



Supply Chain 2 Map

Documentation

Group Members

Mirco Sprung

Mohammad Ali Ghasemi

Sina Mahmoodi

Omid Najaee Nezhad

Mentor

Niklas Petersen



1 Introduction

1.1 Goals

1.2 Project Scope

2 About SCORVoc

2.1 Definition : a Vocabulary based on the Supply Chain Operation Reference Model

2.2 How it works

2.3 Applying to Project

3 System Architecture and Technical Description

3.1 Introduction

3.1.1 Purpose

3.1.2 Scope

3.1.3 Definitions, Acronyms and Abbreviations

3.2 Overall Technical Description

3.2.1 Product Perspective

3.3 Specific Requirements

3.3.1 External Interface Requirements

Hardware Interfaces

Communication Interfaces

Software Interfaces

Ontology

3.3.2 Functional Requirements

3.3.3 Non-functional Requirements

3.3.4 Design Constraints

Memory Usage

Disk Usage

Software Licensing

Immutability

3.3.4 Architectural Design

4 User Manual

4.1 Different parts

4.2 Metrics definition

4.3 How it works

4.3.1 Overview

4.3.2 General Information

4.3.3 Filters

4.3.3 Processes and Metrics

4.3.3.1 Processes

4.3.3.2 Metrics

4.3.3.2 Full example

References

Supply Chain 2 Map

September 14, 2016

1 Introduction

The purpose of this Software Requirements Specification (SRS) document is to provide a detailed description of the functionalities of the SupplyChain2Map system. This document will cover each of the system's intended features, as well as offer a preliminary glimpse of the software application's User Interface (UI). The document will also cover hardware, software, and various other technical dependencies.

1.1 Goals

The main aim of SC2MAP project is to make an easy approach to represent of inter-relation between nodes (entities) according our ontology and data provided as before. As it is mentioned Map is used to represent the raw data (OpenStreetMap). Therefore in the final product it will be shown that the connection in terms of supplying products are linked by colored lines which give visually an overview about a complete supply chain to the user. In this project we have a sample ontology to represent the data on the map. However, it is possible to work with other sources.

1.2 Project Scope

Enterprises often have to deal with the management of their supply chain, to coordinate the flow of goods and services, from the producer to the consumer. Albeit, if done right, this task can help integrate different enterprises together better and improve efficiency, it is usually done manually, which is a complex and costly task. SupplyChain2Map, as a part of SCIAApp, tries to facilitate the automation of this task by visualizing the supply chain of enterprises, embedded with some analytical data about the flow, such as the delivery rate of routes.

2 About SCORVoc

2.1 Definition : a Vocabulary based on the Supply Chain Operation Reference Model

Advancing highly specialized economies require instant, robust and efficient information flows within its value-added and supply chain networks. The Supply Chain Operation Reference (SCOR) is a cross-industry approach to lay the groundwork for this goal by defining a conceptual model. In this paper, we describe the SCORVoc vocabulary which represents SCOR entirely as an RDF vocabulary. In order to operationally use the vocabulary for analyzing, monitoring and optimizing supply chains (in particular for robustness), we present SPARQL queries, which retrieve the respective information from information systems adhering to the vocabulary. We define concrete test scenarios and thus demonstrate the practicality of SCORVoc.

2.2 How it works

Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius.

2.3 Applying to Project

Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius. Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius.

3 System Architecture and Technical Description

3.1 Introduction

3.1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to provide a detailed description of the functionalities of the SupplyChain2Map system. This document will cover each of the system's intended features, as well as offer a preliminary glimpse of the software application's User Interface (UI). The document will also cover hardware, software, and various other technical dependencies.

3.1.2 Scope

Enterprises often have to deal with the management of their supply chain, to coordinate the flow of goods and services, from the producer to the consumer. Albeit, if done right, this task can help integrate different enterprises together better and improve efficiency, it is usually done manually, which is a complex and costly task. SupplyChain2Map, as a part of SCIApp, tries to facilitate the automation of this task by visualizing the supply chain of enterprises, embedded with some analytical data about the flow, such as the delivery rate of routes.

3.1.3 Definitions, Acronyms and Abbreviations

- **SC2M** SupplyChain2Map, which is being described in this document
- **SCOR** The Supply Chain Operations Reference model (SCOR) is the world's leading supply chain framework, linking business processes, performance metrics, practices and people skills into a unified structure.
- **SCORVoc** An OWL vocabulary which fully formalizes the latest SCOR standard, while overcoming identified limitations of existing formalizations.
- **W3C** The World Wide Web Consortium is the main international standards organization for the World Wide Web.

- **HTTP** The Hypertext Transfer Protocol (HTTP) is an application protocol for distributed, collaborative, hypermedia information systems.
- **SPARQL** The SPARQL Protocol and RDF Query Language (SPARQL) is a query language and protocol for RDF.
- **RDF** The Resource Description Framework (RDF) is a general-purpose language for representing information in the Web.
- **ES6 (ES2015)** ECMAScript (or ES) is a trademarked scripting-language specification standardized by Ecma International, and ES6 is the 6th version of this specification. One of the most popular implementations of this standard is Javascript.
- **Babel** Babel is a compiler which has support for the latest version of JavaScript through syntax transformers. This comes into play as not all the features of ES6 are implemented in current browser versions. Babel fills that gap.

