The PROFMUS Application: Development, Status, and Future Progress

Maria João Albuquerque
Universidade Nova de Lisboa
Faculdade de Ciências Sociais e Humanas, Portugal
mjdalb@fcsh.unl.pt

José Luís Borbinha
Universidade de Lisboa
Instituto Superior Técnico, Portugal
jlb@tecnico.ulisboa.pt

H. Sofia Pinto
Universidade de Lisboa
Instituto Superior Técnico, Portugal
hsofiapinto@gmail.com

Luís Filipe Santos
Universidade de Lisboa
Instituto Superior Técnico, Portugal
luis.96.santos@hotmail.com

Tiago Teixeira
Universidade de Lisboa
Instituto Superior Técnico, Portugal
tiagu.teixeira@hotmail.com

Abstract

PROFMUS is a collaborative project that aims to carry out the research and consolidation of information to support further research about the Portuguese musicians active in the period from 1750 to 1986. The information to be collected must include as many relevant attributes as possible, especially about their academic background, professional careers, and personal details. This project considers a large amount of data from a wide time-period, which means there will be various attributes for each object, which will evolve over time or differ from source to source. There is also an issue wit the lack of uniformity of existing sources in multiple institutions, museums, archives, and databases, each with its own data scheme. Since PROFMUS has a long-term perspective, it does not try to create a uniform and prefixed scheme for the data to consolidate but accepts every different scheme and stores all data in a controlled knowledge base. We describe here an application for that purpose, using MediaWiki and Wikibase for storage, and a back-office specific application to manage and publish the data.

Introduction

The PROFMUS application aims to store information gathered about Portuguese musicians, works, professional careers, etc. The goal is to allow researchers to store, retrieve, and publish data in a flexible way on top of a knowledge base reusing open-source solutions. The target audience of this application will be the researchers affiliated with the research project PROFMUS, who will collect data on Portuguese musicians. It is expected that the volume of data in the database will have to cover a large time period. The data upload workflow is designed to be collaborative, with an administrator to validate the uploads made by the researchers. This administrator will be responsible for ensuring the quality, origin, and validity of the records. All versions of the entities will remain in the database with an indication of the source (project, third-party system, etc.) in order to allow tracing the sources of each piece of data.

There are very few similar Portuguese projects in musicology or in other Digital Humanities domains using this technology. The vast majority use relational databases. We know of two projects that use this technology to support their databases. In Portugal, the TechNet EMPIRE project[1] (Nova University of Lisbon) aims to "look to the agents of the colonial 'dispositif' experts and institutions – throughout time and space in order to understand how they created and shaped technoscience networks across the Portuguese Empire". And in Spain,

1 https://technetempire.fcsh.unl.pt/en/presentation (accessed January 12, 2022).

the Ricardo Viñes + Wikibase Project [2] (University of Lleida) intends to trace the famous Spanish pianist's life using Linked Data.

PROFMUS may in the future aggregate, organize, interconnect, and contextualize authors within the scope of cultural heritage studies to support researchers and the general public alike.

1 Conceptual Solution

The PROFMUS application considers the following main requirements:

a. The application should be simple enough so that less technically skilled users can use it with no major difficulties.
b. Different data structures from diverse sources should all be supported. Through the meetings with the researchers, it was clear that Excel files were heavily used, so the system needed to support this kind of file.
c. Data collectors most of the time use forms to collect data, so the application should have support for some kind of forms that can be shared across users.
d. It should be possible to create subsets of data and share or publish them.
e. Reuse of information is important, so the application should be able to export data (items, properties, forms, etc.) so that it can be used in other applications or scenarios.

Wikibase empowers MediaWiki to store structured data into a data repository and access it [1]. Wikibase consists of two extensions, Wikibase Repository and Wikibase Client, which can be enabled either simultaneously or individually [3]. The Wikibase Repository extension turns a MediaWiki build into a repository of structured data, which can be stored and edited. Wikibase Client creates a client for a structured data repository allowing for data access and visualization. With Wikibase, it is possible to create a knowledge base about any subject and with concepts and properties that are more suitable to the respective domain. The Wikibase data model has entities as the most basic element. An entity can either be an item or a property. Items are the entity that represents anything that needs to be described. Every item has its web page, an identifier, a label (which is usually the name of the item), a description and aliases. Each item can have its own set of statements. Statements are pairs of properties and values. This is how facts about an item are described.

