A Software Ecosystem platform for the development of Recommender Systems

André Abdalla
Universidade Federal de Juiz de Fora
Victor Ströele (victor.stroele@ice.ufjf.br)
Universidade Federal de Juiz de Fora https://orcid.org/0000-0001-6296-8605
Fernanda Campos
Universidade Federal de Juiz de Fora
Regina Braga
Universidade Federal de Juiz de Fora
José Maria N. David
Universidade Federal de Juiz de Fora

Research

Keywords: Recommender Systems, Software Ecosystem, Platform, Solutions reuse and sharing

Posted Date: June 10th, 2020

DOI: https://doi.org/10.21203/rs.3.rs-34335/v1

A Software Ecosystem platform for the development of Recommender Systems

André Abdalla, Victor Ströele[1], Fernanda Campos,
Regina Braga, José Maria N. David
Postgraduate Program in Computer Science
Federal University of Juiz de Fora
Juiz de Fora, Brazil
{andre.abdalla, victor.stroele}@ice.ufjf.br; {fernanda.campos, regina.braga, jose.david}@ufjf.edu.br

ABSTRACT: *Many application domains need to recommend resources for users. The development of solutions focused on the reuse of Recommender System components creates an interesting scenario from a Software Ecosystem perspective. Besides the interaction between actors and technology, a Software Ecosystem for the development of Recommender Systems should allow for integration with systems and platforms that support other ecosystems. The problem addressed by this study is the integration of methods, techniques, and approaches of existing Recommender Systems in a systematic way, facilitating the implementation of new solutions, their reuse and sharing, and the collaboration among the actors involved. This study, therefore, aims to propose R.ECOS, a Software Ecosystem platform to support the development and management of Recommender Systems, allowing for integration between multiple applications and other Software Ecosystems. The evaluation was conducted in two stages. First, two feasibility studies were carried out to validate both technology and architecture. Later, two case studies were conducted in real-world contexts. The results point to the viability of the proposal.*

Keywords*: Recommender Systems, Software Ecosystem, Platform, Solutions reuse and sharing.*

1. INTRODUCTION

With the increasing use of the internet, a considerable volume of online information is spread over 1.3 billion websites [1], which contain, in general, information that is irrelevant to users. In an attempt to reduce these shortcomings, some of these environments have software applications that filter the data and improve the user's experience while browsing. These specific software applications are known as Recommender Systems (RS).

Designing a recommender system is not a trivial task, as it requires specialized and often interdisciplinary knowledge. So, some barriers have emerged, as the difficulty in developing and reusing specialist RS solutions, which often leads to rework [2].

There are different strategies for intensive software development, which requires the involvement of different actors with specific expertise. One of these approaches is the Software Ecosystem (SECO) perspective [3], which allows various institutions and organizations to contribute with solutions and innovations for the development of specific systems, such as Recommender Systems. These systems are increasingly abundant in the diversity of content, services, experiences, and ability to find personalized resources aligned with the user's real needs in a specific domain.

Given the variety of recommender systems solutions [2, 4] and the demand for presenting relevant and personalized information to users [5], it is necessary to think about *free* and *open-source* solutions that are reusable and easily adapted to different application domains. It is an interesting scenario for the adoption of solutions from open-source software ecosystems [6] with the reuse and integration of recommendation features.

Without the possibility of defining architecture standards, and the benefits related to techniques and recommendation approaches of reusing and sharing [7], the granularity of recommendation solutions demonstrate the need for establishing a common technological platform to create solutions and take advantage of

[1]**Corresponding author: Victor Ströele**
Tel.: +55 32 2102-3311 or +55 32 98881-9990
Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas – ICE
Departamento de Ciência da Computação – DCC
Rua José Lourenço Kelmer, s/n – Campus Universitário
Bairro São Pedro – Juiz de Fora – MG – Brazil
CEP: 36036-900
E-mail address:victor.stroele@ice.ufjf.br

the existing ones [2]. This scenario encourages research in these two main areas: *Recommender Systems* and *Software Ecosystem*. The present study addresses the integration of various methods, techniques, and approaches of Recommender Systems aiming to facilitate the development of new solutions, to promote reusing and sharing of such solutions, and to enhance collaboration among the actors involved.

Based on the motivation and the stated problem of this study, the research question that arises is the following: *How do the proposal of a recommender system approach, from a software ecosystem perspective, favor the development, reuse, and sharing of recommendation solutions, allowing for integration with other systems and platforms?*

The general objective of this paper is to propose R.ECOS (***R****ecommender systems* ***ECOS****ystem*), an online, free, and open-source platform to support a software ecosystem in the development and management of Recommender Systems. Specific objectives include: (i) to define the R.ECOS platform, with its technological and social components, (ii) to develop and make the R.ECOS platform available, allowing for the development of Recommender Systems, and the sharing and reuse of their solutions, and (iii) to present the results of an evaluation of R.ECOS in a real scenario.

The study followed four main steps: (i) reviewing the literature; (ii) defining an architecture for a Software Ecosystem platform for Recommender Systems; (iii) implementing and making the platform available online; and (iv) evaluating the proposal through two case studies.

The literature review included a tertiary study to detect Recommender Systems solutions and then find secondary studies that help reveal the state of the art.

The architecture comprises of techniques, methods, and approaches with distinct recommender systems functionalities. An exploratory study was carried out to identify and validate the technologies and actors required to define the technological core.

A platform was built with the components identified for the proposal evaluation. At this step, the technologies for the implementation of the system were defined, considering open-source and free solutions. An *Application Programming Interface* (API) was also developed to specify the communication pattern between the platform components and measure the users' access level according to their actor roles. Two feasibility studies helped to decide on the technologies that should be used. Later, two case studies were carried out on real-world contexts to evaluate the proposal. This study is part of the BROAD Project [8–10].

This paper is organized as follows: Section 2 discusses the main concepts; Section 3 presents related work; Section 4 discusses recommender systems from a software ecosystem perspective, identifying the necessary technological and social components, and designs the R.ECOS architecture; Section 5 presents the evaluation; final remarks are made in Section 6.

2. BACKGROUND

GOLDBERG *et al*. proposed one of the first systems with recommendation's characteristics and defined the term "collaborative filtering," referring to an experimental email system designed to prevent users from receiving large amounts of irrelevant messages [11]. Other proposals have emerged since then [12, 13], and, later, RESNICK and VARIAN proposed the term, *Recommendation System (RS)*, justifying that collaboration in these systems is not always explicit [14]. Based on this initial definition, many RS-related surveys have been carried out, and several definitions have been attempted. The present study used the following one: *Recommender Systems are systems that present a set of resources (items, videos, articles, services, among others) to users, considering their interests* [15].

