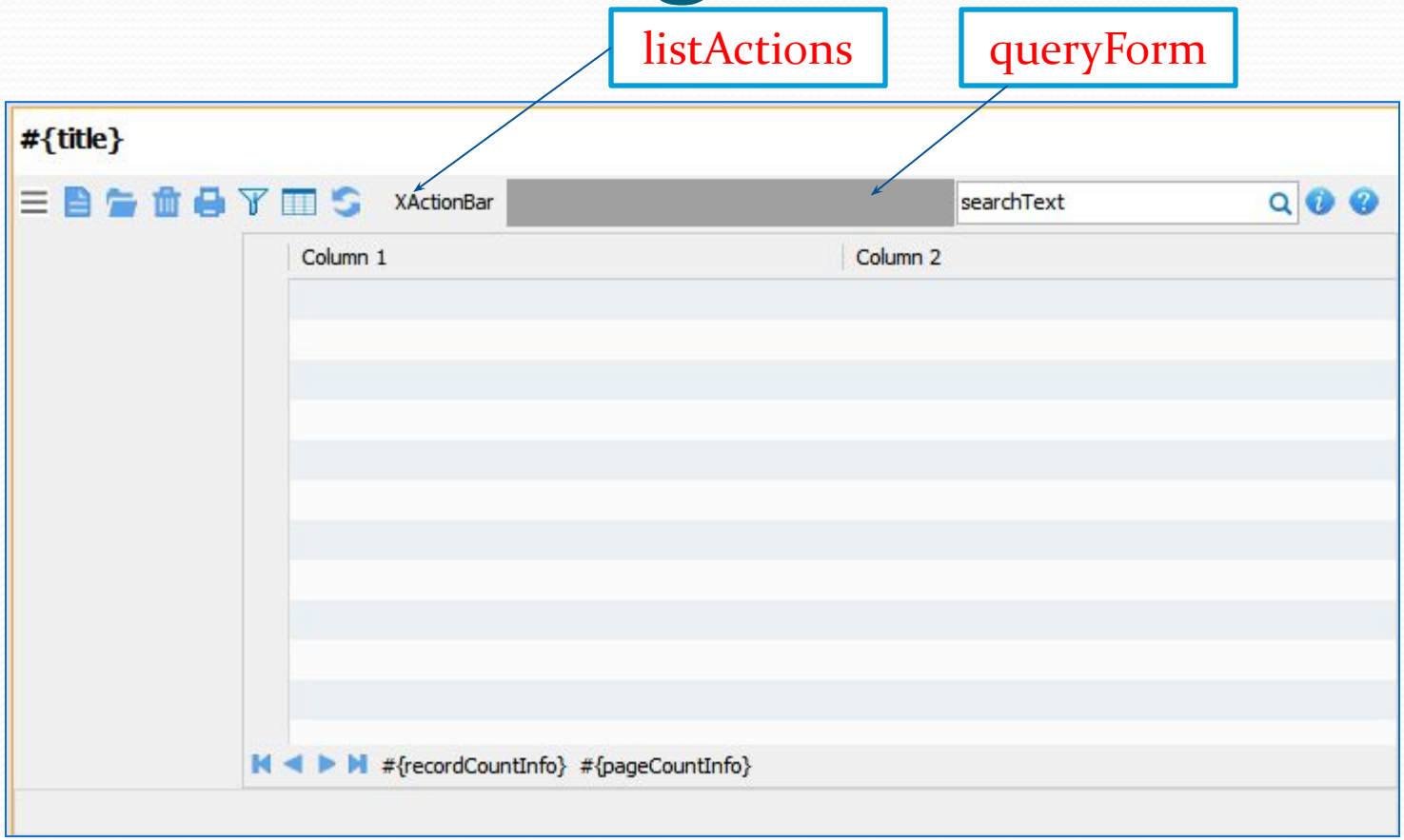
```
1 <folders>
2     <folder id="home">
3         <folder id="violation" caption="Violation" />
4     </folder>
5 </folders>
```

