Deliverable D200.7

Store component

WP 200

Project Acronym & Number:	FIspace – 604 123
Project Title:	FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics
Funding Scheme:	Collaborative Project - Large-scale Integrated Project (IP)
Latest version of Annex 1:	2013-10-03
Start date of the project:	01.04.2013
Duration:	24
Status:	Draft
Editor:	Said Rahma (ATOS)
Contributors (to the "R" part of the deliverable[1]; ordered by project partner)	**ATOS**: Said Rahma Rodriguez, Roberto Castillo Jimenez **IBM**: Andres Garcia-Garcia
Document Identifier:	FIspace-D200.7-FIspace_Integrated_Release_V3-Store-v0.4.docx
Date:	27.02.2015
Revision:	004
Project website address:	http://www.FIspace.eu

1 Contributors to FIspace code ("P") include ATB, UDE, IBM, ATOS, KOC, TOG, AST, NKUA, UPM and LimeTri; contributing persons are listed at https://bitbucket.org/fispace/profile/members

The FIspace Project

Leveraging on outcomes of two complementary Phase 1 use case projects (FInest & SmartAgriFood), aim of FIspace is to pioneer towards fundamental changes on how collaborative business networks will work in future. FIspace will develop a multi-domain Business Collaboration Space (short: FIspace) that employs FI technologies for enabling seamless collaboration in open, cross-organizational business networks, establish eight working Experimentation Sites in Europe where Pilot Applications are tested in Early Trials for Agri-Food, Transport & Logistics and prepare for industrial uptake by engaging with players & associations from relevant industry sectors and IT industry.

Project Summary

As a use case project in Phase 2 of the FI PPP, FIspace aims at developing and validating novel Future-Internet-enabled solutions to address the pressing challenges arising in collaborative business networks, focussing on use cases from the Agri-Food, Transport and Logistics industries. FIspace will focus on exploiting, incorporating and validating the Generic Enablers provided by the FI PPP Core Platform with the aim of realising an extensible collaboration service for business networks together with a set of innovative test applications that allow for radical improvements in how networked businesses can work in the future. Those solutions will be demonstrated and tested through early trials on experimentation sites across Europe. The project results will be open to the FI PPP program and the general public, and the pro-active engagement of larger user communities and external solution providers will foster innovation and industrial uptake planned for Phase 3 of the FI PPP.

Project Consortium

- DLO; Netherlands
- ATB Bremen; Germany
- IBM; Israel
- KocSistem; Turkey
- Aston University; United Kingdom
- ENoLL; Belgium
- KTBL; Germany
- NKUA; Greece
- Wageningen University; Netherlands
- PlusFresc; Spain
- FloriCode; Netherlands
- Kverneland; Netherlands
- North Sea Container Line; Norway
- LimeTri; Netherlands
- BO-MO; Slovenia
- MOBICS; Greece
- Fraunhofer IML; Germany
- Q-ray; Netherlands
- FINCONS; Italy
- Kühne + Nagel; Switzerland
- University Duisburg Essen; Germany
- ATOS; Spain
- The Open Group; United Kingdom
- CentMa; Germany
- iMinds; Belgium
- Marintek; Norway
- University Politecnica Madrid; Spain
- Arcelik; Turkey
- EuroPoolSystem; Germany
- GS1 Germany; Germany
- Mieloo & Alexander; Netherlands
- OPEKEPE; Greece
- Innovators; Greece
- CIT; Spain
- SDZ; Germany
- Snoopmedia; Germany
- EECC; Germany
- CBT; Spain

More Information

Harald Sundmaeker (coordinator)	e-mail:	sundmaeker@atb-bremen.de
Bert Vermeer (deputy coordinator)	e-mail:	bert.vermeer@wur.nl
Project Website	Web link:	http://www.fispace.eu/

Dissemination Level

PU	Public	
PP	Restricted to other programme participants (including the Commission Services)	**X**
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Change History

Version	**Notes**	**Date**
001	Creation of the document Store component	09.12.2014
002	Update of the contents, update the overview section, add new approach of Store security related to use Keycloak technology Update the internal Store REST API, Information model and Interaction model sections, added technological information Minor changes related to rephrasing and typo error Update of the abbreviation table, update of the references table	23.01.2015
003	Internal review process, checking URL links to the FIspace Web online documentation Update of the abbreviation table, update of the references table Final version ready for submission of approved document	06.02.2015
004	Update of the coordinator information in the section "*More Information*" Added FIspace development repository and documentation references, formatting improvement Final version ready for submission to EC	27.02.2015
005		
006		

Abbreviations

AAA	Authentication, Authorisation, and Accounting
ACSI	Artifact-Centric Service Interoperation
AdvB	Advisory Board
AJAX	Asynchronous JavaScript + XML
API	Application Programming Interface
App	Software Application
B2B	Business-to-business
B2C	Business-to-Consumer
BCM	Business Collaboration Module in FIspace
BCO	Business Collaboration Objects in FIspace
BE	Business Entities
BPPC	Business Process Participant Configuration
BSS	Business Support Systems
CDR	Charging Detailed Records
CEP	Complex Event Processing
CSB	Cloud Service Bus
CSS	Cascading Style Sheets
CSV	Comma-Separated Values
D	Deliverable
DAO	Data Access Object
DB	Database
DoW	Description of Work
EC	European Commission
EDI	Electronic Data Interchange
EE	Experimentation Environment
e.g.	Exempli gratia = for example
EPA	Event Processing Agent
EPM	Event Processing Module in FIspace
ESB	Enterprise Service Bus
EU	European Union
FIA	Future Internet Assembly
FI-PPP	Future Internet Public Private Partnership
FP7	Framework Programme 7
GA	Grant Agreement
GE	Generic Enabler
GUI	Graphical User Interface
HTML	HyperText Markup Language
IaaS	Infrastructure as a Service
ICT	Information and Communication Technology
IDE	Integrated Development Environment
IDM	Identity Management
i.e.	id est = that is to say
IE	Integration Environment
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force
I/O	Input / Output
IoT	Internet of Things
IP	Intellectual Property
IP (protocol)	Internet Protocol
IPR	Intellectual Property Rights
IPsec	Internet Protocol Security
IT	Information Technology
ITU	International Telecommunication Union
ISO	International Standardization Organisation
J2SE	Java 2 Platform, Standard Edition
JDK	Java Development Kit
JDT	Related to Eclipse Java Development Tools
JMX	Java Management Extensions
JRE	Java Runtime Environment
JS	JavaScript
JSON	JavaScript Object Notation
JSP	Java Server Page
JVM	Java Virtual Machine
KPI	Key Performance Indicator
LPA	Logistics Planning Application
M	Month
MTBF	Mean Time Between Failures
MVC	Model–View–Controller
OASIS	Organization for the Advancement of Structured Information Standards
OAuth	Open standard Authentication protocol
OMG	Object Management Group
OSS	Operational Support Systems
P2P	Peer-to-peer
PaaS	Platform as a Service
PDE	Related to Eclipse Java Development Tools
PE	Production Environment
PIA	Product Information App

PIE	Preliminary Integration Environment
PKI	Public Key Infrastructure
PM	Person Month
POM	Project Object Model (used by maven tools)
Proton	IBM Proactive Technology Online
QoS	Quality of Service
RBAC	Role-Based Access Control
RCP	Rich Client Platform
REST	Representational State Transfer
RFC	Request for Comments
RSS	Revenue Sharing System
RTD	Research and Technological Development
SaaS	Software as a Service
SDI	System and Data Integration layer in FIspace
SDK	Software Development Kit
SME	Small and Medium Sized Enterprise
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SOA-RM	(OASIS) Reference Model for Service Oriented Architecture
SPT	Security, Privacy and Trust Framework
SSH	Secure Shell
SSL	Secure Sockets Layer
SSO	Single Sign On
ST	Sub-Task
SWT	Standard Widget Toolkit
T	Task
TCP	Transmission Control Protocol
TIC	Tailored Information for Consumers
TLS	Transport Layer Security
TPM	Transport Planning Module
UAA	User Management, Authentication and Authorisation
UI	User Interface
UML	Unified Modeling Language
URI	Universal Resource Identifier
URL	Universal Resource Locator
USDL	Unified Service Description Language
VM	Virtual Machine
VPN	Virtual Private Network
W3C	World Wide Web Consortium
WADL	Web Application Description Language
WLAN	Wireless Local Area Network
WP	Work Package
WS	Web Service
WSDL	Web Services Description Language
XLS/XLSX	Microsoft Excel file Format
XML	eXtensible Markup Language
XSD	XML Schema Definition

Table of Contents

List of Figures

List of Tables

1 Introduction

This document aims at describing the third release (V3) of the FIspace, encompassing the implementations along with usage guidance and technical documentation of each FIspace component.

It reports on the description concerning the **Store core component**, the description of the development and implementation of the Store core components that is part of the FIspace platform.

The App Store provides the infrastructure for providing, finding, and purchasing FIspace Apps, which provide re-usable IT-solutions supporting business collaborations and can be used and combined for the individual needs of users; the FIspace Store includes:

- The **software infrastructure** to support the provisioning, discovery, purchase, and use of FIspace Apps, including a **registry of Apps**.
- **Facilities for financial management** of the FIspace Apps (pricing, payment, revenue sharing).