3.2 Overall Technical Description

3.2.1 Product Perspective

At the core of SupplyChain2Map a Javascript library resides which fetches necessary data from databases (triplestores) and after doing some transformations shows it on a map for the user to see.

In this sense, the project is divided into a client-side and a server-side. The server-side keeps the supply chain data for enterprises adhering to a standard ontology and exposes it to the client. The ownership of the server might be in the hands of the enterprise itself. In turn, the client fetches the necessary parts of data, caches it in the browser, and executes queries on it upon need.

3.3 Specific Requirements

3.3.1 External Interface Requirements

Hardware Interfaces

SC2M is intended as a JS library, therefore the hardware which the client can run on is not very limited, as various hardware devices support a display screen and can be made to run a browser and therefore use the library. Examples are PCs and handheld devices. The data fetching part requires a network card connected to the the internet.

The server-side requires a server, preferably geared towards being a data store with features such as high speed disks and large amounts of memory. The server requires a network card connected to the internet.

Communication Interfaces

Most of the communication happens between the client and the server(s), through client requesting enterprise data from the data stores. The two communication protocols which are used, are HTTP and SPARQL, which itself uses HTTP. HTTP works on top of the TCP/IP stack.

The data transferred using these protocols are either SPARQL queries and results as defined in the SPARQL Protocol Document by W3C, or JSON documents for every other request.

Software Interfaces

The JS library depends on the API provided by the browsers (or Babel) for normal functions such as accessing the user's screen size or accessing the LocalStorage to cache RDF triples. For visualizing the supply chain of an enterprise, SC2M depends on the interface provided by Leaflet, which in turn

is a JS library. Furthermore, the SPARQL Protocol and Query language are used to access and modify the data.

Ontology

The SCOR framework provides many different definitions for expressing enterprise data, including various processes which might occur in a supply chain. Since the data SC2M will use is in form of RDF triples, it depends upon SCORVoc to shape the SCOR framework in terms of concept hierarchies in RDF.

3.3.2 Functional Requirements

ID	Requirement Statement	Must/Want	Comments
FR001	The user shall import data	Must	The user should be able to either import a set of data, or choose an enterprise from the existing data set.
FR002	The website shall have a world map.	Must	
FR003	The map shall be interactable.	Must	Providing general map interaction like zooming or dragging.
FR004	SC data shall be visualized on the map.	Must	
FR005	The nodes on the map shall be selectable.	Must	Selection in order to get additional information about the node.
FR006	The website shall provide analysis functionality over the SC data.	Must	Given already existing algorithms and metrics.

3.3.3 Non-functional Requirements

ID	Requirement Statement	Must/want	Comments
NFR001	Modularity	Must	Different segments of the the product should be architected in a way to interact with each other in a standard and independent fashion.
NFR002	Maintainability	Want	The library is to be architected in a developer-friendly approach in mind. Is crucial for the library to be easily extendable by everyone, not only the initial developers group..
NFR003	Performance	Want	The product is to be reasonably operating on the inputs, regardless of size and complexity.
NFR004	Usability	Want	Due to the focus importance of graphical interface in this project, the app should be as easy and intuitive as possible for the user to interact and learn
NFR005	Accessibility	Must	The product should be possible to run on the latest version of modern browsers (i.e. Google Chrome, Firefox, Edge)
NFR006	Security	Want	Enterprises might want to own their own data, which brings about the question of authentication and secure communication to prevent enterprise data leak.
NFR007	Testability	Want	The functionalities within the source-code should be written in a way that is comprehensively verifiable by an automatic testing suit with a high code test coverage.
NFR008	Documentation	Must	The documentation for each of the elements

	Requirement		of the library should be written and compiled in an easy to use manner. The remarks should be clear and well-organized. The intention of the documentation should be to enable third-party developers easily understand the way to work with the library by reading the documentation.
NFR009	Acceptance requirements	Must	The library should display the location of the supply chain elements accurately and clearly.

3.3.4 Design Constraints

Memory Usage

The final product being a Javascript library which runs on the browser, there are limitations on the amount of memory which could be consumed by the library. These limitations may be due to browser policies, or performance. Since web applications which consume a large chunk of memory tend to perform slower. Besides, since there's no installation of web application, the library can't determine the total or free amount of memory on the user's computer. Analysing the available web applications led to an estimated cap of 200 MB.

Disk Usage

For the aforementioned reasons, and the fact that web applications do not have direct access to the client file system and can only use disk through the LocalStorage API provided by the browser, put a restrictive limit upon disk usage. Disk I/O is usually problematic because it's non-deterministic for a multitude of reasons. A simple disk operation can take anywhere from zero milliseconds to a few seconds. Browsers cope with this by preloading the entire Local Storage key/value store into memory on first request, which makes the limitation for applications even more sensitive. Therefore, the library must use lazy loading methods to fetch the data, only when it is needed, and discarding it directly afterwards to save disk and memory.

