

Creating a semantic portal easily using Sampo-UI

Annastiina Ahola, Heikki Rantala

Sampo-UI framework

- **Framework for developing user interfaces for semantic portals**
 - multiple application perspectives to the data
 - two-step usage cycle: filter & analyze
 - has been used both in the Sampo series of portals as well as external projects
- **Available on GitHub under the MIT license:**
<https://github.com/SemanticComputing/sampo-ui>

Philosophy

- **Faceted searching done with SPARQL**
 - Sampo-UI doesn't require your triple store to implement a certain kind of text indexing to work
 - *Text indexing can be utilized for efficient text-based search*
- **Configuration primarily done through JSON configuration and SPARQL query files without having to touch JavaScript**

Sampo-UI “standard” UI model

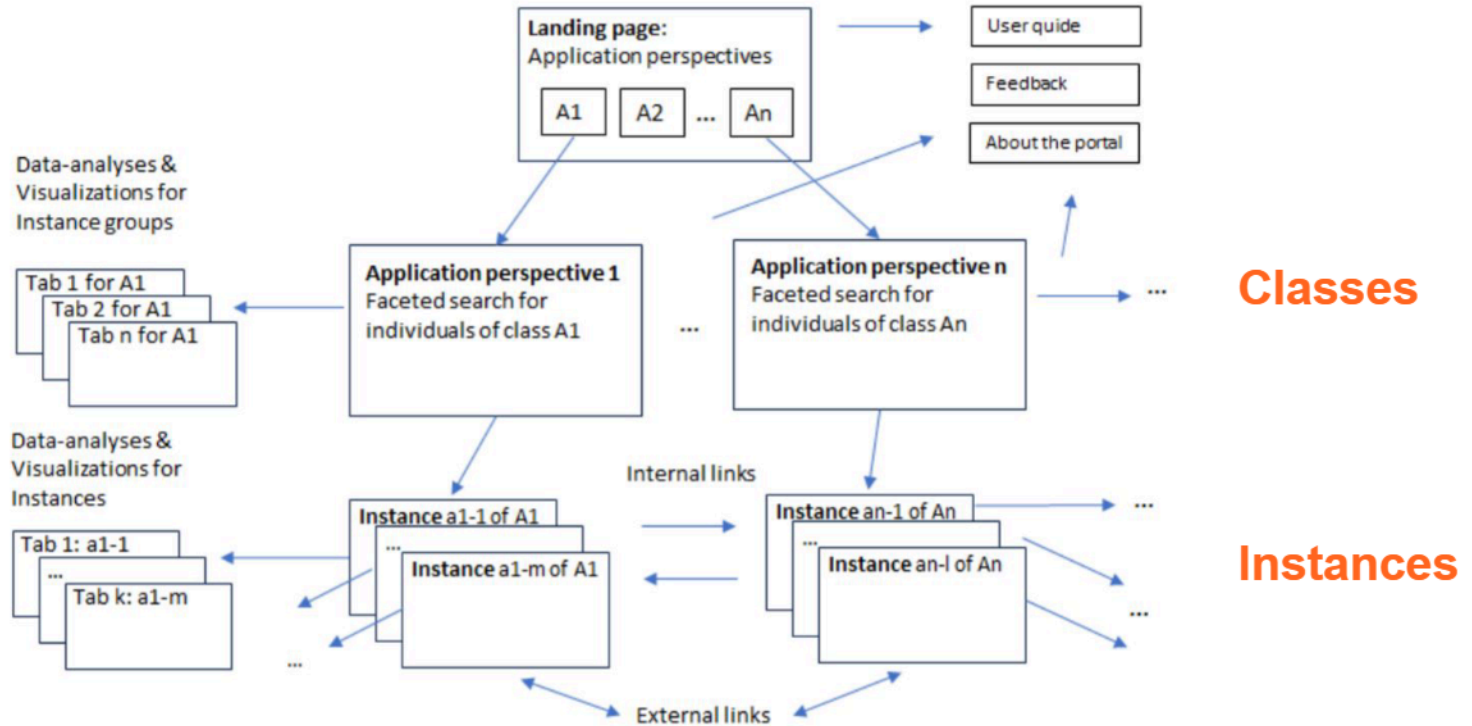


Figure 1: Navigational page structure of a portal based on Sampo-UI.