The FIspace Store is concerned with the software infrastructure to allow for the provisioning and consumptions of FIspace Apps, therewith providing the core elements for the monetization throughout the ecosystem that shall be facilitated by FIspace. All FIspace Apps shall be made available in the Store and consumer will be supported with easy to use search and consumption features. The consumption includes the purchase support as well as deployment and runtime support. Features for the former contain an App purchase processes. Features for the latter include capabilities for dynamically connecting the Apps (which may run on different servers) to the Cloud Service Bus of the FIspace platform. Finally, for App customers are informed about the mandatory and optional rights the App requires (before purchase) and enables him or her to configure those for each App (after purchase). For App developers, publication support is provided together with an integrated compliance check for publishing new Apps in a simple way in the FIspace store. An important part of the App Store will be also the application's lifecycle support, including bug fixes and upgrades and connection with the users that purchased the App.

Finally, Financial management is part of the FIspace Store which enables App providers to run statistics and share revenue with involved partners (e.g., developer of re-used component) using different revenue models.

Online documentation for FIspace Store: http://dev.fispace.eu/doc/wiki/store

1.1 Scope

The aim of this document is mainly to describe and detail the **FIspace Store core component** at development and implementation level, giving detailed and technical information related to the design and the implementation as well as information about the related technologies and standard taken as a .reference to build each component.

Along this development activities and tasks, there is a set of resources, online documentation, tutorial and other external resource that refer to the Generic Enablers that can provide more technical information and user guides for the community and people who want to use the FIspace platform for Business collaboration or developers who want to create and develop business application (Apps developer) for a specific domain of application.

Table 1 shows the links to other online resources related to FIspace project and FI-WARE.

Description	Link
FIspace Business collaboration web site	http://www.fispace.eu/
FIspace Developer Documentation web site	http://dev.fispace.eu/doc/wiki/Home
FIspace Deliverables web site	http://www.fispace.eu/deliverable.html
FIspace Tutorial web site	http://www.fispace.eu/tutorials.html
FIWARE web site	http://www.fi-ppp.eu/projects/fi-ware/
FIWARE Catalogue of the Generic Enablers (GEs)	http://catalogue.fi-ware.org/
FIWARE community web site	http://www.fi-ware.org/community/

Table 1: Other FIspace and FIWARE resources

Table 2 shows the links to the Wirecloud online documentation.

Description	Link
FIWARE - Catalogue - Application Mashup - Wirecloud	http://catalogue.fi-ware.org/enablers/application-mashup-wirecloud
FIWARE - Catalogue - Application Mashup - Wirecloud Documentation	http://catalogue.fi-ware.org/enablers/application-mashup-wirecloud/documentation
FIWARE - Application Mashup - Wirecloud - User and Programmer Guide	https://forge.fi-ware.org/plugins/mediawiki/wiki/fiware/index.php/Application_Mashup_-_Wirecloud_-_User_and_Programmer_Guide
Dashboard - Wirecloud home page	http://conwet.fi.upm.es/wirecloud/
Dashboard - The WireCloud Mashup Platform	http://conwet.fi.upm.es/docs/display/wirecloud/The+WireCloud+Mashup+Platform
Dashboard - Welcome to CoNWeT-Wirecloud Confluence	http://conwet.fi.upm.es/docs/dashboard.action
Dashboard - User Guide	http://conwet.fi.upm.es/docs/display/wirecloud/WireCloud+User%27s+Guide
Dashboard - WireCloud Installation	http://conwet.fi.upm.es/docs/display/wirecloud/

and Administration Guide	Wire-Cloud+Installation+and+Administration+Guide

Table 2: Wirecloud online documentation

Table 3 shows the links to the WStore online documentation.

Description	Link
FIWARE - Catalogue - Store - WStore	http://catalogue.fi-ware.org/enablers/store-wstore
FIWARE - Catalogue - Store - WStore Documentation	http://catalogue.fi-ware.org/enablers/store-wstore/documentation
FIWARE - Store - W-Store - User and Programmer Guide	https://forge.fi-ware.org/plugins/mediawiki/wiki/fiware/index.php/Store_-_W-Store_-_User_and_Programmer_Guide
FIWARE - Store - W-Store - Store - W-Store - Installation and Administration Guide	https://forge.fi-ware.org/plugins/mediawiki/wiki/fiware/index.php/Store_-_W-Store_-_Installation_and_Administration_Guide

Table 3: Store online documentation

Table 4 shows the external development tools references.

Description	Link
Java Environment, JVM, JRE, JDK (Oracle)	http://www.oracle.com/technetwork/java/javase/downloads/index.html
Eclipse IDE (Integrated Development Environment)	https://www.eclipse.org/ , https://www.eclipse.org/downloads/
Maven	http://maven.apache.org/ , http://maven.apache.org/download.cgi

Table 4: External development tools references

Table 5 shows the FIspace development repository and documentation references based on the bitbucket tools for collaborative development.

Bitbucket is a hosting site for the distributed version control systems (DVCS) Git (http://git-scm.com/) and Mercurial (http://mercurial.selenic.com/). The service offering includes an issue tracker and wiki, as well as integration with a number of popular services such as Basecamp, Flowdock, and Twitter.

Description	Link
Bitbucket FIspace repository home page	https://bitbucket.org/fispace
Bitbucket FIspace core component home page	https://bitbucket.org/fispace/core/wiki/Home
Bitbucket FIspace Roadmap page	https://bitbucket.org/fispace/core/wiki/roadmap

Table 5: Bitbucket collaborative environment for FIspace development

1.2 Intended audience

The main interest groups of this deliverable are the participating teams and the responsible partners of FIspace project involved in the development activities, setup and preparation of the development phase. This document is relevant to the software engineer, programmers and developers who are the persons directly involved in the development, participating effectively on the design and implementation of the FIspace platform and the underlying components and sub-systems who want to know more about some technical information intrinsic to the FIspace platform.

At the technical level this document is relevant to: system architects; information systems designers; system developers and application developers; software engineers; other audiences who provide design services and applications using relevant standards and the recommendations of standards bodies like IETF, ITU, ISO, W3C, etc.

Partners involved in the integration tasks include: system integrators; people to test, validate and evaluate the FIspace platform and associated systems; can be also interested.

1.3 General remark

This document follows the ISO/IEC Directives, Part 2: Rules for the structure and drafting of International Standards w.r.t. the usage of the word “shall”. The word “shall” (not “must”) is the verb form used to indicate a requirement to be strictly followed to conform to this specification.

This document describes the corresponding core components involved in the FIspace core platform. It presents the development currently done and the corresponding implementation, the main features developed, as well as the related technologies and environment requirements.

In most of the following sections the structure is organized as:

- **Overview**: provides an overall introduction to the component, a description, of the internal architecture and features among other.
- **Interfaces or Application programming interface (API)**: describes the API accessible for the users or entities of the component (typically applications, but a component may also be used by other components).
- **Information model**: describes or specifies the component from an information perspective describing information objects of the component domain.
- **Interaction model**: describes or specifies main usage component “scenarios” associated with the component/GEs, sequence diagrams.

- **High level composite architecture**: describes or shows the main components constituting the set of components (this perspective is optional, since some component consists of only one main component).

Notice that some components only need to describe some of the item above described.

2 Store

2.1 Overview

The FIspace Store is tool-supported infrastructures for the provisioning, consumptions, purchase, and re-use of IT-solutions for seamless business collaboration. The FIspace Store supports the Financial Management of Apps (pricing, payment, revenue sharing) for both End-Users and App Developers.

The FIspace Store provides the core elements for the monetization of Apps throughout the FIspace ecosystem. App Developers publish and manage the financial aspects of Apps, while End-Users seach and consume Apps.

App Developers are provided with an Apps Interface discovery system, publication support tools and financial management tools. The Apps Interface discovery system is used to reuse functionality and build mashup applications. The publication support tool provides an integrated compliance check that eases the publishing of new Apps. The financial management tools enable developers to run statistics, define revenue models and share revenue with any involved partner (e.g. developer of re-used component).

End-Users are provided with a search, purchase and execution support system. The search system allows End-Users to find applications according to a set of parameters. The purchase support system guides End-Users in the purchase process for Apps. The execution support system provides End-Users with download and installation guides, in case of locally run Apps, and access control systems for Apps executing in the Cloud. A simple right management system enables End-Users to customize Apps after purchasing.

All Apps are stored in an App repository. As shown in the Figure 1, the FIspace Store has 2 main user groups: Consumers as the (Business) End-Users who utilize the FIspace apps to conduct business tasks (esp. the collaborative business activities across organizational boundaries), and App Developers who develop Apps for the FIspace.