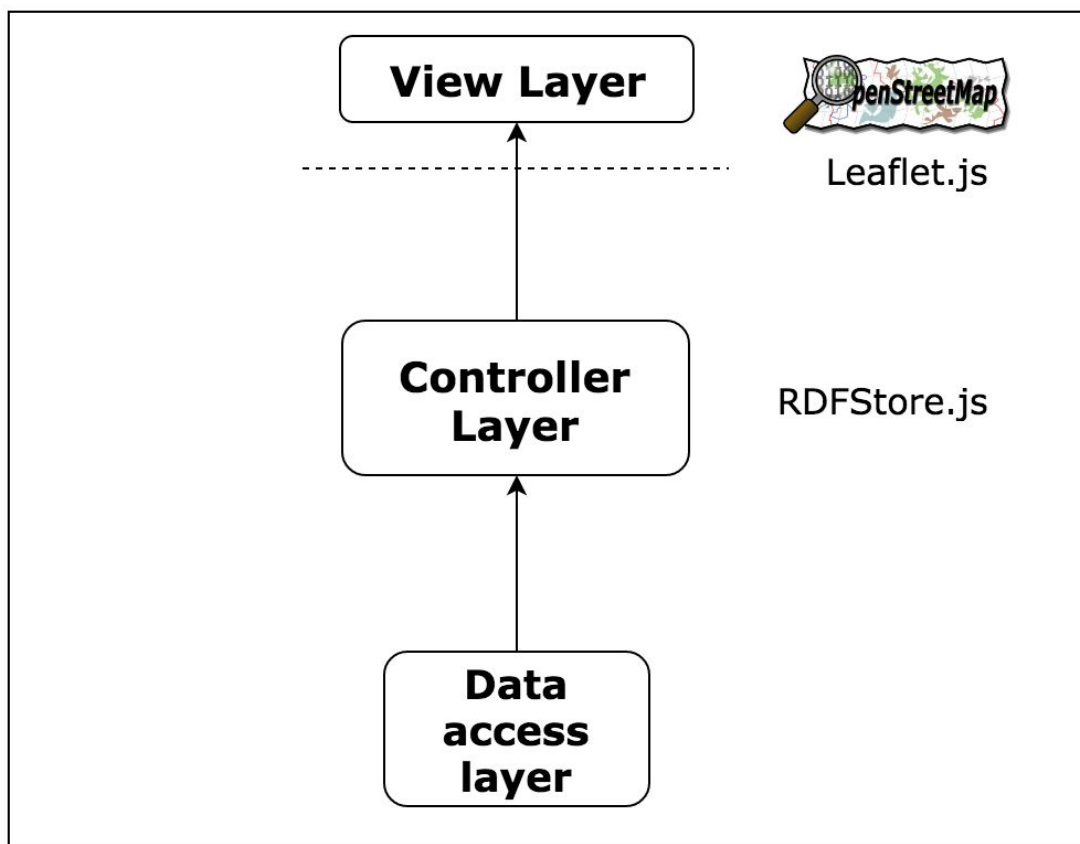
Software Licensing

The project depends upon a variety of open source softwares which are under different licenses. As an example, the GNU GPLv3 license requires anyone who distributes a code or a derivative work to make the source available under the same terms. Therefore, the license of the final library, must adhere to and be compatible with the licenses of the tools and libraries being used.

Immutability

The aim is to reduce the amount of mutability in the source code. This means that by declaring variables in a way which must be. For instance declaring constants, public and private variables. We should also focus on conventions in coders community which make our software capable to be interested by aforementioned community.

3.3.4 Architectural Design



4 User Manual

4.1 Different parts

In this part we are going to show different part of application and show the functions to the user. Then we address some specifications and show how it works as general.

4.2 Metrics definition

The main contribution of SCOR was to provide enterprises with a structure to express their supply chains. However, the key motivation is that once enterprises accomplish this task, these supply chains become comparable. In an attempt to analyze, in a standardized way, certain aspects of a supply chain, SCOR further defined metrics. Therefore, we consider the usage of these metrics as our main motivation for building this vocabulary. There are 286 metrics in total, grouped together into five categories: Reliability, Responsiveness, Agility, Cost and Assets. The usage of these metrics allow supply chain managers to identify weak and strong links within the supply chain.