Sampo-UI framework: Structure

The screenshot shows the Sampo-UI DBpedia demo interface. At the top left is the Sampo-UI logo and the text 'Sampo-UI DBpedia demo'. At the top right is a navigation menu with links for 'WRITERS', 'BOOKS', 'FEEDBACK', 'INFO', 'INSTRUCTIONS', and 'EN'. Below the navigation is a large banner image of old books with the title 'Sampo-UI DBpedia demo' and subtitle 'Sampo-UI demo using DBpedia data'. Below the banner is the instruction 'Select a perspective to search and browse the knowledge graph:'. There are two buttons: 'Writers' with the subtitle 'A perspective for browsing and searching writers' and 'Books' with the subtitle 'A perspective for browsing and searching books'. Below the buttons is a small text: 'images used under license from Shutterstock.com'.

Multiple application perspectives to the data

Sampo-UI framework: Structure

Sampo-UI DBpedia demo

WRITERS BOOKS FEEDBACK INFO INSTRUCTIONS EN

Writers

Results: 51821 writer(s)

Narrow down by:

Genre

Search...

- Unknown [43921]
- Science fiction [972]
- Fantasy [741]
- Children's literature [539]
- Poetry [483]
- Romance novel [468]
- Fiction [419]
- Horror fiction [388]
- Young adult fiction [339]
- Mystery fiction [321]
- Children's fiction [290]