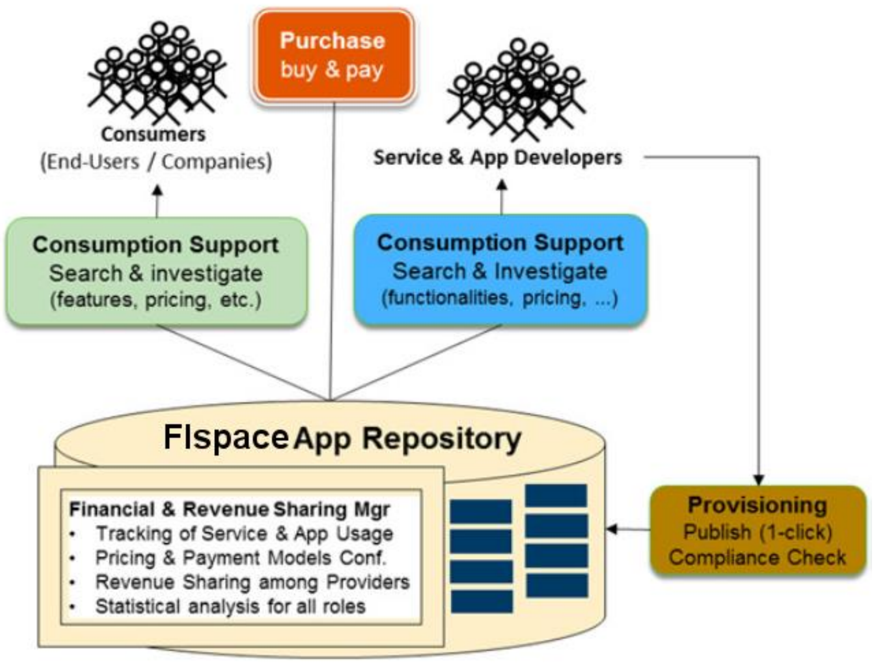

Purchase
buy & pay
Consumers
(End-Users / Companies)
Service & App Developers
Consumption Support
Search & investigate
(features, pricing, etc.)
Consumption Support
Search & Investigate
(functionalities, pricing, ...)
FIspace App Repository
Financial & Revenue Sharing Mgr
• Tracking of Service & App Usage
• Pricing & Payment Models Conf.
• Revenue Sharing among Providers
• Statistical analysis for all roles
Provisioning
Publish (1-click)
Compliance Check

Figure 1: FIspace Store Overview

A Consumption Support component provides the necessary software infrastructure to find and re-use Apps. A Purchase component supports the purchasing process including buying an App (e.g. terms and condition agreement) and paying for it. A Provisioning Support component enables developers to easily upload and publish newly developed Apps, including technical compliance checks. In addition, the Financial & Revenue Sharing Manager, who is part of the FIspace App Repository, handles the monetary incomes and the revenue sharing.

Technically, the FIspace Store and the necessary software infrastructure are developed on top of existing frameworks and technologies. Primary candidates are the Generic Enablers from the FI-WARE IoS Chapter, using (Linked) USDL as the basis for the App Description Language, the Service Repository as basis for the FIspace App repository and the Marketplace GE as basis for features of the Consumption Support and Provisioning components.

2.2 Interfaces / API

Table 6 enumerates the main groups of store interface.

Function	Brief description	Sender	Receiver
Store CDRs in the RSS.	Store one or more CDRs in the RSS for further processing	Store	RSS
Get extensions	List of all available extensions		Repository
Managing collections	Creates / Updates/ Retrieves / Deletes a collection		Repository
Managing resources	Creates / Updates / Retrieves / Deletes a resource		Repository
Additional services	Get a list of additional services		Repository
Searching	Search a collection		Repository
Retrieve Apps Create App	Retrieves information from an existing Apps Creates a new App in the System	SDK SDK	Store Store

Table 6: Functional interface description

2.2.1 API operations

Table 7 enumerates the main Store API operations.

Verb	URI	Description	Mandatory / Optional
POST	/rss/cdrs	Store CDRs in the RSS.	Mandatory
PUT	/{DistinguishedEntryName}	Create or update a resource or a number of resources. The request body of a PUT operation should contain the set of attributes of the entry. E.g. if Content-type was “application/json"	Mandatory
DELETE	/{DistinguishedEntryName}	Delete a registry entry.	Mandatory
GET	/extensions	List of all available extensions.	Mandatory
GET	/{CollectionPath}	Get a collection.	Mandatory
PUT	/{CollectionPath}	Create or update a collection.	Mandatory
DELETE	/{CollectionPath}	Delete a collection.	Mandatory
GET	/{CollectionPath}/{ResourceID}	Get a resource.	Mandatory
PUT	/{CollectionPath}/{ResourceID}	Create or update a resource.	Mandatory
DELETE	/{CollectionPath}/{ResourceID}	Delete a resource.	Mandatory
GET	/{collectionPath}/services	Get a list of additional services.	Mandatory
GET	/{CollectionPath}/search?q={queryString}	Search a collection.	Mandatory
GET	/Fispace/Store/service/getAvailableApps	Retrieves all the available apps using the WStore functionality.	Mandatory

Verb	URI	Description	Mandatory / Optional
POST	/Fispace/Store/service/createApp	Creates an app according to the specified offering information.	Mandatory

Table 7: Store API – Operations description

2.3 Information model

The RSS defines a data model for the storage of transactions in the store. This model includes entities such as providers, users, CDRs, RSS model etc. The complete data model is shown in Figure 2.

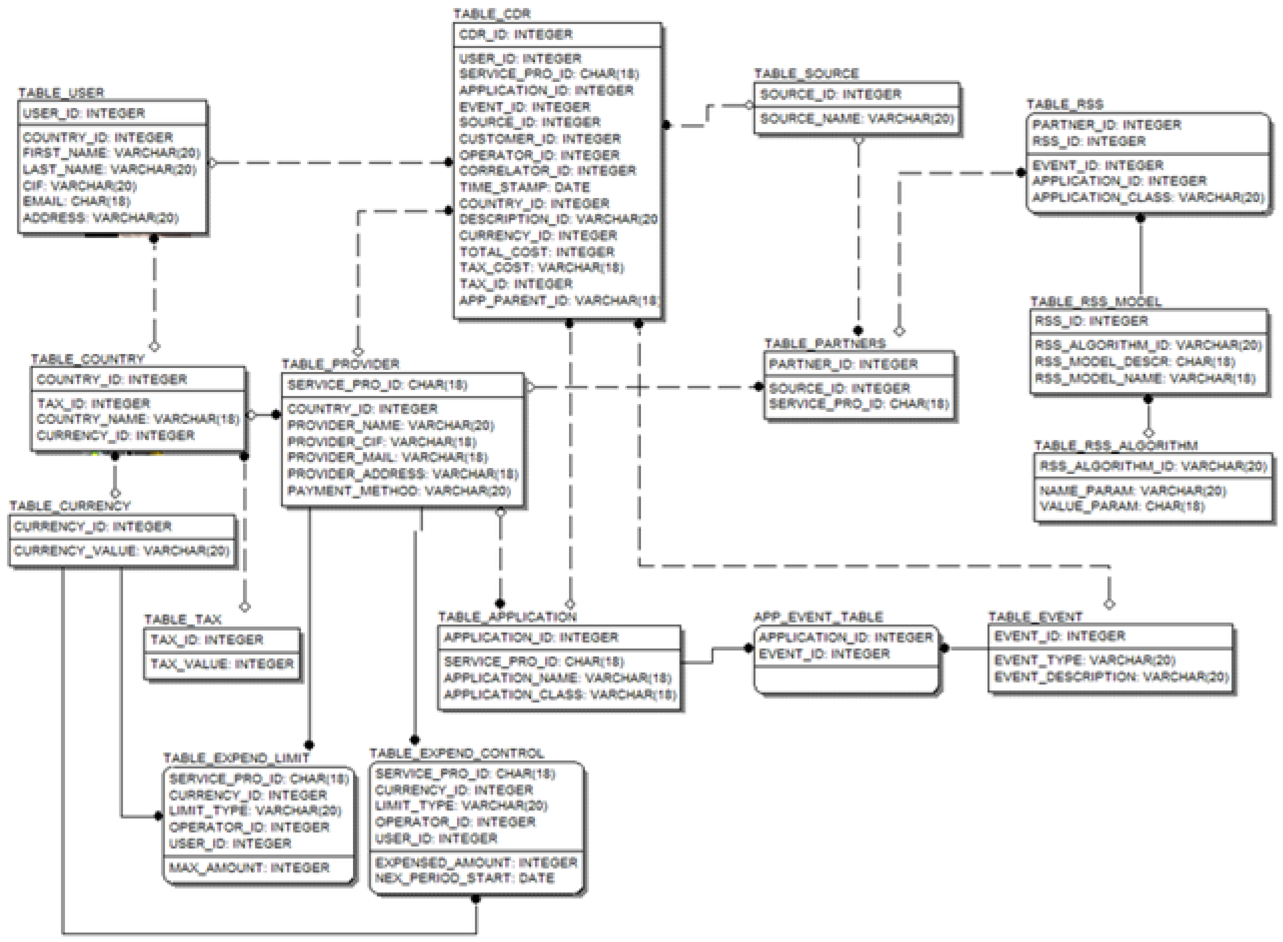


Figure 2: RSS GE Data model

Revenue Sharing models describe the way to distribute revenues between the different stakeholders involved in a given service. The most important attributes include a unique ID, source (system that originates the charging information), event being charged (pay per use, unique payment, etc.), service provider, application id and algorithm used to calculate the share.

