SP Front Office

IP Key LA workshop at INDECOPI, Lima, Peru

Martin Beckman, Emmanuel Collin, Antonio Grau, Arjeton Uzairi
November 2018
What is SP FrontOffice?

SP FrontOffice is a web-based suite of tools that allows clients of the National Offices to:

- easily submit their **trade mark** applications online (TM eFiling module)
- easily submit their **design** applications online (Design eFiling module)
- easily submit their **pre/post registration** procedures online (eServices module)

It is office adaptable and IPO managed!
How is it implemented?

The EUIPO and the EUIPN (European Union Intellectual Property Network) have developed a generic version of the SP FrontOffice. IP offices can implement their own version of the software by following these steps:

1. License agreement is signed between the EUIPO and the IP office.
2. The IP office gets access to the source code of the generic version as well as the documentation.
3. The IP office decides which e-Forms to implement.
4. The IP office customizes their own version of the SP FrontOffice and installs at their own premises.
How many offices implemented so far?

Most of the EU offices have implemented at least one of the modules of the SP FrontOffices suite, including the EUIPO itself.

- **TM eFiling**: 19 national offices in the EU
- **Design eFiling**: 15 national offices in the EU
- **eServices**: 17 national offices in the EU

Additionally, a non/EU IP offices has already implemented the tool as well: The Philippines IP office!

Some examples of FO live at IPOs:
- **EUIPO**: [TM e-filing](#) (wizard form), [TM e-filing](#) (advanced form), [DS e-filing](#)
- **Spain**: [DS e-filing](#), renewal, change, transfer
- **Philippines**: [TM e-filing](#)
## Functionalities and integrations

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Related Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>TM basic information</td>
<td>Retrieve similar TM details</td>
<td>TMview / IPO BO DB</td>
</tr>
<tr>
<td>Nice classification</td>
<td>Search and retrieve terms, automatic validation</td>
<td>TMclass</td>
</tr>
<tr>
<td>Persons</td>
<td>Retrieve applicant and representative details</td>
<td>TMview / IPO BO DB / National registry</td>
</tr>
<tr>
<td>Claims</td>
<td>Different types of claims</td>
<td>TMview / IPO BO DB</td>
</tr>
<tr>
<td>Pre Clearance report</td>
<td>Prevent potential future oppositions</td>
<td>TMview / IPO BO DB</td>
</tr>
<tr>
<td>Fees</td>
<td>Automatic calculation</td>
<td>Payment system</td>
</tr>
<tr>
<td>Log in</td>
<td>Access to user area</td>
<td>IPO Users DB</td>
</tr>
<tr>
<td>User interface</td>
<td>Wizard and one form approach</td>
<td></td>
</tr>
</tbody>
</table>
## Functionalities and integrations

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Integration/Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS basic information</td>
<td>Retrieve similar DS details</td>
<td>DSview / IPO Users DB</td>
</tr>
<tr>
<td>Locarno classification</td>
<td>Search and retrieve product indications</td>
<td>Eurolocarno</td>
</tr>
<tr>
<td>Persons</td>
<td>Retrieve applicant and representative details</td>
<td>DSview / IPO Users DB / National registry</td>
</tr>
<tr>
<td>Claims</td>
<td>Different types of claims</td>
<td>DSview / IPO Users DB</td>
</tr>
<tr>
<td>Fees</td>
<td>Automatic calculation</td>
<td>Payment system</td>
</tr>
<tr>
<td>Log in</td>
<td>Access to user area</td>
<td>IPO Users DB</td>
</tr>
<tr>
<td>User interface</td>
<td>Wizard and one form approach</td>
<td></td>
</tr>
</tbody>
</table>
# Types of e-Services

## Trade marks
- **Renewal**
- **Change owner details**
- **Change Representative/Correspondent**
- **Transfer of rights**
- **Invalidity**
- **Rights in rem**
- **License**
- **Generic**
- **Opposition**
- **Revocation**
- **Objection**
- **Withdrawal / Surrender**
- **Limitation of Goods & Services**

## Designs
- **Renewal**
- **Change owner details**
- **Change Representative/Correspondent**
- **Transfer of rights**
- **Invalidity**
- **Rights in rem**
- **License**
- **Generic**
All the SP Modules must be implemented?

The SP FrontOffice allows the implementation of the specific eForms required by the IP Office. Therefore, it is not required to implement the whole suit.

Then... how is the tool customized?

1. The Office analyses their needs and decides which eForms must be implemented.
   - The *FrontOffice questionnaire* can help to define these needs.