Users are an essential part of recommender systems because, based on their profile and context, it is possible to recommend resources as closely as possible to their interests [5]. The profiles can be extracted in two ways: explicitly or implicitly [9]. Explicit extraction occurs when users fill in their information through forms, surveys, and even evaluations presented to them, which helps to define their initial profile. Implicit extraction occurs when the data is obtained without any explicit user action; i.e., it is usually drawn on the user's behavior in an environment. Systems typically use both extraction approaches.

Considering the context of users or groups of users, any information that can be obtained should be considered by the system (historical data, behavior when using the system, sensors, among others). In this paper, context is categorized as internal (user using the system, user skills, previous knowledge, and preferences) and external (temporal information, location, or even the physical environment). The categorization most frequently used in the literature divides the filtering types into (i) content-based, (ii) collaborative, (iii) demographic, (iv) social, and (iv) hybrid, when two or more filtering approaches are used together in the same recommender system [16] [17, 18],[19], [20].

The ways in which resources can be presented to users are relevant in recommender systems, i.e., the repositories from where the resources are obtained and the interface in which they are recommended. Among the most common ways mentioned in the literature is "top-N," where the N most indicated resources for a user or group of users are chosen and sorted in a list to be recommended. This list is generated by considering the

resources already consumed by other similar users, and by ordering the candidate resources to be presented in decreasing order of relevance [21]. An optional component can evaluate these recommendations. The evaluation may be implicit when the system evaluates whether users liked or disliked the recommended resource through their actions (consuming the resources or not), or explicit, when the users themselves evaluate the recommended resources, sending their feedback.

In order to propose a Software Ecosystem platform that helps in Recommender Systems development, it is necessary to understand how monolithic systems have evolved to the present days. Organizations will adopt the SECO perspective mainly when: (i) their necessity for developing functionalities is greater than their capacity to produce them; (ii) they believe that the support of external actors is an effective mechanism to assist in the customization of available solutions; (iii) attract new users when promoting the platform; (iv) accelerate the production process through shared innovation; (v) reduce the platform cost by sharing solutions with partners [22]. According to Manikas [23], it is not possible to develop a software ecosystem architecture without considering social aspects, along with management (technological) processes and business rules. There must be interaction among actors, as well as between actors and the platform that supports the ecosystem.

One way of analyzing the SECO perspective is considering the three dimensions that support it, involving software reuse, architecture, cooperative work, social networks, socio-technical aspects, software quality, and resource-saving [24]. The "technical dimension" focuses on the platform that supports the ecosystem, the technology used, and the required organization and infrastructure. The "transactional dimension" works on the knowledge flow, its artifacts, resources, and information. The "social dimension" focuses on the actors involved with the software ecosystem and their possible relationships.

In a general way, we can define the actors' functions as *keystones* (platform owners), *dominant players* (key players in the development of the platform), *complementors* (assisting in developing the components on an existing platform), and *integrators* (aiding the integration of elements of different actors) [23]. Besides these essential positions, it is possible to identify other roles that can be performed according to the domain. Based on the actors' role defined in the first version of this proposal [25], the social dimension components of the proposed SECO were categorized as *keystones*, *consumers,* and *contributors*.

3. RELATED WORK

A tertiary study was conducted to identify related papers addressing the following research topics: integration of methods, techniques and recommender systems approaches; use of free or open-source technologies and available solutions to accelerate and reduce the cost of implementing new recommender system approaches; papers discussing the reuse and sharing of RS solutions. The purpose of the tertiary study was to identify published secondary studies and its related research topics [26].

Three surveys addressing, in some way, the research topics of this work were detected [2, 4, 5]. In [5], news recommender systems are discussed. According to the authors, there are a variety of approaches adopted for the design, construction, and evaluation of news recommendation systems. In the end, they present an overview of the surveyed news recommender systems papers, making it possible to identify some frameworks focused on RS solutions sharing.

An overview of existing applications of recommendation technologies in the IoT context is presented in [4]. Although the authors focus on IoT, they surveyed recommendation algorithms, where it was possible to find libraries proposed to facilitate the design of recommender systems.

There are RS solutions that are not reused, and the reasons for that are presented in [2]. The authors claim that an open-source recommendation framework containing the most promising approaches could help to transfer research results into practice. This framework would also help new researchers in the field to access various baselines with which to compare their approaches. Finally, they also highlight some recommender system frameworks.

As no more recent tertiary studies were found, we also conducted an ad-hoc search to find other solutions in Recommendation Systems focused on the research topics of this work. A summary of the articles and a discussion are presented below.

MyMediaLite [27] is a Recommender System library developed in C#, which runs on a .NET platform, whose target audience are industries and researchers. In this study, two scenarios were discussed in the collaborative filtering: rating prediction from explicit user evaluations using a scale of 1 to 5 stars, and even **item prediction** from implicit user evaluations looking for clicks or other user actions in the environment. The library has extensive documentation and tutorials for carrying out typical activities, such as inserting recommendations into a program, implementing a new recommender system, or using available devices in MyMediaLite. The authors use standard practices in free and open-source projects by making all the code available in public repository databases. This library is limited to only two collaborative filtering scenarios, which limits its importance as regards recommendation solutions. Although it has a considerable amount of documentation to aid

in its use, and its code is open and public in a repository, the library still depends on being downloaded and installed in an environment with specific requirements, and this can become a complicated task for users.

Reclab [28] is a system developed to allow the creation and testing of recommendation algorithms for e-commerce sites. The main component of the system is the *Reclab Core*, where the interfaces and APIs for interaction in the Reclab environment are defined. It also provides simple implementations that help developers in designing, testing, and debugging quickly and efficiently the algorithms without having to configure an environment on their own. The Reclab system has become a commercial, closed-source, industry-focused tool. Like other proposals, this system is designed for e-commerce domains. Although the focus is to ensure that building a Recommender System be quick and easy, the system is not used in projects of the users themselves and with recommendations in real-time. First, users should test their algorithm in a testing environment, then submit it to the service provided by the developers. If approved by the system creators, the algorithm is used in virtual stores chosen by the creators. We believe that a platform that would allow users to build their algorithms and readily use them in their virtual stores would be more attractive to the Recommender Systems community.