Properties have a structure similar to items. This includes a label, a description, aliases, and even statements. Additionally, properties have a data type associated with them. This means that each property can only accept values of one type. These types can be a string, an integer, coordinates, an object, etc. Statements can also include qualifiers and references. Qualifiers are pairs of properties and values that can add context and additional information to a statement. For example, if a statement says that a musician was married to a specific person, two qualifiers might be the date and place of marriage. The purpose of references is to indicate the source of the information in the statement. A reference can be an URL to an external website, a book reference, or even a Wikibase object.

For our purpose, two new concepts were introduced: views and forms. A view is a sub-collection of the knowledge base that consists of a set of items and properties. Views are created and managed by users and their purpose is to be published. Publishing a view means that the content of that view will be displayed on its web page, where items and properties can be viewed in a more visually pleasing way. Any file imported to the system can originate new items or a new form. The re-utilization of forms reduces the work needed to upload data into the system. These two concepts are managed in the PROFMUS application. This avoids future compatibility problems if, for instance, the MediaWiki deploy is updated to a newer version.

2 Architecture

The application consists of four different components (Figure 1). The Object Controller is a MediaWiki deploy that will manage and support the Object Repository. Additional information, such as users, views, forms, etc., will be stored in the Management Database. These three components will be coordinated by the Application Controller. The Application Controller was built using Django and contains all the logic of the application for both the back-end and front-end. This contains the interface users will mostly interact with. The application will process data from imported files, connect to MediaWiki and Wikibase through the use of bots[2], and interact with the Management Database to manage its content.

The Management Database is an SQLite database connected to the Application Controller and serves as the storage for views, forms, users, usage logs, and to keep track of deleted items. This database is completely independent of both the Object Controller and the Object Repository.

The Object Controller corresponds to a MediaWiki deploy and is the place where most of the data, such as items, properties, and all the relationships between them, are stored. It also provides more granular management of the data, but it also adds more complexity to the operations.