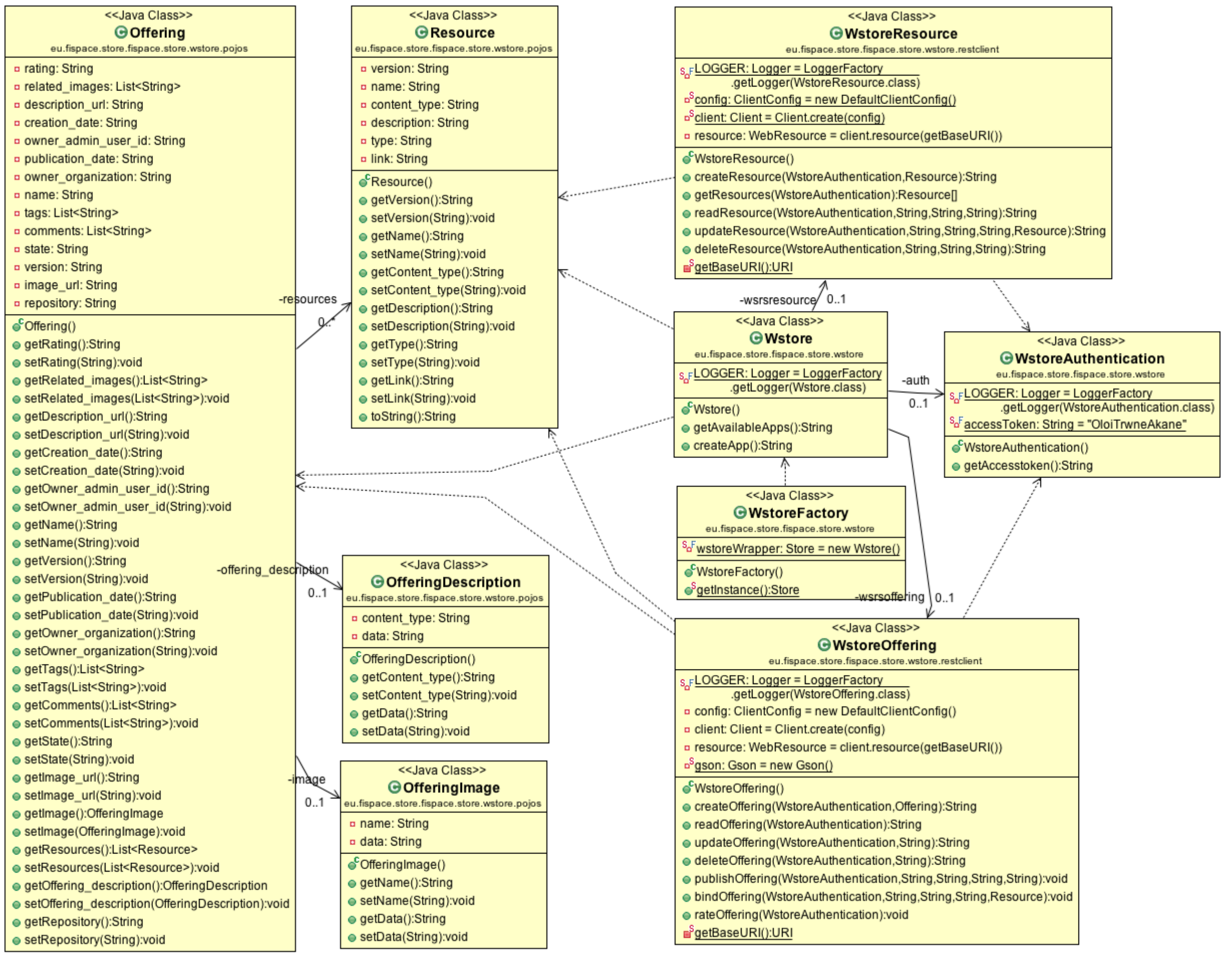
CDRs provide charging information related to the usage of services available. The most important attributes are the source of the CDR (marketplace, store, etc.), type (charge, refund, etc.), application id, event id, cost, currency, user ID for the consumer of the service and service provider ID.

Finally the RSS keeps track of the Providers in order to share the revenue incurred by their applications. The most important attributes are their id, name, payment method and preferred currency.

The WStore GE keeps track of Offerings and Resources in FIspace. Offerings are represented by an offering description and offering image, and are composed of a set of resources.

WStore Offering and WStore Resource represent offerings and resources according to the WStore GE API. These classes act as a façade between FIspace and the WStore GE, and therefore bridge the gap between both systems.

Finally WStore Authentication provides an oauth2 bearer in order to allow access to the WStore GE API.



<<Java Class>> Offering eu.fispace.store.fispace.store.wstore.pojos
rating: String
related_images: List<String>
description_url: String
creation_date: String
owner_admin_user_id: String
publication_date: String
owner_organization: String
name: String
tags: List<String>
comments: List<String>
state: String
version: String
image_url: String
repository: String
Offering()
getRating():String
setRating(String):void
getRelated_images():List<String>
setRelated_images(List<String>):void
getDescription_url():String
setDescription_url(String):void
getCreation_date():String
setCreation_date(String):void
getOwner_admin_user_id():String
setOwner_admin_user_id(String):void
getName():String
setName(String):void
getVersion():String
setVersion(String):void
getPublication_date():String
setPublication_date(String):void
getOwner_organization():String
setOwner_organization(String):void
getTags():List<String>
setTags(List<String>):void
getComments():List<String>
setComments(List<String>):void
getState():String
setState(String):void
getImage_url():String
setImage_url(String):void
getImage():OfferingImage
setImage(OfferingImage):void
getResources():List<Resource>
setResources(List<Resource>):void
getOffering_description():OfferingDescription
setOffering_description(OfferingDescription):void
getRepository():String
setRepository(String):void
-resources 0..*
-offering_description 0..1
-image 0..1
<<Java Class>> Resource eu.fispace.store.fispace.store.wstore.pojos
version: String
name: String
content_type: String
description: String
type: String
link: String
Resource()
getVersion():String
setVersion(String):void
getName():String
setName(String):void
getContent_type():String
setContent_type(String):void
getDescription():String
setDescription(String):void
getType():String
setType(String):void
getLink():String
setLink(String):void
toString():String
<<Java Class>> OfferingDescription eu.fispace.store.fispace.store.wstore.pojos
content_type: String
data: String
OfferingDescription()
getContent_type():String
setContent_type(String):void
getData():String
setData(String):void
<<Java Class>> OfferingImage eu.fispace.store.fispace.store.wstore.pojos
name: String
data: String
OfferingImage()
getName():String
setName(String):void
getData():String
setData(String):void
<<Java Class>> WstoreResource eu.fispace.store.fispace.store.wstore.restclient
LOGGER: Logger = LoggerFactory .getLogger(WstoreResource.class)
config: ClientConfig = new DefaultClientConfig()
client: Client = Client.create(config)
resource: WebResource = client.resource(getBaseURI())
WstoreResource()
createResource(WstoreAuthentication,Resource):String
getResources(WstoreAuthentication):Resource[]
readResource(WstoreAuthentication,String,String,String):String
updateResource(WstoreAuthentication,String,String,String,Resource):String
deleteResource(WstoreAuthentication,String,String,String):String
getBaseURI():URI
-wsrsresource 0..1
<<Java Class>> Wstore eu.fispace.store.fispace.store.wstore
LOGGER: Logger = LoggerFactory .getLogger(Wstore.class)
Wstore()
getAvailableApps():String
createApp():String
-auth 0..1
<<Java Class>> WstoreAuthentication eu.fispace.store.fispace.store.wstore
LOGGER: Logger = LoggerFactory .getLogger(WstoreAuthentication.class)
accessToken: String = "OloiTrwneAkane"
WstoreAuthentication()
getAccesstoken():String
<<Java Class>> WstoreFactory eu.fispace.store.fispace.store.wstore
wstoreWrapper: Store = new Wstore()
WstoreFactory()
getInstance():Store
-wsrsoffering 0..1
<<Java Class>> WstoreOffering eu.fispace.store.fispace.store.wstore.restclient
LOGGER: Logger = LoggerFactory .getLogger(WstoreOffering.class)
config: ClientConfig = new DefaultClientConfig()
client: Client = Client.create(config)
resource: WebResource = client.resource(getBaseURI())
gson: Gson = new Gson()
WstoreOffering()
createOffering(WstoreAuthentication,Offering):String
readOffering(WstoreAuthentication):String
updateOffering(WstoreAuthentication,String):String
deleteOffering(WstoreAuthentication,String):String
publishOffering(WstoreAuthentication,String,String,String,String):void
bindOffering(WstoreAuthentication,String,String,String,Resource):void
rateOffering(WstoreAuthentication):void
getBaseURI():URI

Figure 3: WStore module UML class diagram

2.4 Interaction model

2.4.1 Store component

The Store component is a Façade between the FIspace platform and the underlying GEs. This module exposes an API that translates the operations to the offering format consumed by the WStore GE, which in turn communicates with the RSS GE, Repository GE and Marketplace GE.

Store component exposes an interface with high level operations to the rest of the FIspace platform (e.g. App creation, App listing, Revenue model creation, etc.). This interface is accessed by the SDI using the internal communication system CSB to forward requests from external components (e.g. SDK) to the Store. The Store, in turn, forward these requests to the Business Framework GEs from FIWARE (including WStore, Marketplace, RSS), which expose their functionality through REST interfaces.

2.4.2 App creation

The App creation process starts at the SDK component. The SDK sends an "App" message to the SDI, which is forwarded to the Store. The Store component interacts with the WStore GE in order to create a new App. Finally, the WStore GE publishes the App information in the Marketplace GE.