PEN recsys [29] is a framework for online recommender system evaluation, following the Software-as-a-Service (SaaS) paradigm and implemented using Java EE. Some recommendation algorithms are already available in it. To develop a new algorithm, the creators of the framework must implement a new *getRecommendations()* method. The framework has a control panel that allows the user to configure the general behavior of the environment, activate or deactivate the desired algorithms, among other parameters. There is also a page with metrics on algorithm performance, such as success@k, MAP, and average clicks per visit. Although it is possible for the user to configure the Recommender System, PEN recsys does not specify the roles that users can perform during the RS planning, development, and use. Also, external users can't include new methods in the framework – the framework is specific to news recommendation, and no platform promotes user interaction, which is essential as regards reuse-based solutions.

Recalot [30] is a RESTful web service framework for resource recommendation. It is reusable, generic, and designed to aid in the recommendation system algorithms analysis. This framework provides an API that can be used by applications from multiple domains. Besides, as evaluation tools, a recommendation algorithm library is available. The architecture follows the SaaS paradigm, allowing the framework to be executed in a centralized location. In this framework, an environment for interaction between actors is not foreseen, nor is the availability of solutions developed by third parties. Different roles that actors can play during the life cycle of an RS have not been defined either.

MMRecommender [31] is an open architecture that contains the necessary components for recommender systems building. It is composed of 4 essential steps: (i) Extraction, (ii) Filtering, (iii) Method, and (iv) Recommendation – besides the Enrichment sub-step. The objective was not to propose new models or techniques, but rather an architecture to assist in the creation of the recommender system, based on different strategies, constraints, and domains. The authors believe that the proposal has several benefits, such as the flexibility requirements of different domains; the adoption of models that guarantee the interoperability of components; the availability of models, techniques, and algorithms that allow access to solutions already tested in similar systems; the contribution to the development of high-quality systems for its users. The open architecture was used as the basis for the kernel the platform proposed in this paper. However, MMRecommender did not provide a free and open-source environment for the development of recommender systems, not even a framework from which it would be possible to develop a new solution.

OpenRec [32] is an open and modular Python framework that supports adaptive and extensible recommender system research. Each system is modeled as a graph consisting of a structured set of reusable modules connected through integrated interfaces. For the authors, the philosophy behind the framework is to transform the construction of a complex system into several small ones, making them able to receive new functionalities. OpenRec was developed with a focus on demonstrating the advantages of modularization and reusability in this research area. Another positive point is the possibility of the user to easily exchange modules that make up the system and perform tests, verifying the performance after the change. However, it does not have a platform where a recommender system can be created and used in real systems, so this forces users to download the framework and configure the environment where it will be installed.

3.1. DISCUSSION

The previous proposals present different ways of assisting in recommender systems development, from libraries and frameworks to architecture models and the definition of necessary recommender system components. The main objective of this paper is to define a technological platform to support a software ecosystem for recommender systems. This proposal describes the modules for building recommender systems, the technical components that help in the implementation and consumption of this system, and the roles that users can play in this platform. The following features are considered relevant to the current study:

C1 - Existence of an online technological platform
C2 - Web Services Based Architecture
C3 - Support for different actors' roles
C4 - Open-source
C5 - Free use
C6 - Existence of official documentation
C7 - Metrics dashboard
C8 - Components for interaction between users
C9 - Recommendation Services marketplace

Table 1 presents a comparative study with the main characteristics of the related work addressed and the R.ECOS proposal.

Table 1 - Comparing the related works

	MyMedia Lite	Reclab	PEN recsys	Recalot	MM Recommender	Open REC	R.ECOS
C1	No	Yes	Yes	Yes	Not applicable	Yes	Yes
C2	No	No	No	Yes	No	Yes	Yes
C3	No	No	No	No	Yes	No	Yes
C4	Yes	No	No	Yes	Not applicable	Yes	Yes
C5	Yes	No	Yes	Yes	Yes	Yes	Yes
C6	Yes	Yes	No	Yes	No	Yes	Yes
C7	Yes	Yes	Yes	No	Not applicable	No	Yes
C8	No	Yes	No	No	Not applicable	No	Yes
C9	No	No	Yes	No	No	No	Yes

Based on the exposure by [2] and the purpose of this work in promoting the reuse and sharing of solutions in recommendation systems, R.ECOS was defined considering some elements of previous studies. Thus, in the present paper, we are using the MMRecommender architecture [31] as the basis for determining the kernel of our architecture, the RS modularization presented in OpenREC [32] and Recalot [30], panels with indicators of the metrics most often used concerning RS [27–29], the use of API for communication with the library or framework components [28, 30], and the definition of Recommendation Services based on the RESTful architecture [30].

4. R.ECOS PLATFORM

This section presents an approach to an open-source ecosystem platform of Recommender Systems (RS), being able to integrate different methods, techniques, and approaches of a recommender system in a systematic way, facilitating the reuse and development of systems with recommendation mechanisms.

4.1. RECOMMENDER SYSTEM COMPONENTS

The authors in [31], through a systematic mapping, identified the main recommender system components in the literature. They defined the necessary characteristics to compose an RS and the four main steps thereof: *profile and context extraction*, *filtering*, *model* and *recommendation*, as well as the optional *enrichment* step to aggregate information to user profile/context and assist in the filtering steps. The kernel of the R.ECOS platform was based on these components (Figure 1). To enable the reuse and sharing of solutions, R.ECOS is based on web services. These services can integrate heterogeneous applications, allowing business resources to be available to customers, suppliers, and partners over the internet.

In the R.ECOS platform architecture, recommendation services are currently categorized as:

i) Profile and context extraction services are those used to define the user's or group of users' profile and context, implicitly or explicitly identified. When analyzing user or group, the context itself can be defined as internal (social, tasks, skills, history, preferences, and goals) or external (geolocation, environment and time);

ii) Filtering services define how the resources to be recommended will be selected. They are categorized between content-based (based on choices made by users), collaborative (based on the preferences of similar users), demographic (linking users to groups considering personal characteristics), social (based on social network recommendation), or hybrid (combining two or more different filter);

iii) Enrichment services are responsible for improving the extracted profile and context through information from social networks, linked data, and ontologies;

iv) Model services apply the filtering strategy to relate users with the most appropriate resources and are categorized as *model-based* (creates a model for profile definition and the user or group preferences), *memory-based*, or hybrid;

v) Recommendation services are responsible for presenting the resources to individuals or groups of individuals through the features defined in previous steps. The resources can be available in internal or external repositories.

