Towards an Information Management System for a Research Lab

Jorge Goulart\textsuperscript{a}, Pedro Silva\textsuperscript{a}, Helena Galhardas\textsuperscript{b}, Alberto Vale\textsuperscript{c}, and João D. Pereira\textsuperscript{b}

\textsuperscript{a}Instituto Superior Técnico
Av. Rovisco Pais 1, 1049-001 Lisboa, Portugal
\texttt{jorge.aragao.goulart@tecnico.ulisboa.pt, pedro.m.silva@tecnico.ulisboa.pt}

\textsuperscript{b}INESC-ID and Instituto Superior Técnico
Av. Rovisco Pais 1, 1049-001 Lisboa, Portugal
\texttt{helena.galhardas@tecnico.ulisboa.pt, joao.d.pereira@tecnico.ulisboa.pt}

\textsuperscript{c}Instituto de Plasmas e Fusão Nuclear, Instituto Superior Técnico
Av. Rovisco Pais 1, 1049-001 Lisboa, Portugal
\texttt{avale@ipfn.tecnico.ulisboa.pt}

Abstract. An Information Management System (IMS) is a software platform that enables the management of a vast array of information about products, customers, employees, suppliers, projects, production, assets and finances of an organization. Scientific research labs and education centers also benefit from the use of an IMS. This paper presents the design, implementation and validation of a new IMS for research labs. The system is modular in nature. Each module is independent from the others and designed to be integrated with other external software systems. A set of performance and usability tests were performed to ensure that the system accomplishes its requirements.

Keywords: Information Management System, Human Resources, Production, Assets, IT Support, Research Lab

1 Introduction

An Information Management System (IMS) is a software platform that enables to manage a vast array of information about the products, customers, employees, suppliers, projects, production, assets and finances of an organization. An IMS is typically used by all departments of the organization and by most of its employees and collaborators.

Currently, there are several IMSs available in the market, with slightly different goals and different target audiences\textsuperscript{1}. In the majority of the available solutions, there is a focus on financial, customer relationship, marketing and sales management. Scientific research labs and education centers also benefit from the use of an IMS since they need to manage information about their employees, projects and assets. Nevertheless, there are specific needs for these institutions
that traditional IMSs do not satisfy, such as the management of their scientific production (e.g., publications and scientific projects).

Instituto de Plasmas e Fusão Nuclear (IPFN) is a research laboratory of Instituto Superior Técnico (IST) and is an example of a research lab. The requirements of IPFN in terms of IMS are detailed in [1] and summarized as follows:

- **Human Resource Management**: Member Management, Member Profile Management, Assiduity Management and Mission Management.
- **Asset Management**: Acquisitions Management, Inventory Management.
- **IT Support**: Support Ticket Management, Network Management.

In order to find which existing technology could fulfill IPFN’s requirements, we analyzed the characteristics and functionalities of different IMS available in the market. 1ERP showed to fulfill most part of the requirements, with support for the management of Members, Member Profiles, Assiduity, Missions, Projects, Scientific Production, News, Acquisitions, Inventory and Support Tickets. However, this system is not easily extensible with new functionalities and presents limitations in terms of the devices compatible with the client application. Other systems, like FenixEdu, despite not fulfilling all requirements, are open-source and provide Application Programming Interfaces (API), which allow them to be easily expanded and integrated with other systems.

Although there is a wide variety of IMSs, we concluded that none of the analyzed systems is suitable to be used for IPFN, because none of them satisfies all the requirements.

A possible solution would be to use two or more systems together, taking advantage of their APIs and capability to be extended. However, this approach would require a server with more resources. Moreover, it would lead to significant maintenance and extensibility overhead, and therefore it is also not a suitable solution.

In this paper, we describe the design and implementation of the new IMS for IPFN. The paper is organized as follows. Section 2 summarizes the study carried out about the most relevant IMSs in the market, analyzing their characteristics, and comparing the functionalities offered with the requirements described in Section 1. Section 3 details the solution used to implement the new IMS. Section 4 reports the main results obtained during experimental evaluation. Finally, Section 5 concludes by presenting future work to be developed.

2 http://www.ipfn.tecnico.ulisboa.pt
3 http://tecnico.ulisboa.pt
4 http://www.quidgest.pt/e_qerpPT.asp
5 http://fenixedu.org
2 Related Work


From the analysis that we performed, we conclude that none of the systems fulfills all the requirements for the IPFN’s IMS. ERP has the most required functionality, but it has several limitations, such as the fact that the client application is only available for Windows operating systems. Therefore, IPFN members would not be able to register their assiduity using their smartphone or tablet devices. A possible solution would be to use two or more systems together, taking advantage of their APIs and ability to be extended. The use of various systems would result in less overall performance (and, therefore, require more resources). Moreover, systems would be harder to maintain and extend.