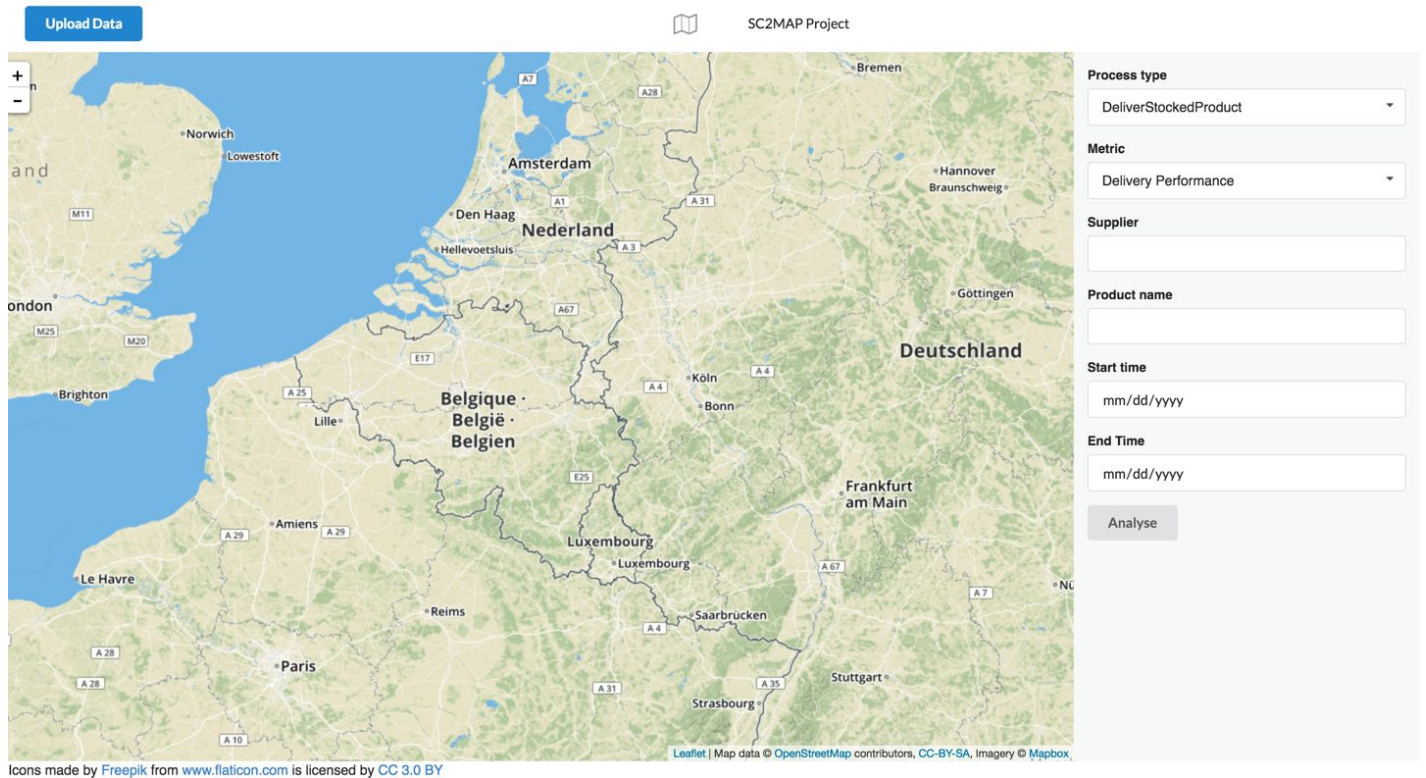
4.3 How it works

4.3.1 Overview

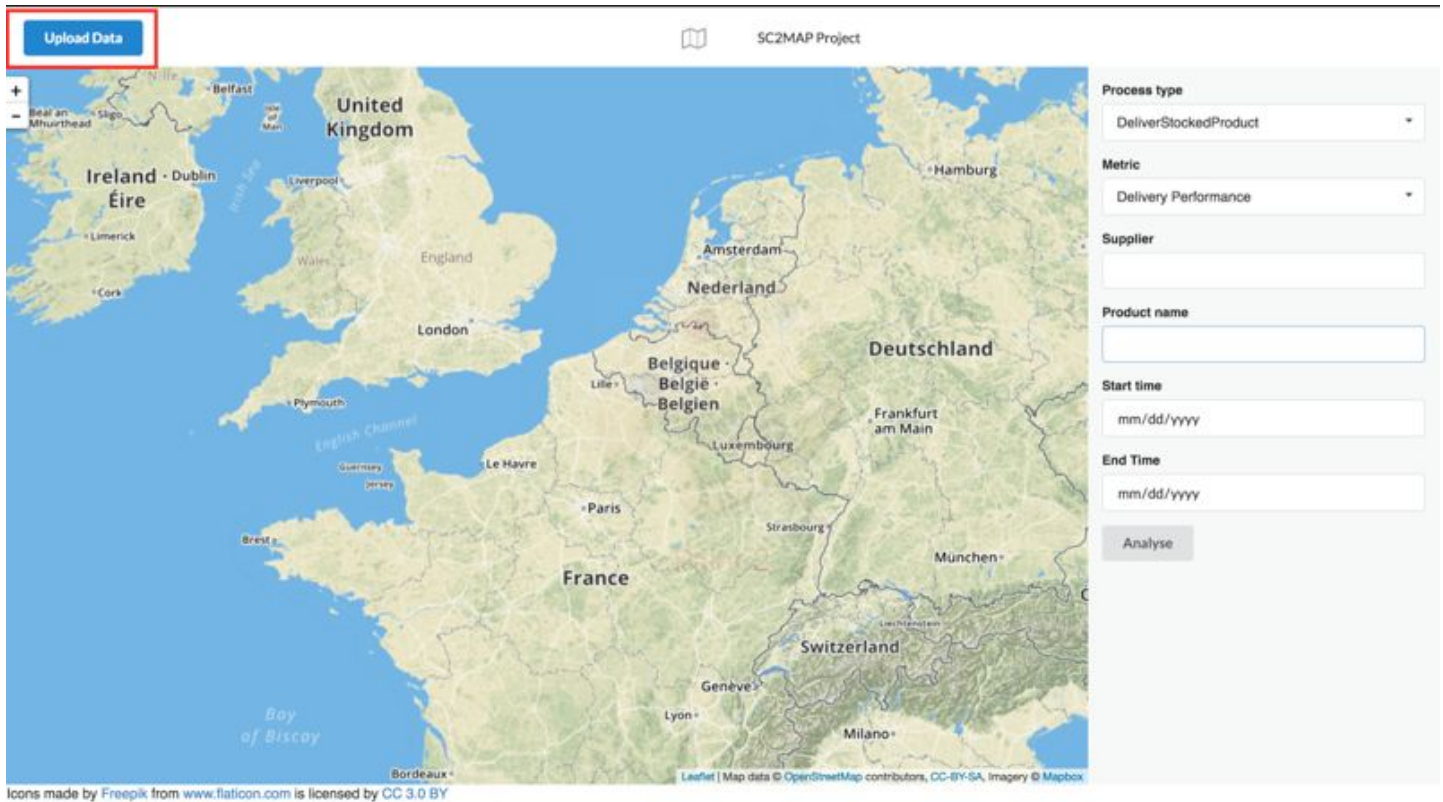
We assume the state that system is ready to use. Therefore, users' point of view should be considered in this manual description. The tutorial shows different parts of the application with related pictures to address each part to get familiar with the app.

4.3.2 General Information

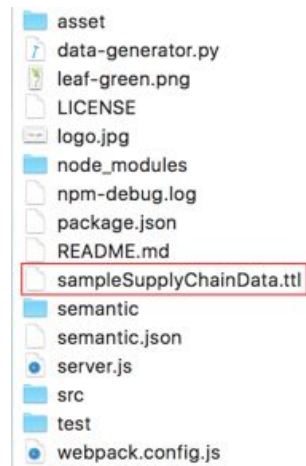