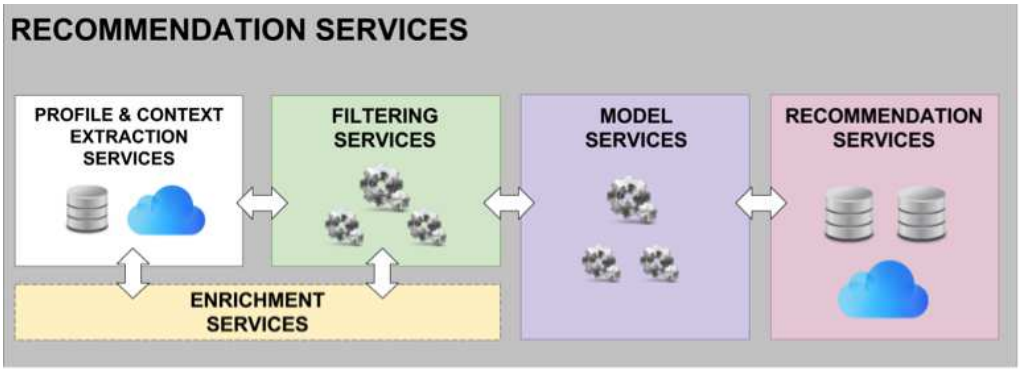

Figure 1 - R.ECOS services

After defining the R.ECOS kernel, a literature review was carried out to identify technologies, actors, and their relationships in recommender systems by analyzing papers published between 2009 and 2019. In addition to this review, components presented in other software ecosystem proposals [23, 33–35] were also considered. The relevance of the components was evaluated by experts in an *exploratory study* (details of this exploratory study are available in Google Drive[2]). Using the categorization defined by [23], the technological and social components are described below.

4.2. SOCIAL COMPONENTS

Given the actors identified in the literature review and validated in the exploratory study, several roles can be discovered, which were primarily categorized as *keystones*, *providers* or *consumers*, adapted from [25, 35]. Figure 2 shows where these actors interact with the platform and its corresponding services.

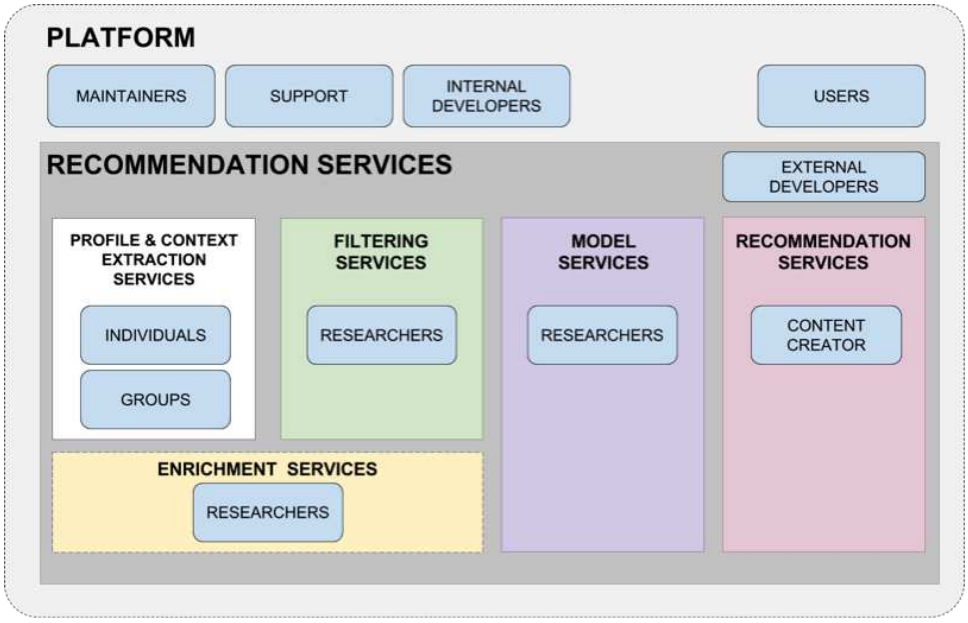

Figure 2 - Social components and their interactions in R.ECOS.

Keystones represent the main category of actors responsible for defining, creating, and managing the platform. Maintainers, developers, internal developers, and a support team are included in this category. **Providers** represent the actors' category that adds value to the platform, with contributions that are crucial to the ecosystem. Some specific actors in this category are external developers, content developers, researchers, and support people. Finally, **consumers** are actors who use software ecosystem solutions, i.e., they are the *end-users* of the platform or the individuals who will receive personalized resources from the system. Table 2 summarizes the social components identified, their function, and some examples of each role.

[2] https://drive.google.com/file/d/1FRtoZCchHuh-We-DCzlhS9GHdJi_5LEG/view?usp=sharing

Table 2 - Social Components, Functions, and Examples

SOCIAL COMPONENT	FUNCTION	EXAMPLE
Maintainers	Create and maintain the technological platform that supports the software ecosystem.	Research Centers
Promoters	Encourage project development.	Support agencies
Internal Developers	Implement the technological components of the platform that supports the software ecosystem, as well as recommendation services.	Developers, database administrators
Master Support	Assist other actors in using resources of the platform.	*Help desk,* attendants
External Developers	Implement recommendation services to be available to the software ecosystem community. They are responsible for developing only RS solutions, while Internal Developers can also develop solutions for the platform itself.	Programmer, Developers
Content Managers	Develop resources such as videos, texts, audios, games, or any other media that can be recommended.	Youtubers, paper authors, news reporter, game developers
Researchers	Propose new solutions in recommender systems or software ecosystem.	Researchers, graduate students
Support	Assist users with the adapter layer configuration in the interface where resources are presented to individuals or groups, as well as any other functionality in the process of using the platform.	System administrator
Users	Use the technological platform and build the recommender system according to the available recommendation services.	Teacher in a Virtual Learning Environment (VLE), owner of a news website
Individuals or Groups	Receive recommended resources.	Students in a VLE, navigators on a news website