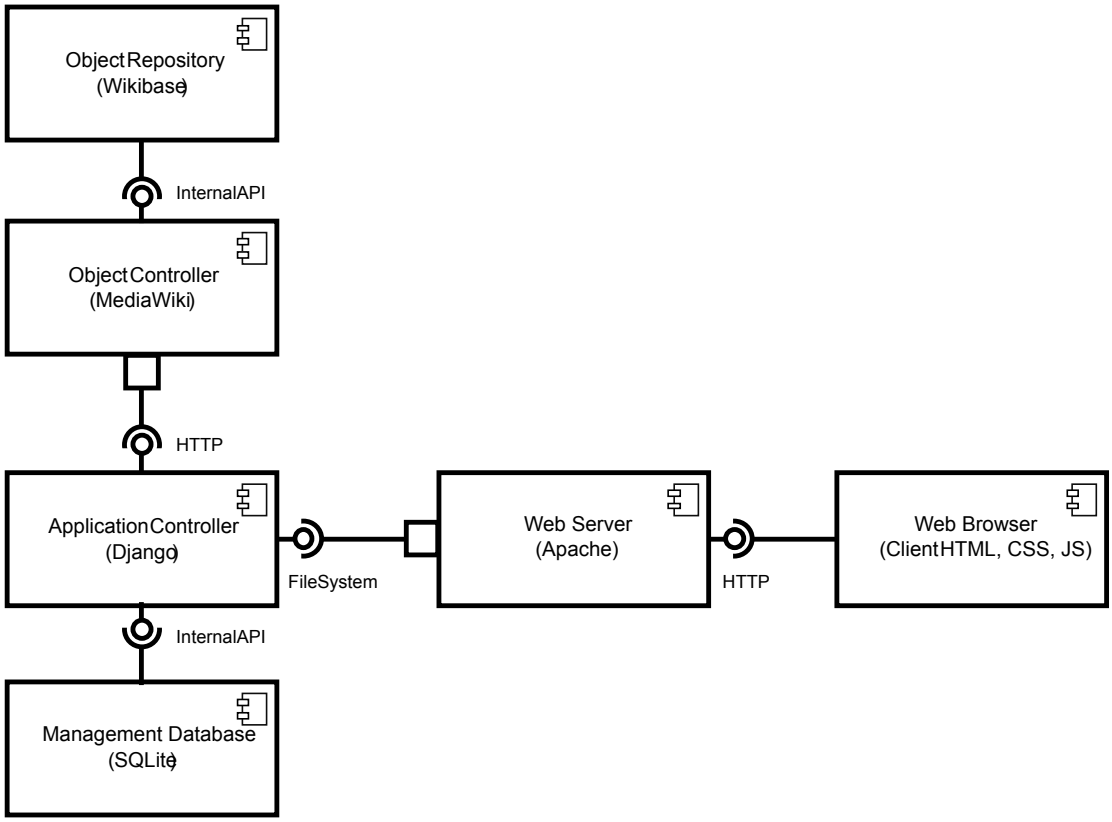


Figure 1: Conceptual architecture of the PROFMUS application [4].

2 A 'bot' – short for robot – is a software program that performs automated, repetitive, pre-defined tasks.

3 Use Cases for Data Management

Given the requirements and the knowledge acquired from the meetings with the researchers, the following set of use cases was created:

a. Create account and login: A user creates an account, logs in, and has access to objects in the system. At this point, only some of the functionalities will be available. Users will have read-only permissions until this is changed by an administrator.
b. Import item set: A user imports a spreadsheet file where each line represents an object and each column the values for a property. This also might create a new normalized form.
c. Export item set: A user selects a set of items to export, and the system creates a file with the objects selected and the properties that hold values about those objects.
d. Create normalized form: To facilitate the import of data, a normalized form is created by importing a file containing a template of the form to be used. This will be saved in the system and can be exported at any time by users with permissions to access it.
e. Create view: A user intends to create a collection of items, called a view, that can be managed and published. So, the user either selects the items or uploads a file with the items that will belong to the view, gives it a name, a description, and defines its privacy. A new view will be created with its own web page.
f. Export view: A user can export the items in a view to an Excel file. This file can then be used to edit the view or generate a new one.
g. Manage item: When items are stored in the knowledge base, they can then be managed by users that have permissions to do so. This includes:
    i. Delete items from the system, by selecting them from the list of items or importing a file with the QIDs[3] of the items to be deleted.
    ii. Restore items from the deleted items list. Wikibase does not allow for items to be truly removed from the knowledge system. So, for example, if an item was deleted by mistake, it can easily be restored later.
    iii. Edit items already imported to the system. The changes made to the system are made using the interface provided by MediaWiki.
h. Manage permissions: To change the permissions of a specific user, an admin can choose the best fitting set of permissions that the user should have.

4 Retrieve Data

The most effective solution for extracting information would be through a query execution system directly in our repository, but the original capabilities and operations available through the API are limited. For operations to which the API cannot correspond, alternative solutions had to be developed [5].

A view is a subset of all the objects in the system, where each object is made up of a set of properties and their values. For example, we can consider as property “Date of Birth” and as its respective value a date that represents the date of birth of the corresponding object. Views can be created and managed by users of the system, plus they can be published meaning that each view will have its own website, which can be shared with anyone.

The creation of views, for publication, is done in another application also developed within the scope of this project, which uses its own SQLite database to store the necessary management data. It is also in this data management application that the ‘connection’ to the view visualization application is made. The Object Controller is created by an implementation of MediaWiki that will manage and support the Object Repository. All kinds of additional information such as the views elements or the aesthetic definitions of each view will be saved in an SQLite database. These elements are coordinated by the Application Controller. This controller is a Django application that connects to MediaWiki via HTTP and to the database via an internal API. The Application Controller is also responsible for the interface shown to the user through an Apache web server.