By opening the project in the browser (Chrome, Firefox, Safari and Microsoft edge browser) you will face with the starting point of the project:



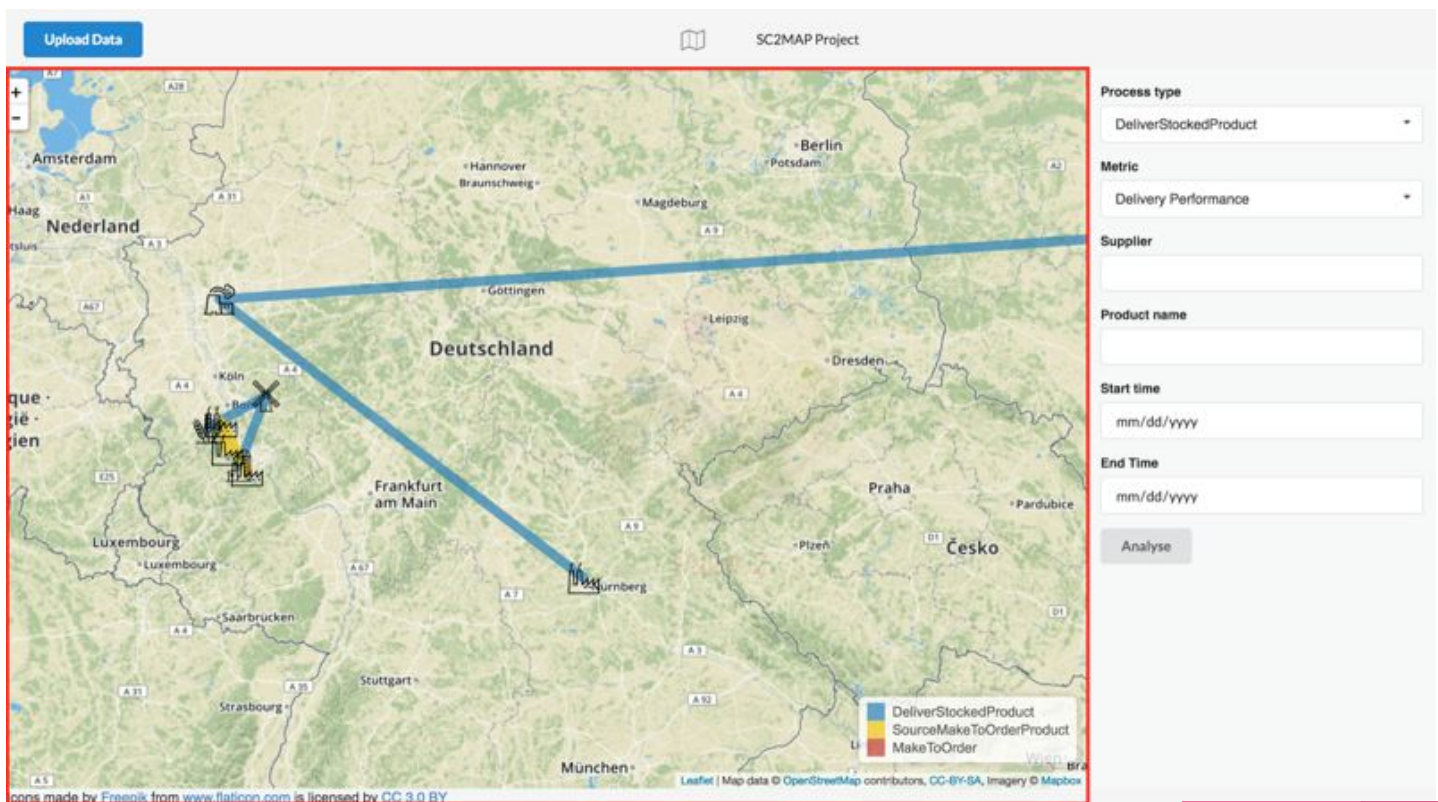
As it is shown in the picture, there is a map powered by OpenStreetMap and in the right side there is a panel which user can define Metrics, Production Name, Supplier and Time span to filter the result. In the up-left corner there is a button which loads the Ontology which will be processed:



Then a file picker will be shown a to load the ontology file with *.ttl extension. In this case we used the sample ontology file:

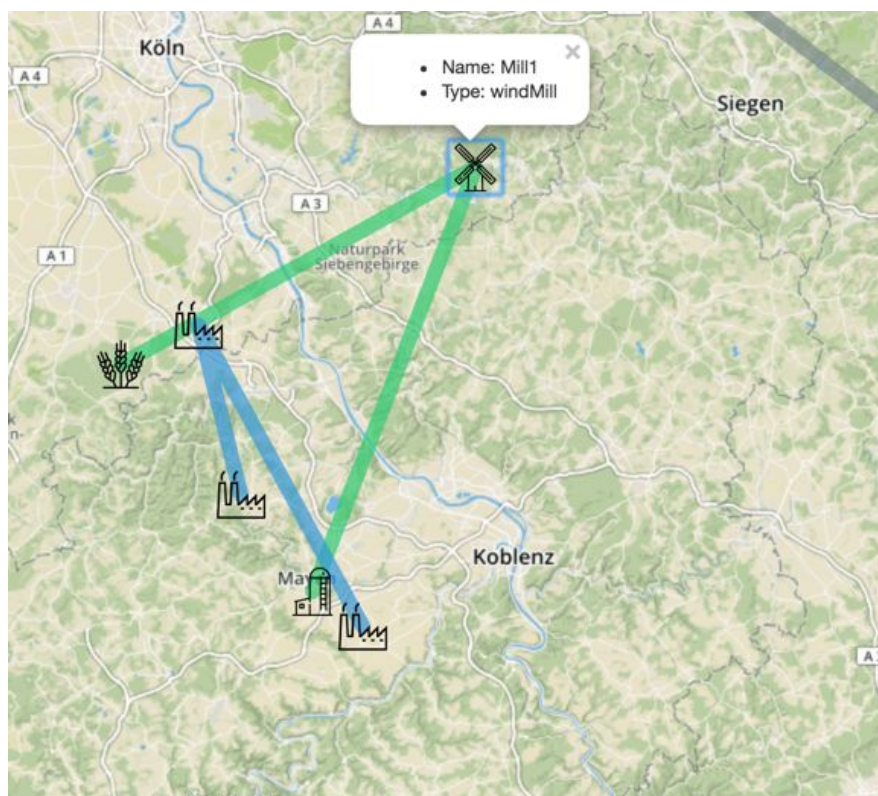


In case of uploading the file successfully, you can see the entities and their relations by links which show different type of information:

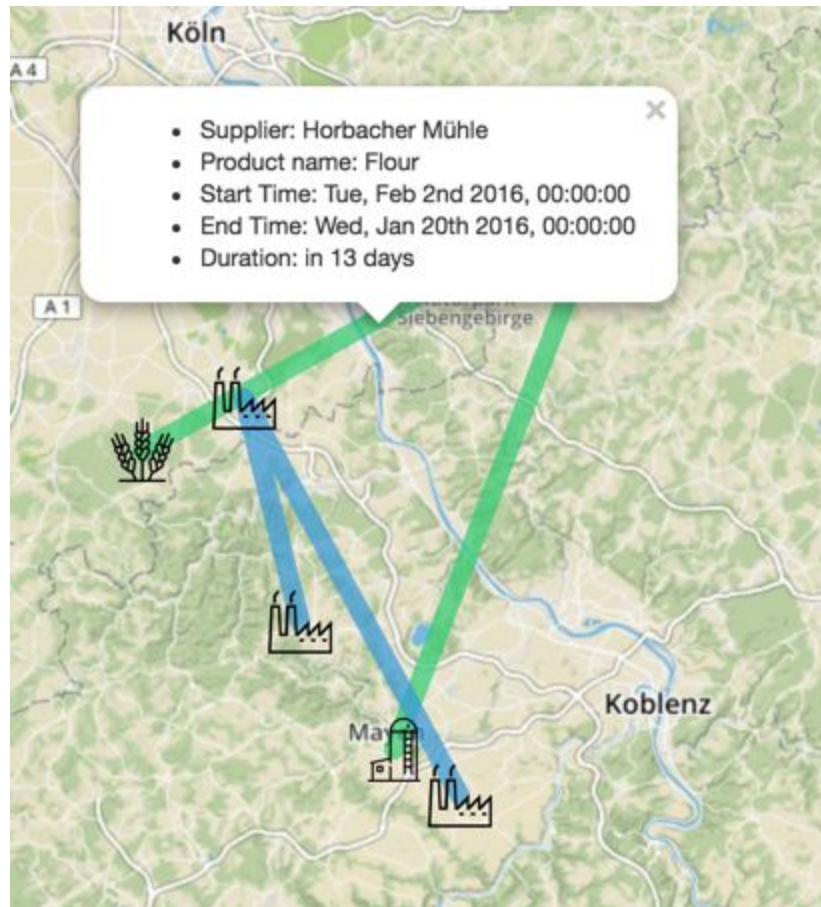


In this simplified example you can see the relationship between different type of entities like factory, power plant, silo etc.

By clicking on each node you can see node's detail like its name and also the type which is categorized in the ontology:



There is also possibility to show fully detailed information for supplier which is handling in current chain between two nodes. By clicking at lines you will get a pop-up panel which shows information like Supplier's name, Production, Start time, End time and duration:



4.3.3 Filters

According different parameters the app is able to filter the result. For example by defining the time period which is desired from user, the map will show different result.



The screenshot shows a filter interface with two sections. The first section is titled 'Start time' and contains a text input field with the value '01/02/2016'. To the right of the input field are three icons: a close button (X), a calendar icon, and a dropdown arrow. The second section is titled 'End Time' and contains a text input field with the value '01/20/2016'.

Then it is possible to use different parameters which is shown here:



The screenshot shows a filter interface with two sections. The first section is titled 'Supplier' and contains a text input field. The second section is titled 'Product name' and contains a text input field.

There is two parameters which is customizing the filtering in a way. Supplier's name and product name are two key parameters.

4.3.3 Processes and Metrics

There are defined three different process type which analyze the data according of it:

Process type

DeliverStockedProduct ▼

DeliverStockedProduct

SourceMakeToOrderProduct

MakeToOrder

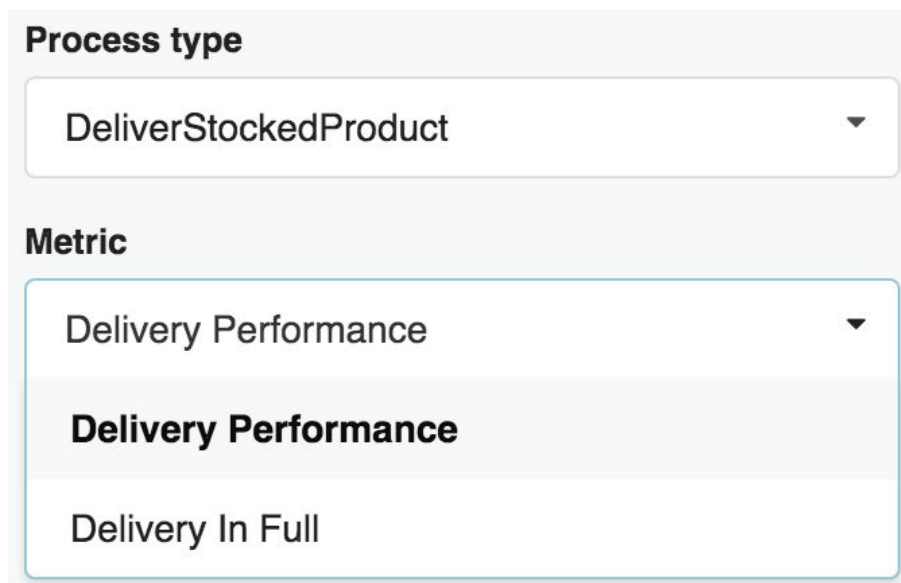
4.3.3.1 Processes

- **Delivery Stocked Product:** Delivers product that is sourced or made based on aggregated customer orders, projected orders/demand and inventory re-ordering parameters

- **Source Make To Order Product:** The processes of ordering and receiving product or material that is ordered (and may be configured) only when required by a specific customer order.
- **Make To Order:** The process of manufacturing in a make-to-order environment adds value to products through mixing, separating, forming, machining, and chemical processes for a specific customer order.