Taking these observations into account, we decided that the best option would be to develop a new IMS. This system should, similarly to other systems, have: (i) a highly modular architecture, with each module implementing only a small fraction of the required functionality, thus allowing the system to be easily maintained and extended with new functionality, and (ii) a web based client application, with a responsive design, in order to allow users to access it from any device, including smartphones and tablets.

3 Information Management System

The new IMS was developed in order to enable its integration with other external systems already existing at IST. Figure 1 (on the left) represents an overview of the new IMS, and how it communicates with external systems, namely:

**CAS Técnico** Central Authentication Service (CAS) allows any user with a TécnicoID to authenticate in any IST system (for example, Fénix and Dot). IPFN members also have a TécnicoID, and thus CAS Técnico should be used to authenticate them in the new IMS.

**Fénix** Using Fénix’s API, it is possible to obtain personal information of the authenticated member. This information is used to synchronize the member’s data in the new IMS with the data maintained by Fénix.

**Dot** Using Dot’s API, it is possible to obtain a list of mission processes related to the authenticated member, as well as detailed information of each of these processes. This information is used to list and generate mission process reports in the new IMS.

**Portal IPFN** The new IMS provides an API to external systems, such as Portal IPFN, which can publish information about news, members and research groups.
The modular architecture also enables to add new modules or modify existing ones (represented on the right of Figure 1) to accomplish new or different requirements. This aspect is particularly important to other research units, as they may develop their own modules, either from scratch or by modifying existing ones, to satisfy their specific needs.

For implementing the system, we used the BennuFramework, as it provides a significant amount of functionality that enables an easy integration with other external systems belonging to IST. Using BennuFramework as a basis enables to organize the various modules of the system in layers, obtaining the structure represented in Figure 2.

Each layer in Figure 2 depends on the layers below it and can call only methods and use classes that belong to itself or to lower layers. Each layer does not have access to the upper or lateral layers. The layers represented in Figure 2 are as follows (starting from bottom to top):

- **Fénix Framework** enables the development of Java applications that require a persistent and transactional domain model.
- **BennuFramework** Java web application development framework.
- **Messaging** BennuFramework module that allows applications developed to use the BennuFramework to send emails.
- **Common Functionalities** Extra module of the new IMS, which implements several utility classes used by other modules, as well as a data structure to support the procedure for searching domain objects in these same modules.
- **Members** All modules depend on the Members module. In addition, the Asiduity Management module also depends on the Project Management module.
- **WebApp** Extra module of the new IMS, whose objective is to integrate all the other modules into a single application.

Each module in the new IMS is divided into three layers. These layers, from bottom to top, are:

- **Domain Layer** Composed by classes generated by the Fénix Framework, each representing an entity in the module’s domain. The Fénix Framework is

https://github.com/FenixEdu/bennu
Fig. 2. Implementation layers of the new IMS.

responsible for providing persistence. Business logic should also be implemented in this layer.

Service Layer Implements all possible actions to be performed on the domain objects. Methods that involve writing data (such as adding or editing data) must be atomic.

Presentation Layer Consists of Controllers, which answer to requests from system users.

In addition, to enable the representation of domain layer objects in the service and presentation layers, we used simple and temporary objects designated as beans. These objects allow the presentation layer to represent domain objects without directly accessing them, thus maintaining the level of abstraction.

A flexible solution for searching and sorting domain objects is an important functionality for IPFN, as many modules require the ability to list and search their domain objects. However, BennuFramework does not support the functionality of searching for domain objects. It only enables to look for a domain object by its ID or, given a domain object, a set of all domain objects that are part of a relation with the former. Therefore, it was necessary to develop a functionality capable of filtering only the objects that correspond to the terms indicated by the user during a search, and then sorting them according to the given indications.

4 Experimental Evaluation

During the final stages of development of the system, before putting it into production, we performed a set of tests to check that the system satisfied IPFN’s needs. We validated the new IMS according to three criteria: (i) performance, (ii) Graphical User Interface (GUI), and (iii) functionality. This section reports the methodology and results of the tests.

To validate the performance of the system under typical conditions in a production environment, we performed several performance tests with JMeter. The performance tests were divided in 9 scenarios: one with a base value of 50 users performing 50 actions each, four with variation in the number of users (each user performing 50 actions), and the remaining four with variation in the number
of actions performed (always using 50 users). In each scenario, a certain number of users performed a set of actions following a certain order. Each scenario was tested twice, one where there was no time elapsed between each action, and another where there was a delay of 1 second between actions.