3 QID is a Unique ID that identifies the item.

The administrator has these options to manage the views:

a. edit names, titles and subtitles;
b. choose the properties to be displayed in the table;
c. edit the display information order;
d. choose the geographical properties to be projected on a map;
e. choose the objects to be projected on the map;
f. choose the color of the geographic marker corresponding to each object;
g. choose the objects to be shown on the timeline.

For the construction of the final visualization, the Application Controller extracts from the database all the settings saved by the administrator and applies them to the base HTML page, which serves as the basis for all views. As a web framework, Django needs a convenient way to generate HTML code dynamically. The most common approach depends on templates. A template contains the static parts of the desired HTML output as well as some special syntax that describes how dynamic content will be inserted. A Django template is a text document or Python string marked using Django template language. Some constructs are recognized and interpreted by the template's mechanism. The main ones are variables and tags. A model is rendered with a context in the form of a Python dictionary, which contains the information that will replace the variables present in the template with the values due. The first step taken by the application to build the interface is to extract an array from the database with the objects' identifiers of the view in question. After that, the application makes an HTTP request to the MediaWiki API to receive a JSON file that contains all the properties and the respective values as well as qualifiers if they exist. This step is repeated for each object.

Conclusion

The PROFMUS application aims to be a platform to import, export, and publish data related to musicians. It was chosen for the project due to its scalability and the fact that successful projects, such as Wikidata, use the same technology. A file type widely used by researchers in this field, Excel, is converted into Wikibase items. This allows for a straightforward experience for the users while still having the flexibility of working with complex data.

There are some current limitations regarding the format of files to be imported, since it only accepts `.xlsx` files at the moment, and also some limitations on data editing. There were also some problems in the import process because the system had some limitations importing numbers. In this sense, after a complex analysis of the functional code responsible for importing the data and after several debugs, it was possible to make this process independent of the type of data imported. Consequently, it was possible to import a first block of the actual data provided. Despite these identified limitations, this system has been subject to an evaluation that has yielded positive results.

Works Cited

[1] Bergamin, Giovanni, and Cristian Bacchi. "New Ways of Creating and Sharing Bibliographic Information: An Experiment of Using the Wikibase Data Model for UNIMARC Data" *JLIS.it* 9, no. 3 (2018): 35–74, https://dialnet.unirioja.es/servlet/articulo?codigo=6794212.

[2] Bernadó Tarragona, Màrius, and Esther Solé Martí. "Ricardo Viñes en la era de los datos abiertos enlazados: un nuevo enfoque de su trayectoria como intérprete" in *Musicología en web. Patrimonio musical y Humanidades Digitales*, eds. María Gembero-Ustárroz and Emilio Ros-Fábregas. Kassel: Reichenberger, 2021, 287–314.

[3] Kleiner, Frank. *A Semantic Wiki-based Platform for IT Service Management*. Karlsruhe: KIT Scientific Publishing, 2015, https://doi.org/10.5445/KSP/1000045291.

[4] Santos, Luís. "Flexible Data Management and Publication with Mediawiki." Instituto Superior Técnico, Lisboa, 2020. MSc dissertation.

[5] Teixeira, Tiago. "Ser Músico em Portugal: Publicação e Visualização de Dados com Solução Mediawiki e Arquitetura MVC." Instituto Superior Técnico, Lisboa, 2020. MSc dissertation.

Application: Development, Status, and Future Progress

Maria João Albuquerque (1), H. Sofia Pinto (2), José Luís Borbinha (2), Luís Filipe Santos (2), Tiago Teixeira (2)
(1) Universidade Nova de Lisboa Faculdade de Ciências Sociais e Humanas, Portugal
(2) Universidade de Lisboa Instituto Superior Técnico, Portugal

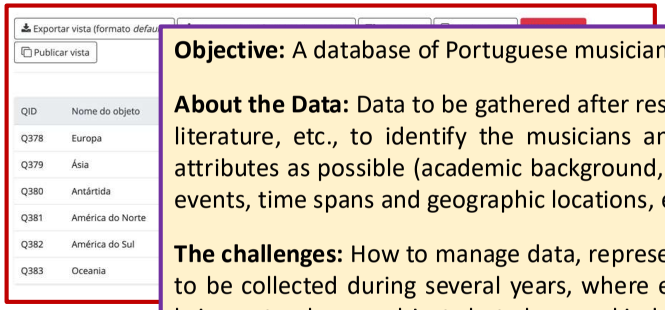
Objective: A database of Portuguese musicians active from 1750 to 1986.