Occupation

Alma mater

Birth country

Death country

Visualization tabs for visualizing the data

TABLE

PIE/BAR CHART

PUBLICATION YEARS

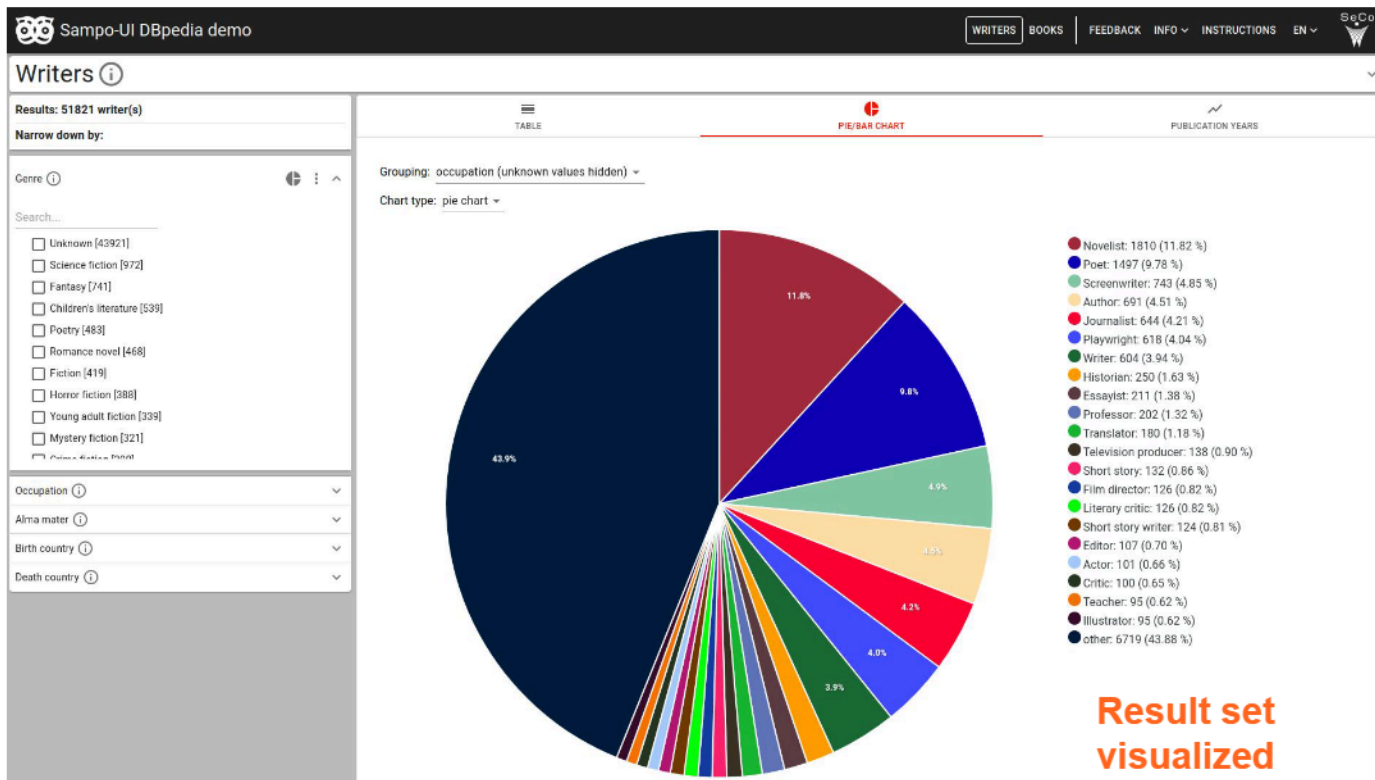
Rows per page 25 1-25 of 51821

Name	Genre	Occupation	Alma mater
Garden Manson	-	-	-
Cahit Saki Tarancı	-	Interpreter ...	-
Cahit Zaimoğlu	-	-	-
Cai Chanying	-	-	Quanzhou Normal University
Cai Dongfan	Historical fiction	-	-
Cai Lujun	-	-	-
Caiseal Mór	-	Artist ...	-
Cairín Davies	-	-	-
Cairín Kimmidge	-	-	-
Cairín Rother	Non-fiction	Journalist	-
Cairín Thomas	-	-	-
Cairín R. Kiernan	Dark fantasy ...	-	-
Cairiona O'Reilly	-	-	-
Cairier Williamson	-	-	-
Caleb Carr	-	-	New York University
Caleb Whiteford	-	-	Edinburgh University
Callie Khoufi	-	Film producer	-
Callimachus	Aetiology ...	-	-
Calvert Watkins	-	-	-
Camerina Pavlin v Ovišed	-	-	-
Cameron Litvack	-	-	-
Cameron Rojcew	Speculative fiction	-	-
Camil Petrescu	-	Novelist ...	University of Bucharest
Camilla Caluso	-	Novelist	-
Camilla Gibb	-	Novelist	American University in Cairo ...

Facet menu for filtering the data

Results

Sampo-UI framework: Structure



Written & video tutorial

- **Tutorial showing the steps of setting up Sampo-UI using DBpedia's endpoint/data**
 - In video format:
<https://vimeo.com/817028609>
 - Written tutorial:
<https://seco.cs.aalto.fi/tools/sampo-ui/Sampo-UI-tutorial.pdf>
 - Code from the tutorial available on GitHub:
<https://github.com/SemanticComputing/dbpedia-sampo-ui-demo>

Requirements

- **Node.js (<https://nodejs.org/en/>) version 16.13.0**
 - Recommended: Node Version Manager (nvm) for managing different versions of Node.js installed on your computer
- **Nodemon (<https://nodemon.io/>)**
- **Optional: Docker**
 - You can also run Sampo-UI inside a Docker container on your computer, but this is not required for development

Getting started

- Fork or download source code from GitHub

The screenshot displays the GitHub interface for the repository `SemanticComputing/sampo-ui`. At the top, there are navigation links for Product, Solutions, Resources, Open Source, Enterprise, and Pricing. A search bar and 'Sign in' / 'Sign up' buttons are also present. Below the repository name, there are buttons for Notifications, Fork (8), and Star (33). The main content area shows the repository structure with a 'Code' button highlighted in green. A list of commits is visible, with the most recent one by `annasahola` dated 5 months ago. The right sidebar contains information about the repository, including the description: 'Sampo-UI – A framework for building user interfaces for semantic portals', the license (MIT), and the current release (v2.0.0).