2. The Business experts of the Office define and document the business requirements for each eForm.
   - This is done filling the *Information Model* documents which are the best way to gather the requirements and pass them to the IT team.
Front Office configuration
Types of configuration

- Installation & Deployment Configuration
- Business Configuration
- User Interface Configuration
- General Configuration

Types of System Configuration
Installation & deployment configuration settings

- Also referred to as system settings: settings via which an administrator can configure the various installation and deployment aspects of the application

- Configuration is achieved through XML and properties files residing in externalized locations
  - Files specific to the servers used
  - Additional properties files specific to Software Package

- Changes of these settings require a restart of the affected component(s)

- Information is documented in the Launch Plan (SP-FO-1.0.0-LPN-FULL-EN.docx)

- Examples:
  - Logging (what to log, where to log)
  - JNDI names for communication between server nodes
  - Connection to DB and Document Repository
  - URLs of services exposed by the legacy systems
  - Server optimization settings (memory, etc.)
User Interface configuration settings

- Settings that can modify the Look & Feel of the system

- Accomplished through the use of CSS files and properties files
  - Standard web resource files: HTML, CSS, images, fonts, etc.
  - Additional properties files specific to Software Package

- In most cases modifications of those parameters will require a restart of the affected component(s)

- Examples:
  - Localised labels and messages
  - Modification of fonts, logos, etc.
  - Modification of colours and stylesheets

Most of the configuration in this area is standard User Interface development (CSS, logo, etc.)
Business configuration settings

- Settings that can modify business-wise the behaviour of the application

- Accomplished through the use of XML files and Business Rules

- In most cases modifications of those parameters will require a restart of the affected component(s)

- Example:
  - Form pages, sections and fields (show/hide, optional/mandatory, etc.)
  - Content of lists (countries, languages, etc.)
  - Data validation (business rules)
  - Fees management (fee amounts, algorithm for calculating fees)
  - Receipt (format and content)
  - Etc.
General configuration settings

✓ Settings that are used for configuring other aspects of the application not directly related to business, Look & Feel or installation settings aforementioned

✓ Accomplished through XML files and parameters stored in the database

✓ Depending on the parameters touched changes are applied either on-the-fly or might require a restart of the affected component(s)

✓ Examples:
  ✓ Enabling/disabling invocations to external services
  ✓ File upload options (formats, number of attachments, maximum size, etc.)
  ✓ User permissions (rights assigned to roles)
  ✓ Number of search results displayed
  ✓ Maximum number of allowed applicants, representatives, designs, etc.
  ✓ Etc.
Front Office architecture overview
Reference architecture

Presentation Tier
- UI / Consumers

Service Tier
- Processes Layer
- Service Layer
- Service Component Layer

Information Tier
- Applications, Data Sources
  - Servers
  - DB

FRONT OFFICE

BACK OFFICE
Reference architecture

**Presentation Tier**
- Includes the consumer applications that implement the interface toward users (UI)
- Standard MVC Web Application that control the interaction with the user and consumes “services” from the underlying layer

**Service Tier**
- Encapsulates the company business logic and business rules into standardized services (service layer)
- Includes the *service components* implementing the services interfaces (service component layer)
- For the Back Office also includes long-running processes orchestrating the services (process layer)

**Information Tier**
- Includes the data sources and the applications (custom, COTS, legacy) available in the participating office context.
This view is illustrative, it does not show the actual UI modules, Core Components and Adapters implemented or used by each SP project.
Front Office software artefacts deployed to single server instance

- Low complexity
- Minimum servers required
- High availability only requires that a single server is clustered
- A single log file can be monitored which will provide a complete view of the system
- Reduction in *over the wire* network traffic due to applications being contained within the same container

---

**Front Office – Deployment View**

- Users
- Presentation
- Service
- Information

**Web Browser** ➔ **Web Server** ➔ **Application Server**

**Zone 1**

**Zone 2**

**Internet** ➔ **DMZ**

**PO systems**

**RDBMS**

**Document Repository**
Front Office – Transformation between domains

UI

UIDomain
common-ui/Component/UIDomain
+ specific classes in tm-eFiling-ui/flow, ds-eFiling-ui/domain and eservices-ui/ domain
  - It contains forms and POJOs used by the UI components

Core

CoreDomain
core/CoreDomain
  - Main application domain used in business services

Integration

ExternalDomain
external/ExternalDomain
  - Used for the communication of core with external services
  - Auto-generated from XSDs in external/WebServicesSchemas
  - Default conventions to map to core