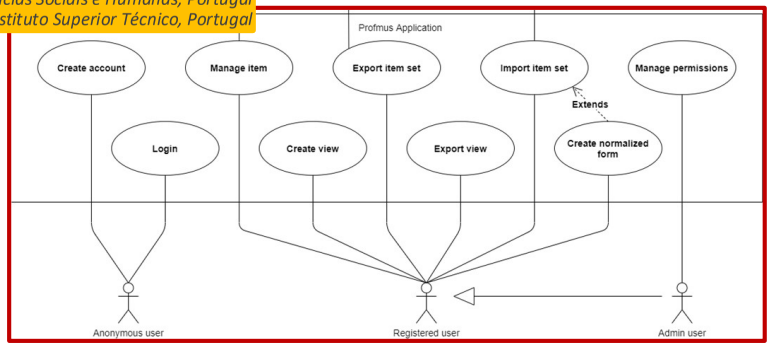
About the Data: Data to be gathered after research in archives, museums, literature, etc., to identify the musicians and collect as many relevant attributes as possible (academic background, professional careers, family, events, time spans and geographic locations, etc.).

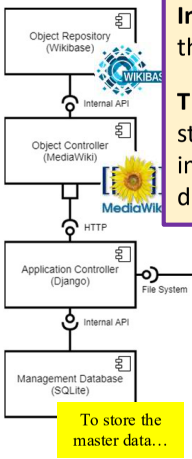
The challenges: How to manage data, representing facts in a wide period, to be collected during several years, where each new data source might bring not only new objects but also new kinds of relevant attributes? How also to do that if long-term technological support might not be assured?

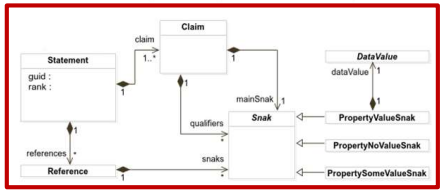
Implications: Hard do design a definitive data schema and stable software that will not result a constraint at some future moment in time…

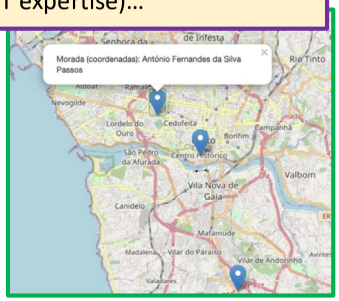
The solution: A flexible application using MediaWiki and Wikibase to store the data as an ontology, and a back-office simple application to import, export and publish the same data according to flexible schemas, defined in spreadsheets (thus, not requiring IT expertise)…











nome	Estado civil	data de nascimento	Morada 2 (residência)
António Fernandes da Silva Passos	Solteiro		Rua Dr. Pedro de Sousa, 578 - Porto
Artur Alves de Almeida	Casado		Rua 1º de Maio, 142 - 1º - Vila Nova de Gaia
Manuel da Silva Carvalho	Casado		Rua Mormugão, 272 - São Mamede de Infesta - Matosinhos
Carlos Alberto Almeida Freitas	Casado		Rua Monte dos Burgos, 1010 A - Porto
Mário Lopes da Costa Ranito	Solteiro		Travessa Dr. Torrinha, 12 - S. Mamede de Infesta - Matosinhos
António Paixão da Silva Pereira	Casado		Rua Coats & Clark, 190 - Vila Nova de Gaia
Samuel Paixão da Silva Pereira	Casado		Rua D. Pedro V, 483 - 2º Esq. - Vila Nova de Gaia
Américo Pereira Macedo	Casado		
José de Magalhães	Casado		Rua Monte da Lapa, 126 - Porto
Carlos Amorim Rodrigues	Casado		Praça do Dr. Pedro Teotónio Pereira, 32 - Porto