Configuration

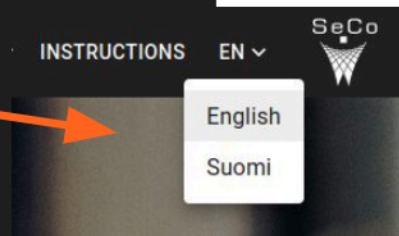
- **Configuration done through JSON configuration files**
 - a file for portal-wide configuration
 - one file for each configured perspective
- **Queries with SPARQL**
 - one file for each perspective with the perspective-specific queries for getting all wanted properties
 - many general queries already exist (e.g., getting facet values), you just need to pass the correct predicates in configuration

Portal configuration

```
{
  "portalID": "sampo",
  "rootUrl": "",
  "perspectives": {
    "searchPerspectives": [
      "writers",
      "books"
    ],
    "onlyInstancePages": [
    ]
  },
  "localeConfig": {
    "defaultLocale": "en",
    "readTranslationsFromGoogleSheets": false,
    "availableLocales": [
      {
        "id": "en",
        "label": "English",
        "filename": "localeEN.json"
      },
      {
        "id": "fi",
        "label": "Finnish",
        "filename": "localeFI.json"
      }
    ]
  },
  "sitemapConfig": {
    "baseUrl": "https://sampo-ui.demo.seco.cs.aalto.fi",
    "langPrimary": "en",
    "langSecondary": "fi",
    "outputDir": "./src/server/sitemap_generator",
    "sitemapUrl": "https://sampo-ui.demo.seco.cs.aalto.fi/sitemap",
    "sitemapInstancePageQuery": "sitemapInstancePageQuery"
  }
}
```

Configure available perspectives by ID

Configure available locales and files for getting the required translations



Perspective configuration: Basics

```
{
  "id": "writers",
  "endpoint": {
    "url": "https://dbpedia.org/sparql",
    "useAuth": false,
    "prefixesFile": "SparqlQueriesPrefixes.js"
  },
  "sparqlQueriesFile": "SparqlQueriesWriters.js",
  "baseURI": "http://dbpedia.org/resource/",
  "URITemplate": "<BASE_URI><LOCAL_ID>",
  "facetClass": "dbo:Writer",
  "frontPageImage": "main_page/works-452x262.jpg",
  "searchMode": "faceted-search",
  "defaultActiveFacets": [
    "prefLabel"
  ],
  "defaultTab": "table",
  "defaultInstancePageTab": "table",
  "resultClasses": {
    "writers": {
      "paginatedResultsConfig": {
        "tabID": 0,
        "component": "ResultTable",
        "tabPath": "table",
        "tabIcon": "CalendarViewDay",
        "propertiesQueryBlock": "writerProperties",
        "pagesize": 25,
        "sortBy": null,
        "sortDirection": null
      }
    }
  }
}
```

Perspective ID (referenced in portal configuration)

Configuration for the endpoint used in the perspective and what prefixes should be used in queries

Name of file containing the SPARQL queries for the perspective

Class of the instances included in the perspective

Variable name for result table query in query file

Perspective configuration: Queries

```
const perspectiveID = 'writers'
```

```
export const writerProperties = `
```

```
{  
  ?id rdfs:label ?prefLabel_id .  
  FILTER(LANG(?prefLabel_id) = 'en')  
  BIND(?prefLabel_id AS ?prefLabel_prefLabel)  
  BIND(CONCAT("/${perspectiveID}/page/", REPLACE(STR(?id), "^..*\\\\\\/(.+)", "$1")) AS ?prefLabel_dataProviderUrl)  
  BIND(?id as ?uri_id)  
  BIND(?id as ?uri_dataProviderUrl)  
  BIND(?id as ?uri_prefLabel)  
}
```

Queries basic label information and forms the link for the instance page

```
UNION
```

```
{  
  ?id dbo:genre ?genre_id .  
  ?genre_id rdfs:label ?genre_prefLabel .  
  FILTER(LANG(?genre_prefLabel) = 'en')  
}
```