The 4 scenarios with variation in the number of users, plus the base scenario, aimed at studying the behavior of the system when the number of users increased (50 actions were performed by 50, 100, 200, 400 and 800 users). The 4 scenarios with variation in the number of actions, plus the base scenario, aimed at studying the behavior of the system when the number of performed actions increased (50 users performed 50, 100, 200, 400 and 800 actions). During the execution of these tests, we measured three different metrics: (i) error frequency, (ii) throughput, and (iii) average response time.

When the number of users increased, the number of errors grows exponentially. Adding a delay between actions did not significantly impact the error frequency. With an increase in the number of actions, we verified that the error frequency remained constant. The growth in error frequency is, thus, related to the number of sessions being managed by the Bennu Framework, and not related to the overload due to processing each action.

The graphics represented in Figure 3 show the results obtained regarding the variation of throughput of the system with the number of users and the number of actions, respectively. In the case where the number of users varied, the throughput decreases until we reach 400 users, and then starts to increase again. A similar situation is observed when there is 1 second of delay between actions, with the exception of the case with 50 users. This low throughput value, when we have a low number of users and a delay between actions, is due to the fact that there is a small number of requests being performed every second. When the number of actions varied, the throughput decreases until we reach 200 actions per user, starting to increase after this point. However, when there is a delay between actions, the throughput increases logarithmically with the number of actions. With the increase of the number of actions performed every second, the decrease of the throughput indicates that the time needed to process each request increases, leading to less requests being answered each second. However,
the processing time of each request does not increase fast enough to continuously decrease the throughput with the increased number of users or actions per user, thus allowing the throughput to increase with large numbers of users or actions.

The graphics of Figure 4 show the results obtained regarding the variation of the average response time of the system with the number of users and the number of actions, respectively.

![Fig. 4. Variation of the average response time with the number of users and with the number of actions, respectively.](image)

By analyzing Figure 4, we conclude that the average response time increases linearly with the number of users. Introducing a delay between actions does not significantly impact this value. When the number of actions varied, the average response time increased until it reached 200 actions per user, after which it starts to decrease. However, when there is 1 second of delay between actions, the average response time is constant. Knowing that the total number of actions remains constant, we conclude that the increase of the average response time is due to a large number of sessions being managed by the Bennu Framework, and is not related to the time required for processing each action.

Regarding the validation of GUI, we performed tests with the future users of the system (i.e., IPFN members). The goal of these tests was to evaluate the usability of the GUI, and to assess the user satisfaction when using the new system.

User satisfaction tests were performed with 13 members of IPFN. For each user, we requested to perform 22 tasks in the system and collected four types of data: (i) whether the user was able to finish the task, (ii) time taken to finish the task, (iii) number of clicks performed, and (iv) number of mistakes performed. Based on the results of these tests, we concluded that, overall, the tasks were intuitive to the user and easy to perform. Some users left suggestions to improve some functionalities whose tasks were more prone to errors and waste of time.

After finishing each test, the user was invited to fill in a satisfaction questionnaire. This questionnaire was divided in three sections: (i) personal information and preferences, (ii) System Usability Scale (SUS) [7], and (iii) additional comments. The answers to the questionnaire had an average score of 85.4 out of 100,
with a standard deviation of 9.1. This score is high, which indicates that the users felt that system has a good usability, and they had a positive experience.

5 Conclusions

The new IMS was developed to satisfy all the needs of IPFN. These needs are shared by other research laboratories that can also benefit from using this system. We analyzed the existing IMSs and concluded that no system fulfilled all the requirements of IPFN.

The IMS developed has a modular architecture, enabling easy add-ons and modifications to be performed in each module without affecting the other ones. It is part of the IST ecosystem, using multiple resources from the systems CAS Técnico, Fénix and Dot, and providing additional information to external systems such as Portal IPFN with a JSON API. The system uses the BennuFramework, which also has a modular architecture and offers multiple functionalities.

We performed an experimental evaluation of the new IMS. The evaluation assessed: graphical user interface, performance and functionality. Based on the results obtained, we concluded that the system satisfies the requirements.

The new IMS was placed in a production environment on March 13, 2017, and it has been used by all members of IPFN since then. Support to the system was maintained in order to correct bugs that were found and to continue the development of new functionalities.

Acknowledgements. This work was supported by Fundação para a Ciência e a Tecnologia with references UID/CEC/50021/2013 and UID/FIS/50010/2013.

References