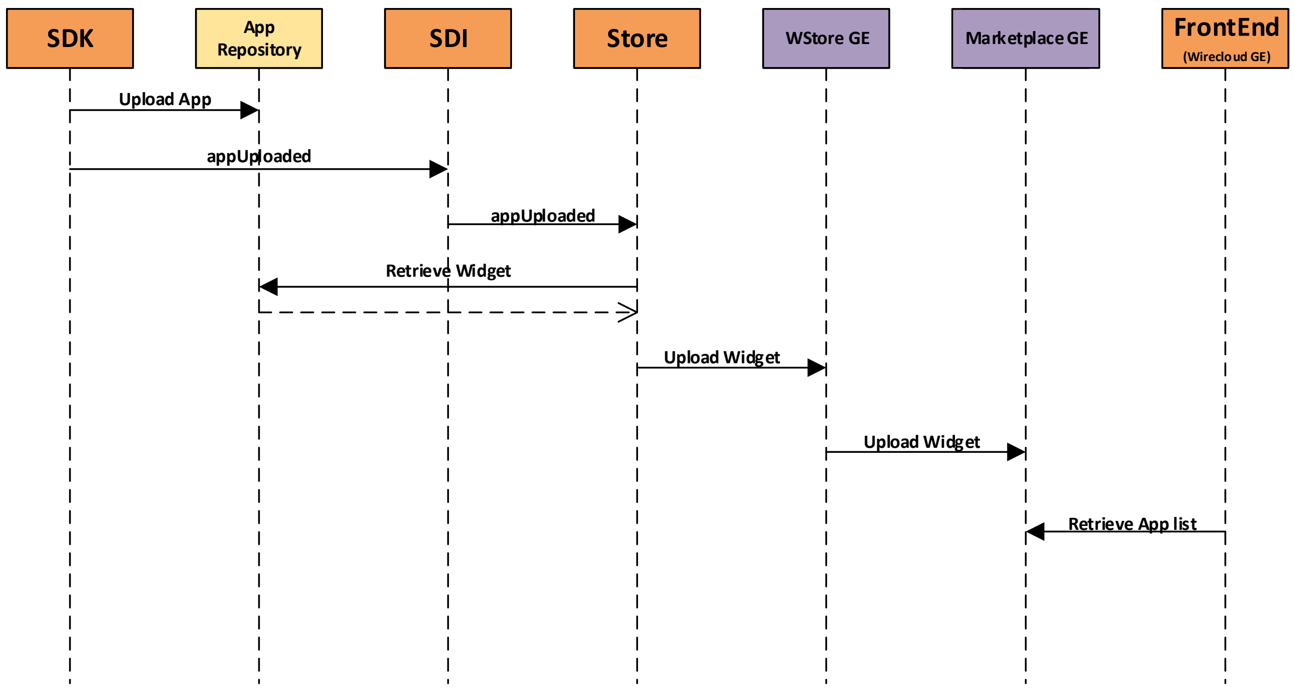


Figure 4: App creation sequence diagram

2.4.3 Linked USDL

Linked USDL (Unified Service Description Language) is a service description language that describes the technical, business and operational aspects of a service. It is based on the Resource Description Framework (RDF) and consists of several modules for a service description. It can be easily extended with additional domain-specific extensions, making it an ideal candidate for the description of FIspace Apps.

2.4.4 Marketplace GE

The Marketplace GE brings together offering and demand for business operations. It provides hub functionality for the implementation of App stores through its API. These operations include registering stores, publishing and retrieving offerings and demands; search and discover offerings according to specific consumer requirements; and lateral functions like review, rating and recommendation. The Discovery & Matchmaking component provide operations for the comparison of offers and demand. The Marketplace GE also provides a single point of access to the system by connecting to multiple App stores.

2.4.5 Repository GE

The Repository provides a consistent and uniform API to Linked USDL service descriptions and associated media files. Service providers use the Repository GE to publish their Linked USDL descriptions or any other files. The Repository GE provides the following functionality:

- Uploading, storing and deleting RDF files
- Uploading USDL service description
- Upload, store, delete of various data.

Specifically, the Repository GE interacts with the following components:

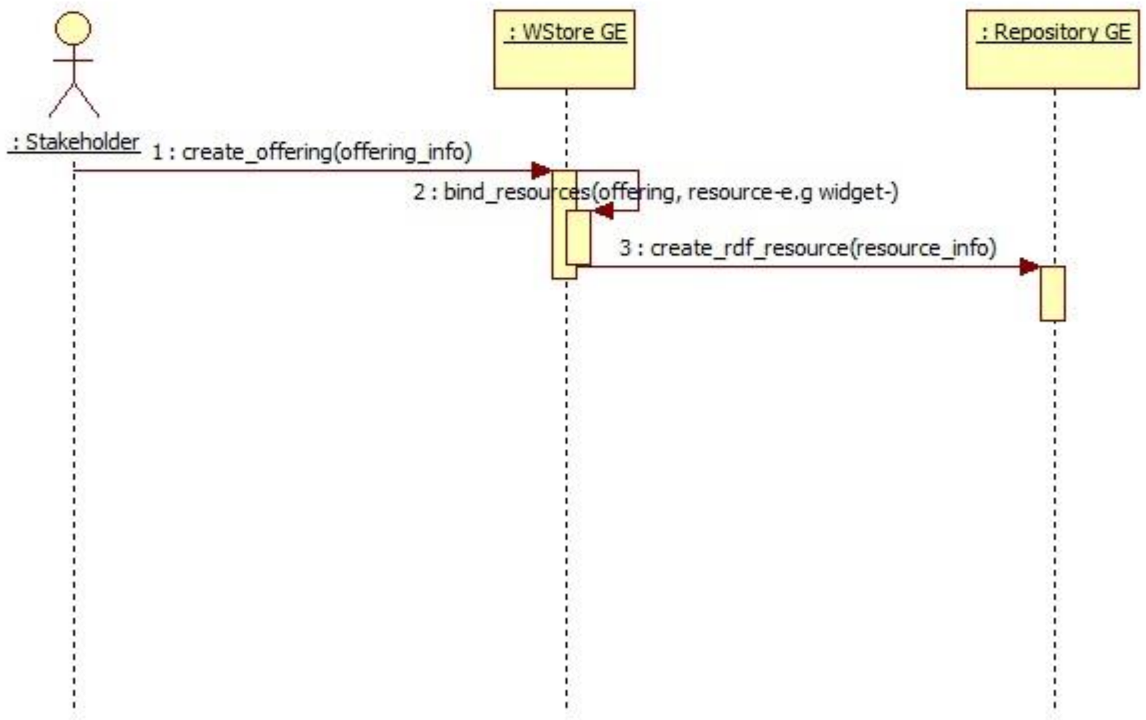
- Marketplace GE read RDF service description from the Repository
- WStore GE create/read RDF services description from the Repository

2.4.6 WStore GE

The WStore GE provides the management operations (accounting, billing, etc.) for services of a store. In order to access these operations, offerings must be registered in the WStore. The process starts with an offering creation request by a client. The WStore store the RDF description in the Repository GE, and publishes the offering in the Marketplace GE. After the offering is published, it can be purchased by customers and managed by clients using the WStore API.

For the next milestone we have installed the newest release of WStore GE, v0.4.1. This version fixes a large number of bugs and extends the integration WStore-RSS. These new features are useful for the integration of RSS with the rest of the platform.

Figure 5 and Figure 6 show the sequence diagram of create offering and publishing offering operations respectively.



: Stakeholder
: WStore GE
: Repository GE
1 : create_offering(offering_info)
2 : bind_resources(offering, resource-e.g widget-)
3 : create_rdf_resource(resource_info)

Figure 5: Create Offering operation

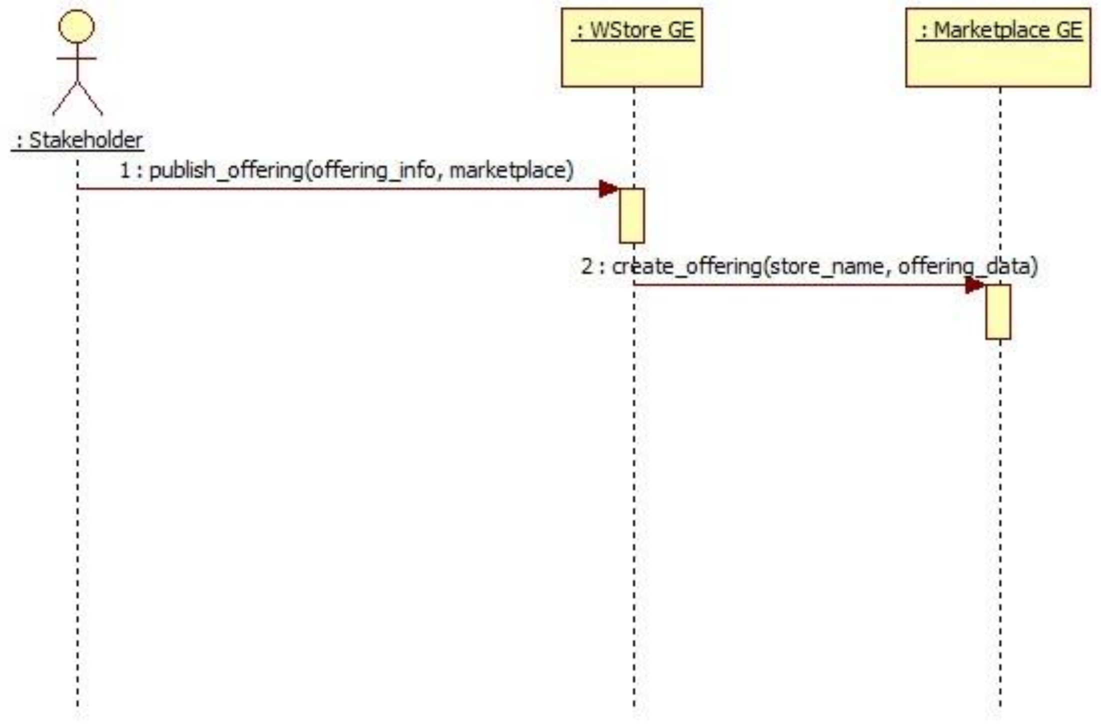

: Stakeholder
: WStore GE
: Marketplace GE
1 : publish_offering(offering_info, marketplace)
2 : create_offering(store_name, offering_data)

Figure 6: Publish Offering operation

2.4.7 WStore GE and Integration of WireCloud GE

The WireCloud GE interacts with the WStore GE and Marketplace GE. Widgets offerings are provided via the Marketplace, and deployment and payment is handled by the WStore.