Queries the writer's written genre and its label in English

```
UNION
```

```
{  
  ?id dbo:occupation ?occupation_id .  
  ?occupation_id rdfs:label ?occupation_prefLabel .  
  FILTER(LANG(?occupation_prefLabel) = 'en')  
}
```

Queries the writer's occupation and its label in English

```
UNION
```

```
{  
  ?id dbo:almaMater ?almaMater_id .  
  ?almaMater_id rdfs:label ?almaMater_prefLabel .  
  FILTER(LANG(?almaMater_prefLabel) = 'en')  
}
```

Queries the writer's alma mater and its label in English

```
export const writersByOccupationQuery = `
```

Perspective configuration: Properties

```
"properties": [  
  {  
    "id": "uri",  
    "valueType": "object",  
    "makeLink": true,  
    "externalLink": true,  
    "sortValues": true,  
    "numberedList": false,  
    "onlyOnInstancePage": true  
  },  
  {  
    "id": "prefLabel",  
    "valueType": "object",  
    "makeLink": true,  
    "externalLink": false,  
    "sortValues": true,  
    "numberedList": false,  
    "minWidth": 250  
  },  
  {  
    "id": "genre",  
    "valueType": "object",  
    "makeLink": false,  
    "externalLink": false,  
    "sortValues": true,  
    "numberedList": false  
  }  
]
```

Included properties (columns in the table) and their configuration are configured in the “properties” list

ID should match the name used in the SPARQL queries

Object type properties should at least have an ID (e.g., genre__id) and a label (genre__prefLabel) from the queries

Rows per page: 25 | 1-25 of 51821 | < | >

Name ⓘ	Genre ⓘ	Occupation ⓘ	Alma mater ⓘ
Caden Manson	-	-	-
▼ Cahit Sitki Tarancı	-	Interpreter ...	-
Cahit Zarifoğlu	-	-	-
Cai Chongda	-	-	Quanzhou Normal University

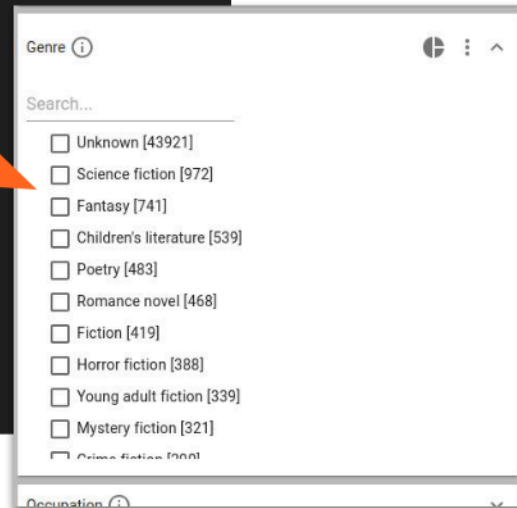
Perspective configuration: Facets

```
"facets": {  
  "prefLabel": {  
    "sortByPredicate": "rdfs:label"  
  },  
  "genre": {  
    "containerClass": "ten",  
    "facetType": "list",  
    "filterType": "uriFilter",  
    "facetLabelPredicate": "rdfs:label",  
    "facetLabelFilter": "FILTER(LANG(?prefLabel_) = 'en')",  
    "predicate": "dbo:genre",  
    "searchField": true,  
    "sortBy": "instanceCount",  
    "sortByPredicate": "dbo:genre/rdfs:label",  
    "sortDirection": "desc",  
    "pieChartButton": true  
  },  
  "occupation": {  
    "containerClass": "ten",  
    "facetType": "list",  
    "filterType": "uriFilter",  
    "facetLabelPredicate": "rdfs:label",  
    "facetLabelFilter": "FILTER(LANG(?prefLabel_) = 'en')",  
    "predicate": "dbo:occupation",  
    "searchField": true,  
    "sortBy": "instanceCount",  
    "sortByPredicate": "dbo:occupation/rdfs:label",  
    "sortDirection": "desc",  
    "pieChartButton": true  
  }  
}
```

Type for facet (e.g., checkbox, integer range, ...)