Violation Type Listing

The **CrudList.xml** Template

```xml
<workunit>
    <code class="com.rameses.seti2.models.CrudListModel"/>
    <pages>
        <page template="com.rameses.seti2.views.CrudListPage"/>
    </pages>
</workunit>
```

The **CrudListPage**



The **violationtype_list** workunit

- Extends the **CrudList.xml** for list template
- Minimum required attributes:
  - **extends** – then generic **template** workunit to extend
  - **schemaName** – the target schema
  - **col** - the columns to display on the list

The **violationtype_list** workunit

- Add the **violation_list.xml** and enter the workunit definition

```
1  <workunit
2      extends="com/rameses/seti2/templates/CrudList.xml"
3      schemaName="violationtype"
4      cols="state,objid,title"
5  >
6      <invokers>
7          <invoker folderid="/home/violation"
8              action="init"
9              caption="Violation Types"
10         />
11     </invokers>
12 </workunit>
```

Testing the Plugin

The etracs255-client-tester

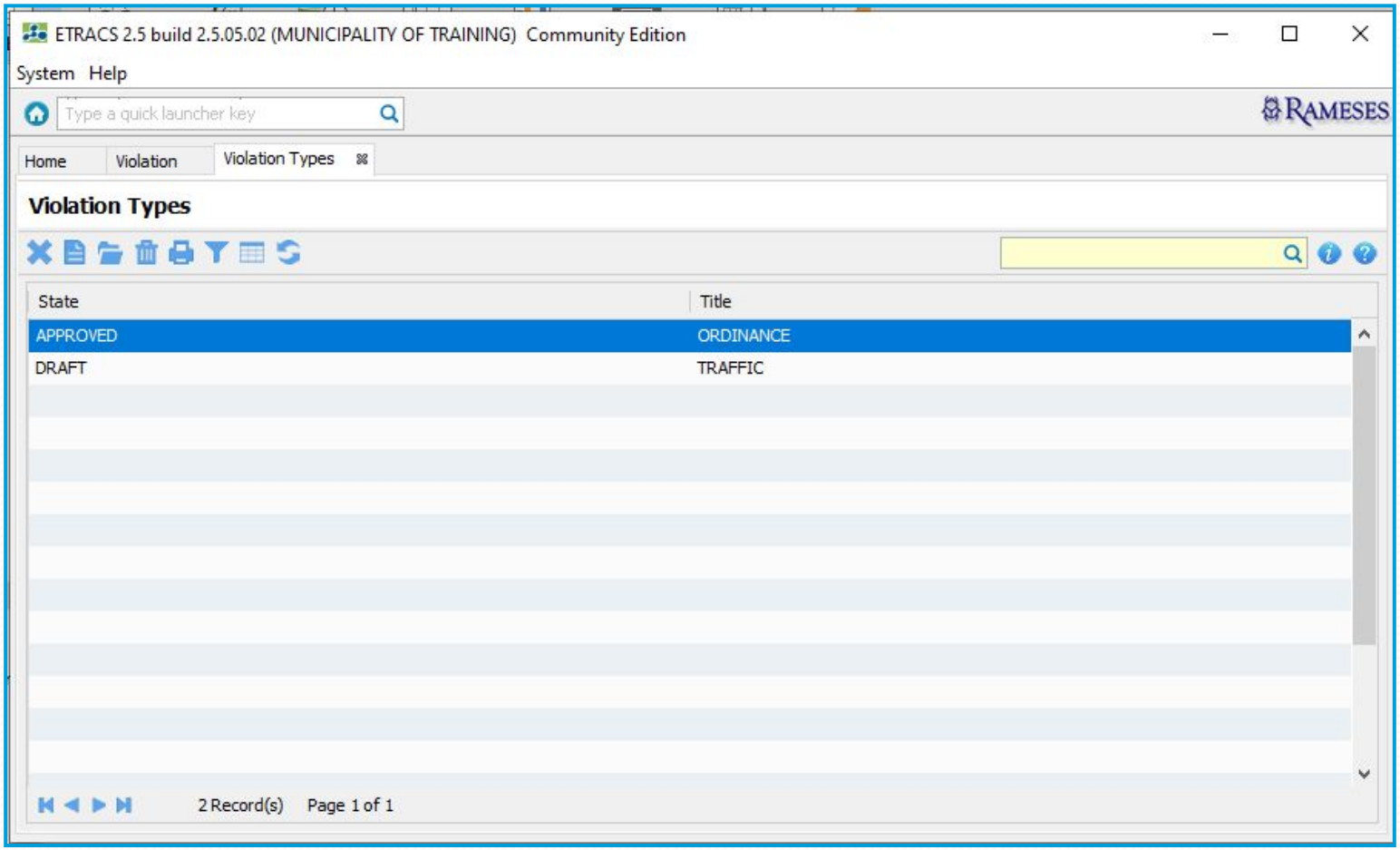
- Simplifies development and testing of plugin
- Does not require deployment to server
- Does not require the Main class.