4.3. TECHNOLOGICAL COMPONENTS

In addition to the social components, it is necessary to define the technological ones that, together with the business components, are the main elements of a software ecosystem. Figure 3 shows the technological components according to their functionality:

i) The **platform** is the supporting technological infrastructure for the development of Recommender Systems. It is through the platform that an actor connects to other actors and the technological components;

ii) The **software** can be used by the actor for support and use of the platform, and the development of technological components. Among the most relevant platform software are Marketplace, scoreboard (indicator panels), discussion forum, and helpdesk systems;

iii) **Servers** of the technological platform are the infrastructure where the deployed code and databases are stored. In order to use open-source technology, the servers are based on free tools;

iv) **API** is a set of specifications used to enable interoperability and integration of internal and external services. There are also data security rules;

v) **Digital resources** are static or dynamic objects recommended to the appropriate actors;

vi) **Social networks** are structures that connect people, communities, and organizations through many types of relationships. Through a social network, it is possible to extract or enrich the profile/context of individual or group of individuals who are candidates for receiving resource recommendation**;**

vi) Ontologies are models that represent concepts and their relationships in a specific domain and can be used to infer new knowledge. When using an ontology to represent the user's interests and features, it can also help define an *individual's* profile and context;

vii) **Linked data** is a set of structured and organized data so that it can be interlinked and become more useful through semantic queries;

viii) **Digital resource repositories** are the structures that store resources or information about them, which are recommended through the system. They are also used to store information about user's recommendation evaluations and even their behavior in the system;

ix) The **documentation** for the technological platform usage contains information that guides the users. It is available in HTML pages, discussion forums, and wikis. Table 3 presents the technological components identified in this study, their functions, and some examples.

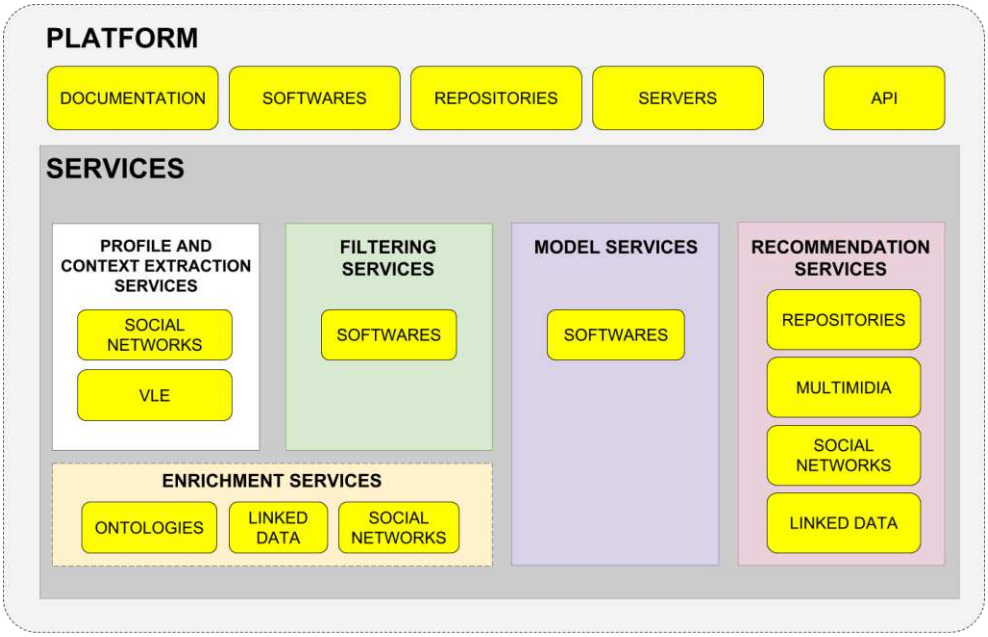

Figure 3 - Technological components in R.ECOS

Table 3 - Technological components and examples

TECHNOLOGICAL COMPONENTS	**FUNCTION**	**EXAMPLES**
Platform	The technological infrastructure of the system.	Android, .Net, BROAD-ECOS, R.ECOS
Software	Systems for actors' support and development of technological components.	Windows, Moodle, WordPress, Android, *Marketplace*, scoreboard
Platform Servers	Storage systems for the codes and databases implemented.	Apache, Postgres, MySQL
API	Set of specifications for interoperability and integration of services.	MeaningCloud, YouTube API, R.ECOS-API
Digital Resources	Digital objects recommended to individuals or groups.	Video, scientific paper, websites
Social Network	Structures that connect people, communities, or organizations across a variety of relationships.	Facebook, ResearchGate, LinkedIn
Ontology	Models that represent concepts and their relationships in a specific domain.	FOAF Ontology
Linked Data	Set of structured and organized data to facilitate understanding their concepts and definition.	DBPedia
Digital Resources Repositories	Structures that store resources or their information, which are recommended by a recommender system.	YouTube, DailyMotion, ACM Digital Library
Documentation	Documents to guide users on the correct use of the technological platform.	Blog, Wiki, forums, tutorial

Several definitions in the literature specify that there must be relationships between the software ecosystem's components, both between actors mutually and between actors and technologies. Such relationships are discussed in the proposal evaluation.

4.4. R.ECOS ARCHITECTURE

The architecture of a Software Ecosystem platform is defined as the structure of a SECO concerning its elements, the properties of such elements, and the relationship between them. Software ecosystem elements are systems, the components of those systems, actors, and relationships between actors and the platform [36].

The recommender system domain is a scenario with fragmented and specific solutions, where several services are developed by different organizations and are tightly integrated into a unified solution. In order to improve this scenario, the current study proposes the R.ECOS reuse-based platform for resources recommendation from a software ecosystem perspective, defining its structure, rules, technologies, and possible actors' roles.

Figure 4 shows the R.ECOS architecture, with its kernel, configuration layers, adapter, and interface, as well as its technological and social components. The **Services layer** is the kernel of the R.ECOS architecture. Actors with internal or external developer roles interact through the implementation of services. From the composition of the available services, it is possible to build Recommender Systems. As previously defined, the categories of the services available are profile and context extraction, profile and context enrichment, filtering, modeling, and recommendation.

The **Configuration layer** is in charge of actors with maintainer roles. In this layer, the *R.ECOS documentation* (with the specifications of its integrations, functions, parameters) and *wikis and discussion forums* are available, facilitating the platform usage and encouraging the social dimension. In this layer, the servers (hosting the technological components and services) were configured. Actors with internal developer roles also interact in this layer, as they are responsible for the implementation of services to make the platform more user-friendly.

The **Adapter layer** was defined to enable the interaction between R.ECOS solutions and environments where the recommendations will be presented. The Adapter layer of R.ECOS is defined with similar characteristics to those proposed by [33]. This layer is directly adapted to user environments where resources are recommended, such as VLEs, social networks, websites, or any other environments where users can be connected. The adapter is presented as an intervention in the environment source code, a plug-in, or an extension.