Configuration for getting the labels for values

Predicate for getting label values from instances



Perspective configuration: Visualizations

```
"resultClasses": {
  "writersByProperty": {
    "resultClasses": {
      "writersByBirthCountryWithoutUnknown": {
        }
      }
    }
  },
  "releasesLineChart": {
    "tabID": 2,
    "component": "ApexCharts",
    "tabPath": "publications_by_year",
    "tabIcon": "ShowChart",
    "sparqlQuery": "publicationsByYearQuery",
    "facetClass": "writers",
    "filterTarget": "author",
    "resultMapper": "mapLineChart",
    "resultMapperConfig": {
      "fillEmptyValues": true
    },
    "createChartData": "createSingleLineChartData",
    "title": "Publications per year",
    "xaxisTitle": "Vuosi",
    "xaxisType": "category",
    "xaxisTickAmount": 30,
    "yaxisTitle": "Publications",
    "seriesTitle": "Publications",
    "stroke": {
      "width": 2
    }
  }
}
```

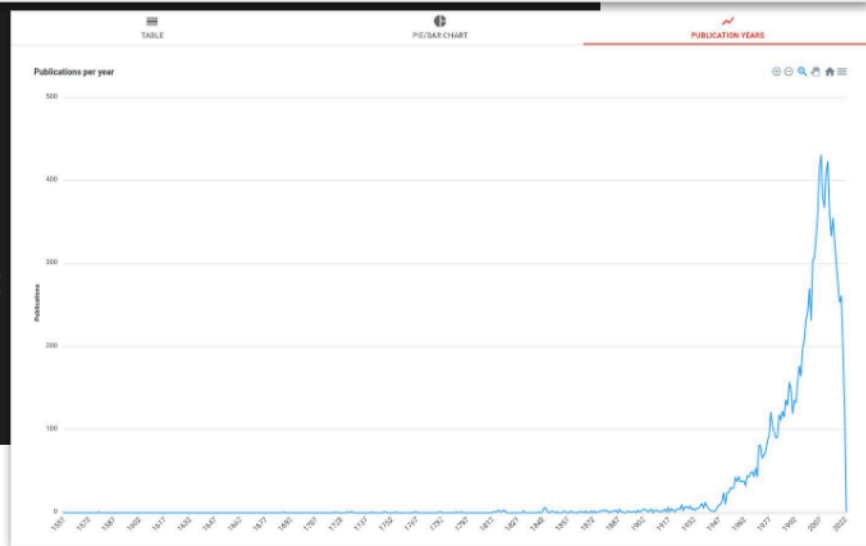
Component used for visualizing

Mapper for mapping the data from SPARQL bindings

Function taking the mapped data and outputting the data with the necessary configuration and format for the visualization component (e.g., adding chart options)

```
SELECT ?category (COUNT (DISTINCT ?book) as
?count) WHERE {
  <FILTER>
  ?book dbo:author ?author .
  ?book dbo:releaseDate ?date .
  BIND(YEAR(xsd:dateTime(?date)) as ?category)
}
GROUP BY ?category
ORDER BY ?category
```

SPARQL query for the visualization data



Other tools and frameworks for creating web applications for LOD

- **Linked Data Reactor (LD-R)**
- **Metaphactory**
- ...

Conclusions

- **After learning the basic process, setting up a new portal using Sampo-UI is easy and can be done in a few hours**
- **Sampo-UI includes many ready-to-use components that only require editing JSON and writing SPARQL to configure**
 - With basic use there is no need to touch the underlying JavaScript code or have deep understand of how it works
 - Examples of different kinds of facets and visualizations come with the framework and can be used as a base
 - *For more advanced use, Sampo-UI can be extended with new components, mappers, chart data creation functions to allow for more customization*

Links

- Tool page: <https://seco.cs.aalto.fi/tools/sampo-ui/>
- GitHub: <https://github.com/SemanticComputing/sampo-ui>
- Demo: <https://sampo-ui.demo.seco.cs.aalto.fi/>
- Tutorial:
 - Video: <https://vimeo.com/817028609>
 - Written tutorial:
<https://seco.cs.aalto.fi/tools/sampo-ui/Sampo-UI-tutorial.pdf>