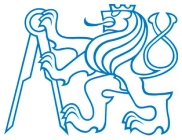

Software Requirements Specification

for project

Semantic Pipelines Editor

by

Artem Kelpe, Yan Doroshenko

Czech Technical University in Prague

November 24, 2016

Contents

1 Introduction

1.1 Purpose

This document describes the Semantic Pipelines Editor software, a specialized graph editor for the Semantic Pipelines framework, as of version 1.0.

1.2 Intended Audience

1.3 References

JSON-LD Primer
RDF Primer
P. Křemen, Z. Kouba, Ontology-Driven Information System Design
ESWC 2016, JOPA: Efficient Ontology-based Information System Design

2 Overall Description

2.1 Product Scope

Semantic Pipelines Editor is an application, whose purpose is to be integrated with Semantic Pipelines framework and to provide a graphical editor in form of an oriented graph for it with a possibility to execute pipelines from the editor itself. It is also capable of module saving and loaging, which is provided by persisting data into the ontology storage. The main goal is to provide a free and open source software edition tool for the abovestated framework and to enable developers to interact and manage pipelines in a self-explanatory and convinient way without having to deal with proprietary software or restrictive licenses.

2.2 Product Functions

Product functions can be divided into two groups:

- Graph CRUD operations
- Pipeline execution

Functions will be described in more detail in chapter System Feautures.

2.3 User Classes and Characteristics

There are two user roles defines:

- Unauthorized Any user that did not log in with has permissions to create new pipelines, edit and execute them, but has no read/write access to the ontology storage and therefore can not save or load pipelines.

- Authorized Logged in users can see, modify and create pipelines as well as save and load them.

Authorization capabilities are enabled by Spring Security on the backend. From the UI standpoint the difference is the following:

- There are no Save/Load features shown to an unauthorized user
- The link for unauthorization is shown to an authorized user

2.4 Operating Environment

2.5 Design and Implementation Constraints

3 External Interface Requirements

3.1 User Interfaces

Interaction with the user is provided by a web-interface, built around the JavaScript library for graph representation Sigma.js. The main window consists of two parts: left panel and a working surface for editing graphs (pic.1). Left panel contains elements for authorization, savind and loading graph and list of available types of nodes that can be added to the graph. At the working surface user can see the current graph (loaded or created), edit it (move nodes, draw edges etc.). Node double click shows the form generated from this node and its dependencies in the popup (pic. 2).

3.2 Software Interfaces

Application consists of frontend and backend parts with backend accessing the ontology storage as well as a Semantic Pipelines webservice. Communication between backend and frontend is implemented with JSON-LD format messages sent through REST API. All communication is done through unsecured HTTP protocol. Authentication encryption is provided by integrated SpringSecurity tools.

4 System Features

As mentioned before, there are 5 main functions of Semantic Pipelines Editor software.

4.1 See pipelines

Pipelines are represented in form of an oriented graph. Nodes represent modules and edges are dependencies between them, which makes for intuitive and functional user interface.

4.2 Generate form (execute pipeline)

Semantic Pipelines Editor enables semantic forms to be generated from the pipeline. Module dependencies are executed recursively.

4.3 Create pipelines

There exists a possibility for creating new pipelines from scratch by adding individual modules and dependencies between them.

4.4 Save/load pipelines

The software allows saving pipelines into the ontology storage and loading them.

4.5 Alter pipelines

Pipelines can be modified by adding or deleting modules, changing their properties and dependencies between them.

5 Other Non-functional Requirements

6 Other Requirements

Appendix A: Glossary

Appendix B: Analysis Models