Finally, the **Interface layer** represents the place where individuals and groups receive the recommended resources. From these actors' point of view, no knowledge about the use of a platform is required to implement these recommendations. Examples of this layer are websites, VLEs, and even smartphone applications.

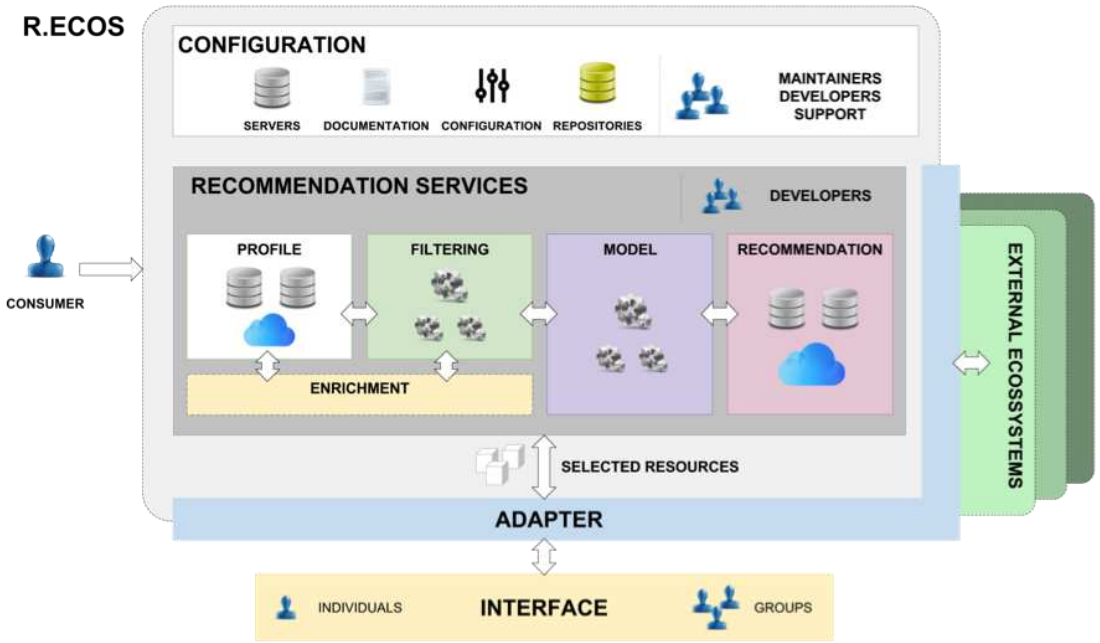

Figure 4 – The R.ECOS architecture, with its layers and social and technological components

The definition of a consistent and easy-to-use API favors the ecosystem, enabling the consumption of available solutions in addition to helping in its integration with external software ecosystems. The R.ECOS-API follows the REST architecture standards. From the user's point of view, R.ECOS-API works like a black box, without prior technical knowledge to make requests when following the standards available in the platform documentation.

4.5. FEASIBILITY STUDIES

Considering the solution amplitude, this section presents a training evaluation of R.ECOS through two feasibility studies. The objective is to test the platform, verifying the feasibility of the architecture and technologies adopted in the proposal development. The main goal of a feasibility study is to create a sufficient amount of knowledge about the platform without trying, in principle, to find a definitive answer to the research question [37].

The platform was developed to promote integration with external services, using free and open-source solutions. Some features were added to the platform, such as non-restriction to technology when using a communication architecture that can be consumed by systems of different languages and standards; integration with external APIs already known by the community, encouraging the implementation of solutions within the R.ECOS; and recommendation services marketplace, where external users and developers can have access to existing solutions, promoting and facilitating the reuse of those services.

The R.ECOS platform was developed based on a Service-Oriented Architecture (SOA), with transparent communication with other solutions and services. The adoption of SOA was considered according to the following principles [33]: (i) development efficiency; (ii) service reuse; (iii) composition of applications from services; and (iv) agility, flexibility and alignment with an architecture that defines information standards and models, allowing compliance validation against the established standard.

In the context of this study, the technical feasibility is demonstrated, also the technologies chosen to develop the platform. Thus, the first feasibility study evaluates the operation of technological components adopted in R.ECOS, used by an external collaborator to register a new recommendation service on the platform. The second feasibility study presents the design of a recommender system on a website, through which video resources on specific subjects of the site are recommended. The structure used in the feasibility studies was adapted from the case study formalization presented by [38]: (I) study definition; (II) objective formalization; (III) planning; (IV) execution; and (V) presentation of the observed evidence. A synthesis of these feasibility studies is presented in Table 4 and Table 5.

Table 4 - Feasibility Study I

Registration of new services by external developers	
Description	This feasibility study evaluates the operation of the technological components of the platform. These components were used by an actor with an EXTERNAL DEVELOPER role, whose objective was to register a new recommendation service on the platform. The service developed by the external user is a PROFILE AND CONTEXT EXTRACTION, named *recos-profiling-researchgate*, which extracts information about users from the ResearchGate Social Network, analyzing the relationships of types “follows,” “followed” and “collaborates with.” The actors involved in this feasibility study are EXTERNAL DEVELOPER, MAINTAINERS, SUPPORT, and INDIVIDUALS. The technological components are PLATFORM, DISCUSSION FORUM, HELPDESK, OFFICIAL DOCUMENTATION, MARKETPLACE, SOCIAL NETWORK, and SERVICE. Figure 5 shows the technological and social components involved in this feasibility study. Its complete description is available at [3].
Observed Evidence	Some of the technological components identified in the literature review were validated, such as discussion forum, helpdesk system, and Marketplace. Forum and helpdesk proved to be relevant in streamlining troubleshooting and facilitating new services implementations. The Marketplace helped to publicize the services by presenting all solutions implemented and available for use in the platform. Although the available resources were enough to solve the problems faced by the external developer, s/he reported that it would be interesting that the platform had a chat box for instant communication. Among the actors involved in the feasibility study, the external developer was able to satisfy his/her need to implement and disseminate a new profile extraction service using the R.ECOS platform. For the implementation and availability of new services in R.ECOS, the external developer needed to have technical knowledge in the programming language (PHP), as well as the communication standards of R.ECOS-API, available in the documentation. According to the developer, the documentation was beneficial during the process. Still, s/he reported some points in the documentation that could be better described and presented, for instance, more tutorials and examples of the R.ECOS-API. Finally, evidence was found that the platform and its components facilitated the registering of new services by external developers, using the online platform, documentation, discussion forum, and helpdesk system.