- Activity:
  1. Right-click on Libraries, select Add Project and add **violation** project

Testing **violationtype** Listing

1. Run ETRACS Server
2. In NetBeans
    1. Clean and Build **violation** project
    2. After the build is complete, run **etracs255-client-tester**

Violation Type Listing

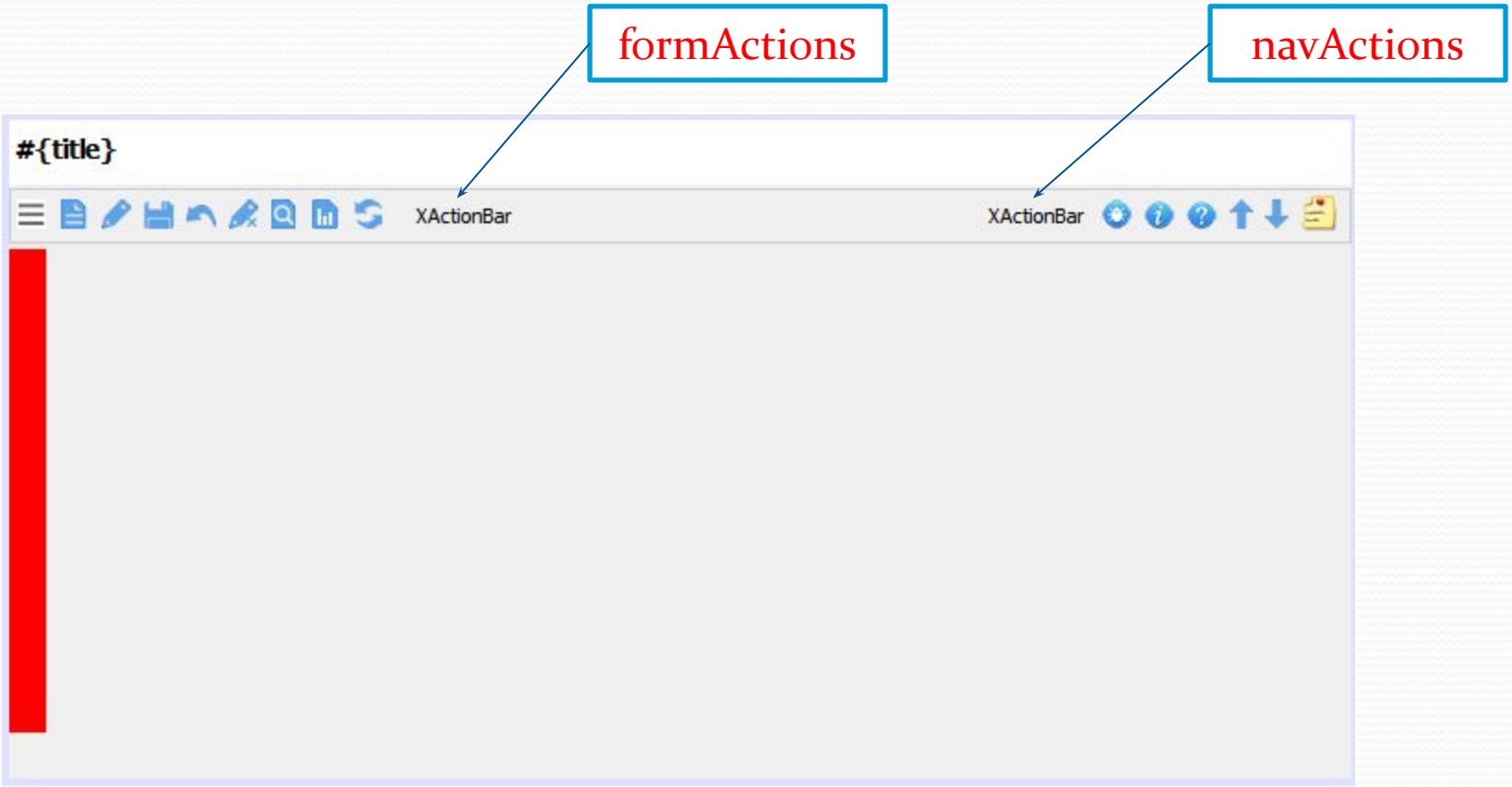


The Violation Type Form

The **CrudForm.xml** Template

```
1  <workunit>
2      <code class="com.rameses.seti2.models.CrudFormModel"/>
3  </workunit>
```

The **CrudFormPage**



The **violationtype** workunit

- Extends the **CrudForm.xml** for basic CRUD interface
- Minimum required attributes:
  - **extends** – the generic workunit to extend
  - **schemaName** – the target schema
- Add invokers for **create** and **open** using the following convention **schemaName:action**
  - Example:
    - violationtype:create
    - violationtype:open

The **violationtype** workunit

- Create the **violationtype** workunit

```
1   <workunit
2       extends="com/rameses/seti2/templates/CrudForm.xml"
3       schemaName="violationtype"
4   >
5       <invokers>
6           <invoker type="violationtype:create"
7               action="create"
8               caption="Violation Type (New)"
9           />
10          <invoker type="violationtype:open"
11              action="open"
12              caption="Violation Type"
13          />
14      </invokers>
15      <pages>
16          <page template="violation.views.ViolationTypePage" />
17      </pages>
18  </workunit>
```

The **ViolationTypePage**

State :	#{entity.state}
Code : *	entity.objid
Title : *	entity.title

The **ViolationTypePage**