WireCloud allows developers to purchase widgets or mash-ups. Developers can then either use them as apps, or build upon them to create new mash-ups. These mash-ups are described using USDL, and can be published to the Marketplace via WireCloud.

Figure 7 shows the sequence diagram and the interaction between Wirecloud GE and WStore GE and underlying GE components.

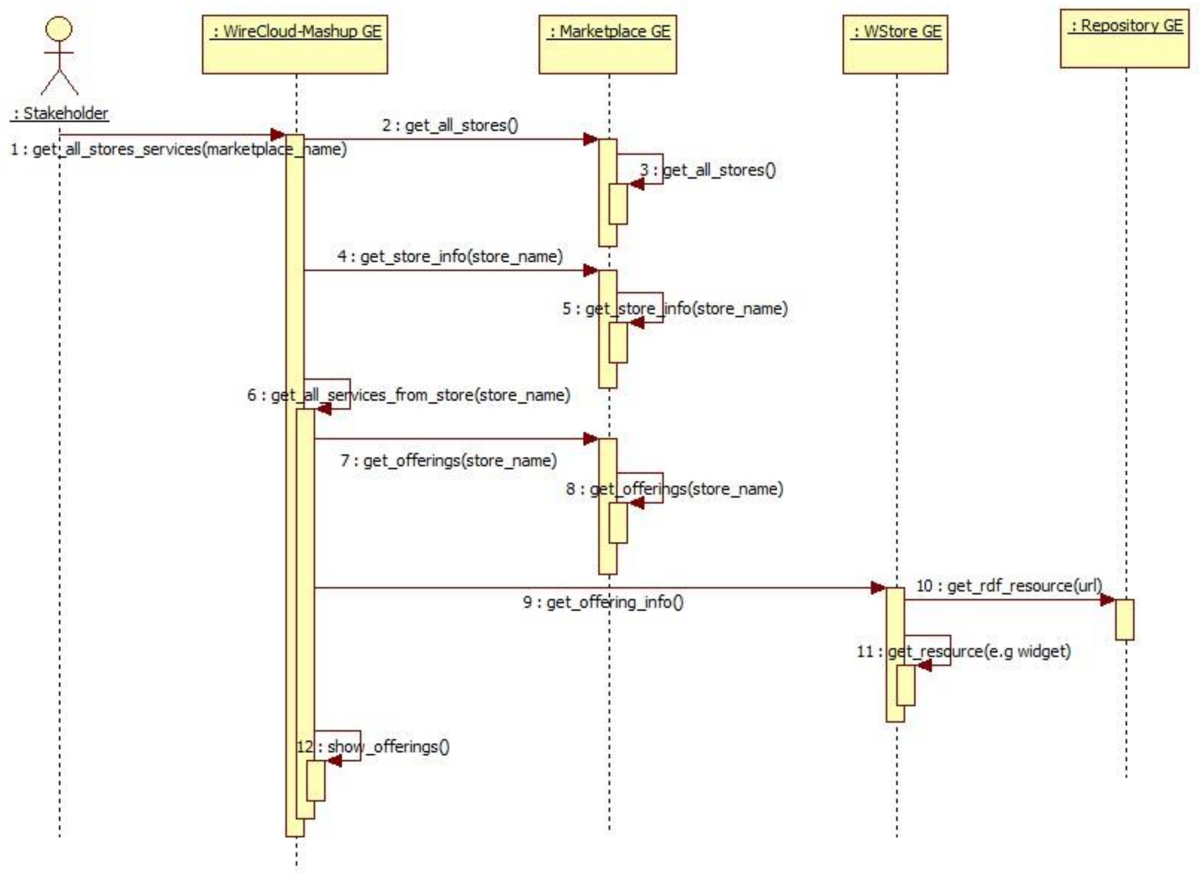

: Stakeholder
: WireCloud-Mashup GE
: Marketplace GE
: WStore GE
: Repository GE
1 : get_all_stores_services(marketplace_name)
2 : get_all_stores()
3 : get_all_stores()
4 : get_store_info(store_name)
5 : get_store_info(store_name)
6 : get_all_services_from_store(store_name)
7 : get_offerings(store_name)
8 : get_offerings(store_name)
9 : get_offering_info()
10 : get_rdf_resource(url)
11 : get_resource(e.g widget)
12 : show_offerings()

Figure 7: Wirecloud GE,Marketplace GE, WStore GE interaction

2.4.8 Keycloak IDM integration with WStore GE and Store component.

The WStore GE has been modified to support Keycloak IDM [32] (Identity Management) in order to provide a single sign on mechanism with the rest of the FIspace platform. This mechanism allows for the secure exchange of information between components. Each GE has a specific role in the IDM, so that rights for the use of operations can be defined individually. This system restricts each component to perform only the operations it needs.

WStore GE is configured by default to support Keyrock IDM GE. This default configuration is incompatible with Keycloak IDM (the one used by the FIspace platform) due to the structure of the information exchanged in the authentication process.

In order to support SSO, the authentication backend of WStore has been modified to handle the user information according to the Keycloak format. These modifications have been performed in collaboration with the WStore development team.

Finally the Store component also integrates with the Keycloak IDM in order to communicate with the WStore interface.

2.4.9 Revenue Sharing System GE

This component is in charge of processing the payments of the store and shares the revenue between the involved parties.

When a new offering is registered in a store and marketplace, the store sends the revenue sharing model of the offering to the Revenue Sharing System (RSS). Revenue sharing models are defined for each offering, and can define different models (single time payment, pay per use...) and different actors involved.

When a consumer buys a service, the store generates Charging Detailed Records (CDRs) which are forwarded to the RSS for processing. Then the RSS is in charge of splitting the revenue between the involved actors. If the service has a periodic model of payment (monthly, pay-per-use, etc.) the store schedules for periodically charging the consumer, generating CDRs and sending them to the RSS.

The RSS then periodically aggregates these CDRs, process them and split the revenue.

The store charges consumers, and the RSS takes this money and splits it between the developers/providers (or anyone involved in the offering) according to the defined policy.

2.5 High level composite architecture

Figure 8 shows the high level composite architecture for the Store components.

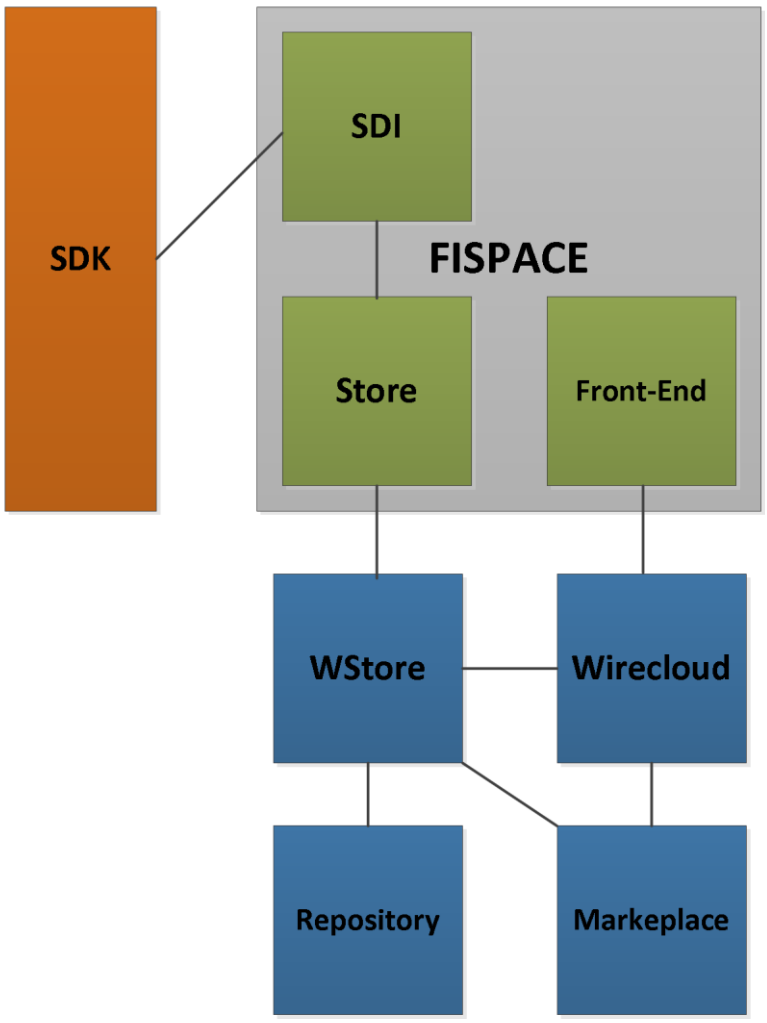


Figure 8: Store – High level composite architecture

The High level composite architecture related to the store components is composed by the mains building block:

- **SDK**: Provide the functionality to upload apps to the Store component using the SDI´s interface.
- **SDI:** The entry point to FIspace, allows the communication between the components that integrate the platform.
- **Front-End:** The gateway to FIspace for users of the platform. Through the integration of external widgets (e.g., from the store, externally developed Apps or other external providers), the User Front-End facilitates an ‘all you need in one place’ user experience and creates a central access point.