[3] https://drive.google.com/file/d/1mFpJV7R0HyD4ohMx47On-edOxO28RcIa/view?usp=sharing

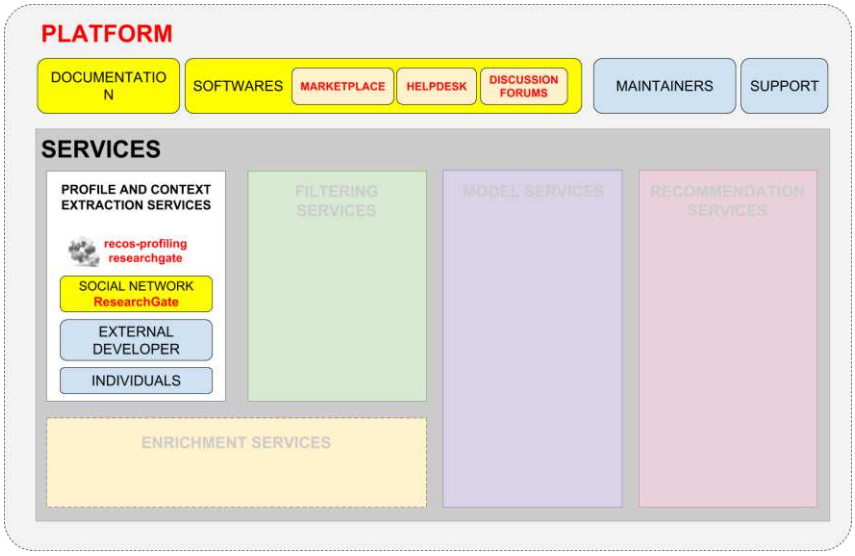

Figure 5 - Technological and social components of Feasibility Study I

Table 5 - Feasibility Study II

Recommendation of videos on school management	
Description	An external USER needs to incorporate recommendation components into his/her website, which deals with school management. The USER received the access link to R.ECOS to evaluate the platform regarding resource recommendation in an uncontrolled environment because unknown INDIVIDUALS access the website through online search services, such as the Google engine. In this study, services already existing in the R.ECOS MARKETPLACE, together with the SCOREBOARD used for decision making, were analyzed. The analysis considered two main aspects: (i) the quality of recommendations made on the proprietary website, where articles related to school management are published; and (ii) the use of the R.ECOS platform by the USER. The actors involved in this second feasibility study were: USERS, INDIVIDUALS, and SUPPORT. The technological components were: MARKETPLACE, SCOREBOARD, and R.ECOS-API. Figure 6 shows technological and social components present in this feasibility study, including the services that make up the recommender system created in the project. Its complete description is available at [4].
Observed Evidence	Looking at the progress of the feasibility study, the use of the platform by external USERS in the context of R.ECOS proved successful. Also, the implementation of a recommender system on a website with the help of R.ECOS proved viable. Through an informal interview with the USER, some aspects of the platform were highlighted. The scoreboard provided the USER with information to analyze the system behavior, enabling him/her to improve the quality of recommendations. The USER regarded this as a significant contribution because, without these indicators, s/he would not be able to know if the recommended resources were matching INDIVIDUALS' interests. Another highlight pointed out by the USER was the easy way to change the services of his/her project. Since s/he immediately visualized the changes in the indicators, s/he could see an improvement in the recommendation adherence. Also, the USER was able to develop the recommender system and integrate it into his/her website, demonstrating that the R.ECOS platform has components that enable and support such implementation. The use of the platform by the USER did not require previous knowledge of the technologies thereof. However, the actor with a SUPPORT role was needed to know the standards of R.ECOS-API and the technologies used in the USER's website, given that a small intervention in the website source code was necessary. In conclusion, it is possible to state that the objective of the study was achieved as it enabled the creation of a Recommender System composed of services available on the online platform.

[4] https://drive.google.com/file/d/1wKMmVOPR3t0smQeFcGvwtHi9u1j_kvg-/view?usp=sharing

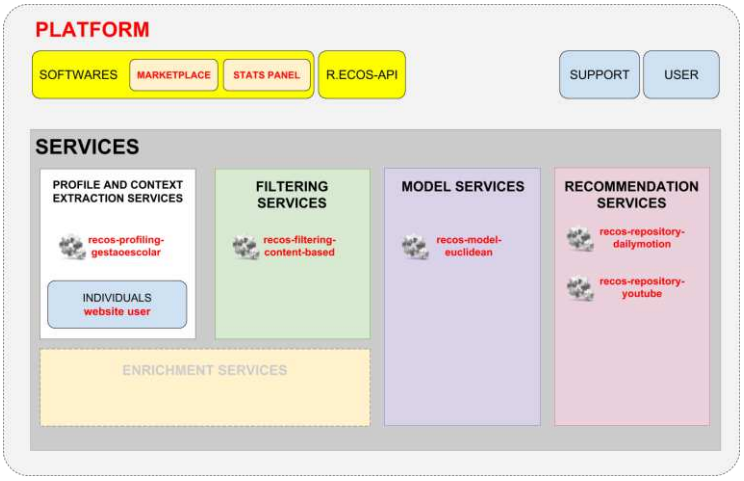

Figure 6 - Technological and social components of Feasibility Study II

4.5.1. Analysis of the Feasibility Studies Results

The feasibility studies relied on informal data collection, whose main results were described in the Observed Evidence. From the presented evidence, the technical feasibility of the platform was confirmed. It offers components that can facilitate external actors' contributions, and the platform can recommend resources in the interface desired by the user. The technologies were sufficient for the full operation of the platform. The first indications of technical feasibility arose when carrying out these studies. Problems in the platform were identified and corrected (such as pages not found or broken links), as well as some adjustments (usability and organization of information for users), improving relevant aspects.

After analyzing the results of the feasibility studies, some improvements were necessary for the platform. After that, we carried out two case studies to increase the reliability of the evaluation process of the present proposal.

5. R.ECOS EVALUATION

In this section, the following characteristics of R.ECOS were evaluated: the integration with other ecosystems and API, the quality of recommendations produced by the proposed solution, and the contribution of external actors. The scope of each case study was defined according to the Goal/Question/Metric (GQM) approach [39], and are described as follows: "***To analyze*** *the R.ECOS platform, considering its technology, actors, and relations* ***for the purpose of*** *verifying the viability of supporting users* ***in relation to*** *the development, reuse, and sharing of Recommender System solutions* ***from the point of view*** *of stakeholders and actors involved* ***in the context*** *of engineering software for recommender systems."*

Based on the scope defined above, the general Research Question (RQ) was formulated: "*How do the proposal of a recommender system approach, from a software ecosystem perspective, favor the development, reuse, and sharing of recommendation solutions, allowing for integration with other systems and platforms?*".