```
1   ------------------------------------------------
2   XControl        Property        Value
3   ------------------------------------------------
4   XFormPanel      n/a             n/a
5
6   XLabel          expression      #{entity.state}
7
8   XTextField      name            entity.objid
9                   caption         Code
10                  required        true
11
12  XTextField      name            entity.title
13                  caption         Title
14                  required        true
15                  preferredSize   [0,20]
```

The **ViolationTypePage** Template

- Annotate the page with the **CrudFormPage** template class.
  1. Open the **ViolationTypePage** and click on **Source** button.
  2. Above the public class definition, add the **@Template(CrudFormPage.class)** annotation

```
package violation.views;

import com.rameses.rcp.ui.annotations.Template;
import com.rameses.seti2.views.CrudFormPage;

@Template(CrudFormPage.class)
public class ViolationTypePage extends javax.swi
```

  3. Perform a **Fix Import** to import the required packages

Testing **ViolationType** Master Data

- View violation type listing
- Search Violation Type

- Create Violation Type
- Open Violation Type
- Edit Violation Type
- Delete Violation Type

How it works?

- How were the records being persisted ?
- Why did the search works?

System Services

- **Persistence Service**
  - def create(entity)
  - def read(entity)
  - def update(entity)
  - def removeEntity(entity)
  - def save(entity)
- **QueryService**
  - def findFirst(param)
  - def getList(param)
  - def getNodeList(param)

End of Module