The GEs required by the store components are the followings:

- **WStore**: Keeps track of Offerings and Resources in FIspace. Offerings are represented by an offering description and offering image, and are composed of a set of resources.
- **Wirecloud**: Integrates heterogeneous data, application logic, and UI components (widgets/gadgets) sourced from the Web to create new coherent and value-adding composite applications.
- **Repository**: Provides a consistent and uniform API to Linked USDL service descriptions and associated media files
- **Marketplace**: Provides functionality necessary for bringing together offering and demand for making business. These functions include basic services for registering business entities, publishing and retrieving offerings and demands, search and discover offerings according to specific consumer requirements as well as lateral functions like review, rating and recommendation.

The architecture of the Store is divided in two clearly separated domains: the FIspace platform and the FIWARE Business Framework GEs.

The FIspace platform includes all the FIspace internal components. In the picture we have only included those that interact with the Store. FIspace is a tightly integrated system with a SSO facility, provided by the Keycloak IDM, and an internal communication system, provided by CSB.

The FIWARE Business Framework GEs is an ecosystem of Generic Enablers provided by the FIWARE foundation for implementing different business processes. The ones relevant for the FIspace Store are the ones implementing the App publishing and purchase capabilities. These GEs are WStore, Wirecloud, Repository and Marketplace. These GEs are integrated with each other, but exists outside the FIspace platform. That means that they do not communicate using CSB and, generally, don´t integrate with the SSO system. As an exception, WStore and Wirecloud GEs have been modified to support the FIspace SSO system, since these GEs are in direct communication with FIspace components.

The Store component exposes its API to the FIspace platform through CSB. This API is consumed by SDI, which exposes all FIspace APIs to software external to the platform. The main consumer of the Store API is the SDK, which uses its functionality to create Apps in the system.

Store component publishes user Apps in WStore, which in turn publish Apps in the Marketplace and makes them visible in Wirecloud. Then, the Front-End uses Wirecloud to allow users to purchase, use and mashup Apps. This flow of information implies that there is communication between Store and Front-End, although this communication is indirect between GEs interaction, and not directly between components.

2.6 Technology

2.6.1 Features provided

- App deployment and management.
- App purchase.
- Revenue Sharing model for Apps.

2.6.2 Related technologies

- FIWARE Business Framework GEs (WStore GE, Marketplace GE, Repository GE).
- Keycloak [32] open source project (JBoss project).
- FIspace CSB component (Cloud Service Bus, which is part of the FIspace operative environment).

2.6.3 Environment requirements

- Apache server to host Store component, Marketplace GE and Repository GE.
- Django server [33] and MongoDB [34] to host WStore GE.
- Custom modifications to WStore to support SSO based on Keycloak [32].

3 Glossary

The glossary provides the coherent terminological framework used in this document.

3.1 Terms and definitions

This section provides definitions of any terms that may be needed in order for the reader to understand the terminology used in the document. The author should define any definition/acronym or technical term used in the document that may be unfamiliar to the reader, and it is best to err on the side of too many rather than too few definitions. This also allows the author to frame a word within a specific context, which provides the reader with a common understanding of the author's definition.

Access control

Authorisation (or denegation) for performing a certain action (based on privileges management). The access control is carried out once the Identification and Authentication procedures have been performed.

Accounting

Process of gathering information about the usage of resources by subjects.

Acceptance and trust

Acceptability indicates the degree of approval of a technology by the users. It depends on whether the technology can satisfy the needs and expectations of its users and potential stakeholders. Within the framework of introducing new technologies, acceptability relates to social and individual aspects as well.

Application

Use of capabilities, including hardware, software and data, provided by an information system specific to the satisfaction of a set of user requirements in a given application domain.

Application Domain

Integrated set of problems, terms, information and tasks of a specific thematic domain that an application (e.g. an information system or a set of information systems) has to cope with.

Application Schema [ISO/FDIS 19109:2003]

Conceptual schema for data required by one or more applications.

Architecture (of a system) [ISO/IEC 10746-2:1996]

Set of rules to define the structure of a system and the interrelationships between its parts.

Architecture (of a system) [ISO/IEC 10746-2:1996]

Set of rules to define the structure of a system and the interrelationships between its parts.

Authentication

Process of verifying the identity of a certain subject. In other words authentication indicates whether a subject is who/what it seems to be.

Generally speaking, this proof can depend on a secret that can be, e.g. what somebody has (key, smart card, …), what somebody knows (password, …), what somebody is (biometrical data, …)

Authorisation

Process of determining whether a subject is allowed to have the specified types of access to a particular resource. This is done by evaluating applicable access control information contained in a so called authorisation context. Usually, authorisation is carried out after the identification and authentication. Once a subject is identified and authenticated, it may be authorized (or not) to perform different types of access.

Availability

Availability refers to the degree to which a system, subsystem, or equipment is in a specified operable and committable state at the start of a mission, when the mission is called for at an unknown, i.e., a random time. So, availability is the proportion of time that a system is in operating condition.

Capability

Capabilities are a set of functionalities, through a combination of software and hardware, used to provide services and data. They can reside in a system or for example in a terminal itself as embedded capabilities or they can be available through the network services and infrastructure and others communication technologies as external capabilities.

Catalogue [derived from http://www.opengeospatial.org/resources/?page=glossary]

Collection of entries, each of which describes and points to a feature collection. Catalogues include indexed listings of feature collections, their contents, their coverages, and of meta-information. A catalogue registers the existence, location, and description of feature collections held by an Information Community. Catalogues provide the capability to add and delete entries. A minimum Catalogue will include the name for the feature collection and the locational handle that specifies where these data may be found. Each catalogue is unique to its Information Community.

Certificate Authority

A Trusted Third Party, responsible for ensuring the binding between the public keys and the personal data of their respective owners.

Component

Hardware component (device) or Software Component.

Conceptual model [ISO/FDIS 19109:2003(E); ISO 19101]

Model that defines concepts of a universe of discourse.

Conceptual schema [ISO/FDIS 19109:2003(E); ISO 19101]

Formal description of a conceptual model.

Coverage [ISO 19123]

Function from a spatial, temporal or spatiotemporal domain to an attribute range. A coverage associates a position within its domain to a record of values of defined data types. Thus, a coverage is a feature with multiple values for each attribute type, where each direct position within the geometric representation of the feature has a single value for each attribute type.

Data acquisition

Methods of data acquisition include methods to collect background data, digitally acquire data from sensors, and subjective data (such as data acquired from questionnaires). In addition, data in the form of manually or automatically transcribed data and reductions of collected data is also considered sensor acquired data (but with a manual sensor – the analyst).

Description Logics

Family of logic based knowledge representation languages that are a decidable subset of first order logic with well-defined semantics and inferencing (problem decision procedures). In Description Logics, a distinction is made between the terminological knowledge and the assertional knowledge. This distinction is useful for knowledge base modelling and engineering: for modelling it is just natural to distinguish between concepts and individuals; for engineering it helps by separating key inference problems.

Digital Certificate

A kind of digital document that contains structured information about the identity of its owner along with her/his public key, signed all together with a Certificate Authority's private key.

Digital Signature

The encrypted form of a message with the private key of the owner, indicating in a secure way the creator of the message, as well as the identity of a signed data.

Encryption

The act of modifying the contents of a message in an algorithmic and secure way, so that it can not be observed or altered in while in transit.

End-User

All users that are involved in an application domain and that use the applications, the services built by the system users according to the system and service Architecture.

Feature [derived from ISO 19101]

Abstraction of a real world phenomenon [ISO 19101] perceived in the context of an Application. In this general sense, a feature corresponds to an "object" in analysis and design models.

Framework [http://www.opengeospatial.org/resources/?page=glossary]

An information architecture that comprises, in terms of software design, a reusable software template, or skeleton, from which key enabling and supporting services can be selected, configured and integrated with application code.

Generic

A service is generic, if it is independent of the application domain. A service infrastructure is generic, if it is independent of the application domain and if it can adapt to different organisational structures at different sites, without programming (ideally).

Identification

The identification process allows relating a person/device with the service environment. The “electronic identity” is something like a credential or a “business card”, suitable to be verified throughout the authentication process.

Implementation [http://www.opengeospatial.org/resources/?page=glossary]

Software package that conforms to a standard or specification. A specific instance of a more generally defined system.

Info-structure Service

Service that is required to operate a system oriented service in the sense that it plays an indispensable role in the operation of an architecture or system oriented service.

Interface [ISO 19119:2005; http://www.opengis.org/docs/02-112.pdf]

Named set of operations that characterize the behaviour of an entity.

The aggregation of operations in an interface, and the definition of interface, shall be for the purpose of software reusability. The specification of an interface shall include a static portion that includes definition of the operations. The specification of an interface shall include a dynamic portion that includes any restrictions on the order of invoking the operations.

Interoperability [ISO 19119:2005 or OGC; http://www.opengeospatial.org/resources/?page=glossary]

Capability to communicate, execute programs, or transfer data among various functional units in a manner that require the user to have little or no knowledge of the unique characteristics of those units [ISO 2382-1]. (http://www.opengeospatial.org/ogc/glossary/i)

Loose coupling [W3C; http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/#loosecoupling]

Coupling is the dependency between interacting systems. This dependency can be decomposed into real dependency and artificial dependency: Real dependency is the set of features or services that a system consumes from other systems. The real dependency always exists and cannot be reduced. Artificial dependency is the set of factors that a system has to comply with in order to consume the features or services provided by other systems. Typical artificial dependency factors are language dependency, platform dependency, API dependency, etc. Artificial dependency always exists, but it or its cost can be reduced. Loose coupling describes the configuration in which artificial dependency has been reduced to the minimum.