External Systems

Other sources like tmview, tmclass etc
The sources could be in several formats (custom transformation classes have to be implemented) but current implementations use smooks to transform from XML to ExternalDomain objects

FilingDomain

core/FilingDomain
  - Used to generate the application.xml
  - Auto-generated from SP-DS-eFiling-V2-00.xsd and SP-TM-eFiling-V2-00.xsd
  - Dozer mapping files for tm/ds dozerMapping.xml

Dozer

Tradeoff classes
Development
## Development: Tools

<table>
<thead>
<tr>
<th>Technology</th>
<th>Description</th>
<th>Additional Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java</td>
<td>Java SE Development Kit 7 (7u60+)</td>
<td><a href="http://www.oracle.com/technetwork/java/index.html">http://www.oracle.com/technetwork/java/index.html</a></td>
</tr>
<tr>
<td></td>
<td>Java EE 6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>a project object model (POM), Maven can manage a project's build, reporting</td>
<td></td>
</tr>
<tr>
<td></td>
<td>and documentation from a central piece of information.</td>
<td></td>
</tr>
<tr>
<td>Gradle</td>
<td>A tool for software project build, Gradle is used in the context of</td>
<td><a href="http://gradle.org/">http://gradle.org/</a></td>
</tr>
<tr>
<td></td>
<td>Software Package Front Office to manage deployment of the built artefacts</td>
<td></td>
</tr>
<tr>
<td></td>
<td>to the various servers.</td>
<td></td>
</tr>
</tbody>
</table>

- IDEs: Eclipse, IntelliJ IDEA, etc.
- Frameworks: JQuery, Spring (MVC, Core, Web Flow, Security), Hibernate, Drools
- Other tools: Anypoint Studio (or Eclipse plug-in), SoapUI
- Source Control: Subversion
Development: Modules and components

Modules
✓ Map to high level source code structure in SVN
✓ Group components according to either:
  ✓ Functional areas (presentation tier)
  ✓ Layers (service tier)
  ✓ Others (installation/documentation)

Components
✓ Map to functional elements/areas (UI)
✓ Define second level elements in the source code structure
✓ Group business/technical services

Naming conventions
✓ UI layer components: take name from its functional areas
✓ Core layer components: named after each functional element
✓ External layer: functional element, plus “Consumer”
✓ Integration layer components: functional element, plus either “Provider” or “Adapter” (consumed by SP)
Development: Presentation Tier

**tm-efiling, ds-efiling and eservices**
- Code and resources of the Presentation Tier modules
- Artefacts are built and deployed as WAR files
- Static resources are deployed to Apache HTTP Server
- WAR files are deployed to Apache Tomcat Server

**common-ui**
- Provides common, resuable UI components
- Includes the UI Domain and UI Services
- Artefacts are built as JAR files
- Utilised by tm-efiling, ds-efiling and eservices
Development: Service Tier – Core (1/2)

**core**
- Code and resources of the Service Tier components
- Artefacts are built and deployed as either EAR or JAR files
- EAR/JAR files are deployed to JBoss Application Server

**external**
- Code and resources of web service end points (WSDL/XSD) and corresponding Java interfaces
- Includes the components used to perform web services calls (service consumers)
- Artefacts are deployed as part of the core components

**common-utils**
- Provides utility classes utilized by the core module
- Artefacts are built and deployed as JAR files
Development: Service Tier – Core (2/2)

**core**
- CoreDomain and CoreServices provide generic business interfaces
- Components such as PersonSearch and Fee Calculation provide specific implementations of the core services interfaces. These implementations can be customised or created as required by Participating Offices
- FilingDomain defines the filing form objects corresponding to the Application XML (TM and DS schemas)

**external**
- WebServicesSchemas provides the definition of the external Web Services (WSDL and XSD)
- ExternalDomain and ExternalServices provide the Java interfaces corresponding to the WS
- Components such as PersonSearchConsumer manage the web services call logic. They are used by core components that are proxies to external legacy systems
Development: Service Tier – Integration

**integration**
- Code and resources of the Integration Layer within the Service Tier
- Provides integration modules for:
  - Consumed services (xxxAdapter)
  - Provided services (xxxProvider)
- Artefacts are built and deployed as Zip files
- Zip files are deployed to MuleESB server
- Integration with tmdn.org external services such as TMClass and TMview are provided out of the box
- Mock components require specific implementations