4.3.3.2 Metrics

Then according the process type which is selected related Metrics are listed in the Metrics Dropdown Box. For example: by selecting DeliverStockedProduct, related Metrics are listed which are:



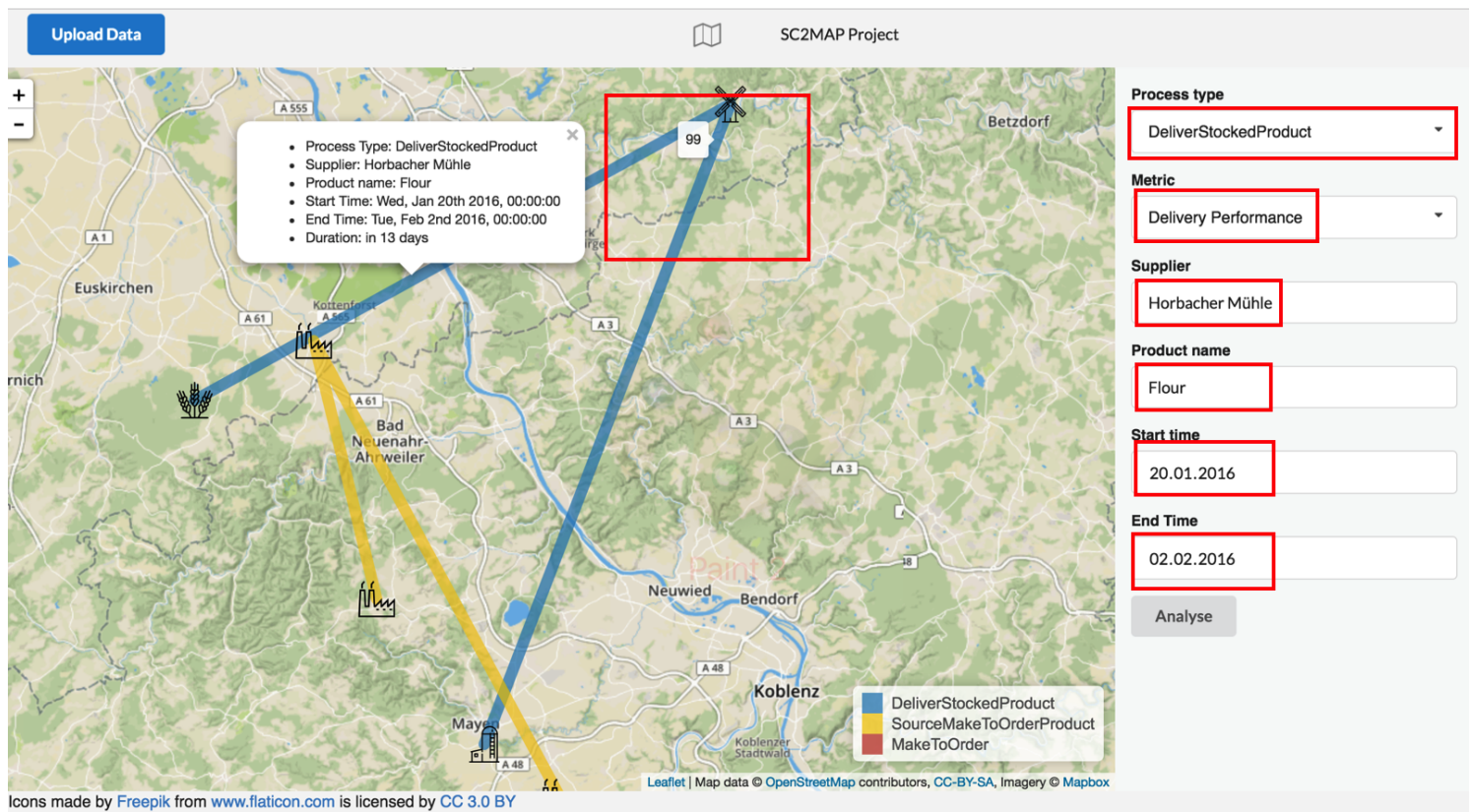
The screenshot displays a user interface for selecting metrics based on a process type. It consists of two main sections: 'Process type' and 'Metric'. The 'Process type' section has a dropdown menu with 'DeliverStockedProduct' selected. The 'Metric' section has a dropdown menu with 'Delivery Performance' selected, and a list of three metrics is shown below it: 'Delivery Performance' (highlighted), 'Delivery In Full', and 'Delivery In Full'.

Process type
DeliverStockedProduct

Metric
Delivery Performance
Delivery Performance
Delivery In Full

4.3.3.2 Full example

Here is an example which shows how the app works with multiple filters:



References

1. http://eis.iai.uni-bonn.de/upload/paper/ISWC2015dataonto_submission_7.pdf