Middleware [http://www.opengeospatial.org/resources/?page=glossary]

Software in a distributed computing environment that mediates between clients and servers.

Open Architecture [based on (Powell 1991)] [35]

Architecture whose specifications are published and made freely available to interested vendors and users with a view of widespread adoption of the architecture. An open architecture makes use of existing standards where appropriate and possible and otherwise contributes to the evolution of relevant new standards.

Operation [ISO 19119:2005; http://www.opengis.org/docs/02-112.pdf]

Specification of a transformation or query that an object may be called to execute. An operation has a name and a list of parameters.

Performance indicators definition (PI)

PIs are quantitative or qualitative measurements, agreed on beforehand, expressed as a percentage, index, rate or other value, which is monitored at regular or irregular intervals and can be compared with one or more criteria.

Platform (Service)

Set of infrastructural means and rules that describe how to specify service interfaces and related information and how to invoke services in a distributed system.

Reference Model [ISO Archiving Standards; http://ssdoo.gsfc.nasa.gov/nost/isoas/us04/defn.html]

A reference model is a framework for understanding significant relationships among the entities of some environment, and for the development of consistent standards or specifications supporting that environment. A reference model is based on a small number of unifying concepts and may be used as a basis for education and explaining standards to a non-specialist.

Reliability

Reliability is the ability of a system or component to perform its required functions in routine circumstances, as well as hostile or unexpected circumstances, under stated conditions for a specified period of time.

Resource

Functions (possibly provided through services) or data objects.

Service [ISO 19119:2005; ISO/IEC TR 14252; http://www.opengis.org/docs/02-112.pdf]

Distinct part of the functionality that is provided by an entity through interfaces.

REST

Representational state transfer (REST) is an abstraction of the architecture of the World Wide Web; more precisely, REST is an architectural style consisting of a coordinated set of architectural constraints applied to components, connectors, and data elements, within a distributed hypermedia system. REST ignores the details of component implementation and protocol syntax in order to focus on the roles of components, the constraints upon their interaction with other components, and their interpretation of significant data elements.

Service [ISO 19119:2005; ISO/IEC TR 14252; http://www.opengis.org/docs/02-112.pdf]

Distinct part of the functionality that is provided by an entity through interfaces.

Session

Temporary association between a subject and a principal as a result of an authentication process initiated by the subject. Information about a session is stored in authentication session information.

SOAP

Simple Object Access protocol is a protocol specification for exchanging structured information in the implementation of web services in computer networks. It uses XML Information Set for its message format, and relies on other application layer protocols, most notably Hypertext Transfer Protocol (HTTP) or Simple Mail Transfer Protocol (SMTP), for message negotiation and transmission.

Software Component [derived from component definition of http://www.opengeospatial.org/resources/?page=glossary]

Software program unit that performs one or more functions and that communicates and interoperates with other components through common interfaces.

Source System

Container of unstructured, semi-structured or structured data and/or a provider of functions in terms of services. The source systems are of very heterogeneous nature and contain information in a variety of types and formats.

Support Service

Service that facilitates the operation of an architecture or system oriented service, e.g. providing an added value by combining the usage of Info-Structure Services.

System [ISO/IEC 10746-2:1996]

Something of interest as a whole or as comprised of parts. Therefore a system may be referred to as an entity. A component of a system may itself be a system, in which case it may be called a sub-system.

Note: For modelling purposes, the concept of system is understood in its general, system theoretic sense. The term "system" can refer to an information processing system but can also be applied more generally.

System User

Provider of services that are used for an application domain as well as IT architects, system developers, integrators and administrators that conceive, develop, deploy and run applications for an application domain.

Terminal

Terminals are a mobile device that is capable of running mobile services and/or mobile applications.

Use case

A common definition of use cases is the one described by Jacobson (Jacobson et al (1995) [36]): "*When a user uses the system, she or he will perform a behaviourally related sequence of transactions in a dialogue with the system. We call such a special sequence a use case*". In Other words, a use case is a textual presentation or a story about the usage of the system told from an end user's perspective.

The use cases provide some tools for people, with different skills (e.g. software developers and non-technology oriented people), to communicate with each other. The use cases are general descriptions of needs or situations that often are related to basic scenarios and that are independent of the technologies and implementations of the underlying system.

User

Human acting in the role of a system user or end user of the service and system.

WADL

The Web Application Description Language is a machine-readable XML description of HTTP-based web applications (typically REST web services) WADL models the resources provided by a service and the relationships between them. WADL is intended to simplify the reuse of web services that are based on the existing HTTP architecture of the Web. It is platform and language independent and aims to promote reuse of applications beyond the basic use in a web browser.

Web Service

Self-contained, self-describing, modular service that can be published, located, and invoked across the Web. A Web service performs functions, which can be anything from simple requests to complicated business processes. Once a Web service is deployed, other applications (and other Web services) can discover and invoke the deployed service.

W3C Web Service [W3C, http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/#webservice]

Software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

4 References

The following references are used as background documents for the preparation of this document. References are categorized standards (i.e. standards and specifications from the consortium working groups or alliances and specifications or drafts standardization bodies) and other documents, publications and technical or scientific books.

[1]	FIspace project. FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics. Deliverable D200.1 *"FIspace Design and Release Plan"*, 2014.
[2]	FIspace project. FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics. Deliverable D200.2 *"FIspace Technical Architecture and Specification"*, 2014
[3]	FIspace project. FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics. Deliverable D200.3 *"FIspace Integrated Release V1"*, 2014
[4]	FIspace project. FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics. Deliverable D200.4 *"FIspace Development Progress Report and V1 Updates"*, *2014*
[5]	FIspace project. FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics. Deliverable D200.5 *"FIspace Integrated Release V2"*, 2014.
[6]	FIspace project. FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics. Deliverable D200.6 *"FIspace Development Progress Report and V2 Updates"*, *2014*
[7]	FIspace project. FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics. D200.5 Annex *"FIspace Front-End User Guide"*
[8]	FIspace project. FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics. D200.5 Annex *"FIspace SDK User and Developer Guide"*
[9]	FIspace project. FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics. Deliverable D200.7 *"FIspace Integrated Release V3"*, 2014.
[10]	FIspace project. FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics. D200.7 Annex *"FIspace Front-End User Guide"*
[11]	FIspace project. FIspace: Future Internet Business Collaboration Networks in Agri-Food, Transport and Logistics. D200.7 Annex *"FIspace SDK User and Developer Guide"*

[12]	FIspace Business collaboration web site. http://www.fispace.eu/
[13]	FIspace Developer Documentation web site. http://dev.fispace.eu/doc/wiki/Home
[14]	FIspace Deliverables web site. http://www.fispace.eu/deliverable.html
[15]	FIspace Tutorial web site. http://www.fispace.eu/tutorials.html
[16]	FIspace Front-End Users Information web site. http://dev.fispace.eu/doc/wiki/gui
[17]	FIspace Front-End User Guide web site. http://dev.fispace.eu/doc/wiki/gui/gui-guide
[18]	FIspace App Developer Intro web site. http://dev.fispace.eu/doc/wiki/App%20Developer%20Intro
[19]	FIspace SDK Guide web site. http://dev.fispace.eu/doc/wiki/sdk
[20]	FIspace apps for newbies web site. https://bitbucket.org/fispace/apps/wiki/FIspace%20apps%20for%20newbies
[21]	FIWARE web site. http://www.fi-ppp.eu/projects/fi-ware/
[22]	FIWARE Catalogue of the Generic Enablers (GEs). http://catalogue.fi-ware.org/
[23]	FIWARE community web site. http://www.fi-ware.org/community/
[24]	FIWARE - Catalogue - Application Mashup – Wirecloud web site. http://catalogue.fi-ware.org/enablers/application-mashup-wirecloud
[25]	FIWARE - Catalogue - Store – Wstore web site. http://catalogue.fi-ware.org/enablers/store-wstore
[26]	Eclipse web site. https://www.eclipse.org/ , https://www.eclipse.org/downloads/
[27]	Maven web site. http://maven.apache.org/ , http://maven.apache.org/download.cgi
[28]	RabbitMQ web site. http://www.rabbitmq.com/
[29]	Express web site. http://expressjs.com/
[30]	ACSI EU-project web site. http://www.acsi-project.eu/
[31]	Proton - CEP recorded webinar and tutorial. http://edu.fi-ware.eu/course/view.php?id=58 Proton - CEP user and programmers guide. http://forge.fi-ware.org/plugins/mediawiki/wiki/fiware/index.php/CEP_-

	_User_and_Programmer_Guide
[32]	Keycloak JBoss project. http://keycloak.jboss.org/ Keycloak JBoss documentation. http://keycloak.jboss.org/docs
[33]	Django. high-level Python Web framework. https://www.djangoproject.com/
[34]	MongoDB. Open-source document database, NoSQL database. http://www.mongodb.org/
[35]	Powell, D. (Ed.) (1991). Delta-4: A Generic Architecture for Dependable Distributed Computing. Re-search Reports ESPRIT. Project 818/2252 Delta-4 Vol.1. ISBN 3-540-54985-4 Springer-Verlag 1991.
[36]	Jacobson, I., Bylund, S., Jonsson, P., and Ehneboom, S. (1995), “Mod-eling with Use Cases: Using contracts and use cases to build plugable architectures”. Journal of Object Oriented Programming, Vol. 8, No. 2, pp. 18-24.

FIspace
Business Collaboration