This research question unfolds into the following specific questions:

RQ1: Is it possible to allow the reuse of recommender system solutions, considering external users?

RQ2: Is it possible to develop a recommender system using the R.ECOS platform?

RQ3: Is it possible to allow the integration with external solutions enhancing reuse and sharing of RS solutions?

RQ4: Is it possible to recommend personalized resources to both users and groups of users?

Two case studies were conducted to answer the above questions, each with specific objectives based on GQM. A case study, in the context of Software Engineering, is an empirical investigation based on different sources of evidence, used when the object of the study is a contemporary phenomenon that is difficult to study in an isolated way. The main advantages of a case study are the ease of planning and the fact that they are relatively more realistic. At the same time, a disadvantage can be the difficulty in generalizing and interpreting the data obtained.

This evaluation contributes to verify the developed artifacts (platform and technological components) and to answer the research question. In order to conduct these case studies, adapted steps were used according to the case study pattern introduced by [38].

5.1. Case Study I - Integration with external services and software ecosystem

Case Study I consisted of evaluating the integration of R.ECOS with external services and another software ecosystem. This integration involved BROAD-ECOS [33] – an e-Learning ecosystem that required a system for educational resources recommendation, and MeaningCloud and YouTube APIs.

BROAD-ECOS represents an approach from an e-Learning ecosystem perspective that identifies individuals, communities, organizations, and software resources in an environment. It defines the architecture to transform existing e-Learning environments into platforms that allow external services integration and favor the development, reuse, and sharing of compatible educational services in an inter-organizational context.

The Virtual Learning Environment (VLE) used in this study was Moodle, which is open-source and free for commercial use, with an active online community. Currently, there are more than 125 million Moodle users worldwide, with more than 93,000 registered websites [40].

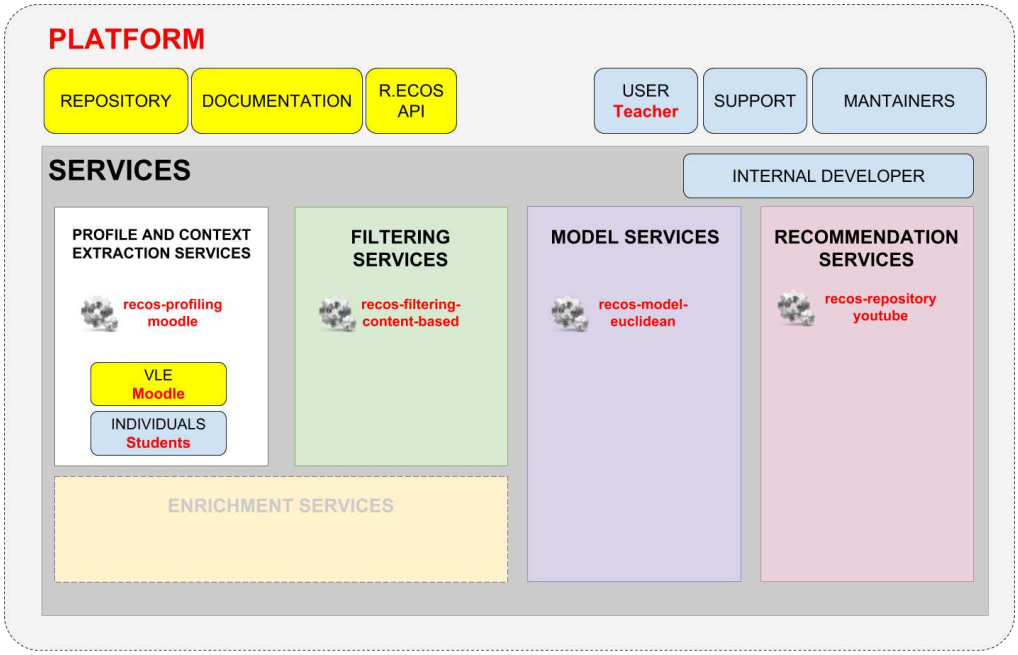

Figure 7 - Technological and social components of Case Study I.

The integration was implemented to assist teachers and students in the use of the VLE in question. In the context of this case study, the TEACHER wanted to recommend educational resources to distance learning students who use Moodle at the Federal University of Juiz de Fora, Brazil. The class had a total of 55 students, and the recommendation needed to be automatic and personalized to each student's profile. The educational profile in this study is defined at the cognitive and behavioral levels (students' involvement in the VLE activities and forums) [25].

For the R.ECOS platform to recommend educational resources, it was necessary to obtain student profiles and context and present the resources in the student's Moodle class. As BROAD-ECOS can receive requests for information about students, classes, and activities, the integration between the two software ecosystems seeks to share and reuse solutions about e-Learning and recommender system.

The actors involved in this case study were MAINTAINERS, INTERNAL DEVELOPERS, SUPPORTERS, INDIVIDUALS (in this study as students), and USER (the teacher). The technological components were PLATFORM, REPOSITORIES, SERVICES, R.ECOS-API, and DOCUMENTATION. Figure 7 shows the technological and social components present in Case Study I, including the services that compose the system.

Objective

The purpose of this case study is **to analyze** the R.ECOS platform and its components **in order to** demonstrate its integration with external services and software ecosystems **with respect to** components available on the platform **from the point of view of** the involved actors **in the context of** educational resources recommendation.

Planning

The BROAD-ECOS platform is already integrated with Moodle [33], but no service extracts students' profiles and context. Thus, an actor with a DEVELOPER role in BROAD-ECOS had to implement a new service, named

BROAD-MOODLE-INFO, where information about STUDENTS' educational profiles is returned, besides information about the class activities.

In R.ECOS, an actor with DEVELOPER role, either internal or external to the platform, implements the services that compose a project for resources recommendation according to the STUDENTS' needs. Two of these services must integrate with the BROAD-MOODLE-INFO service (BROAD-ECOS): *recos-profiling-moodle* and *recos-repository-youtube*. Other services that make up this recommender system are the content filtering service (*recos-filtering-contentbased*) and the memory-based model category service (*recos-model-euclidean*), the latter using Euclidean distance to calculate the similarity between resources and students.

Figure 8 presents an overview of integration solutions between R.ECOS and BROAD-ECOS. The execution of this case study is described below, explaining the process of requests between the platforms.

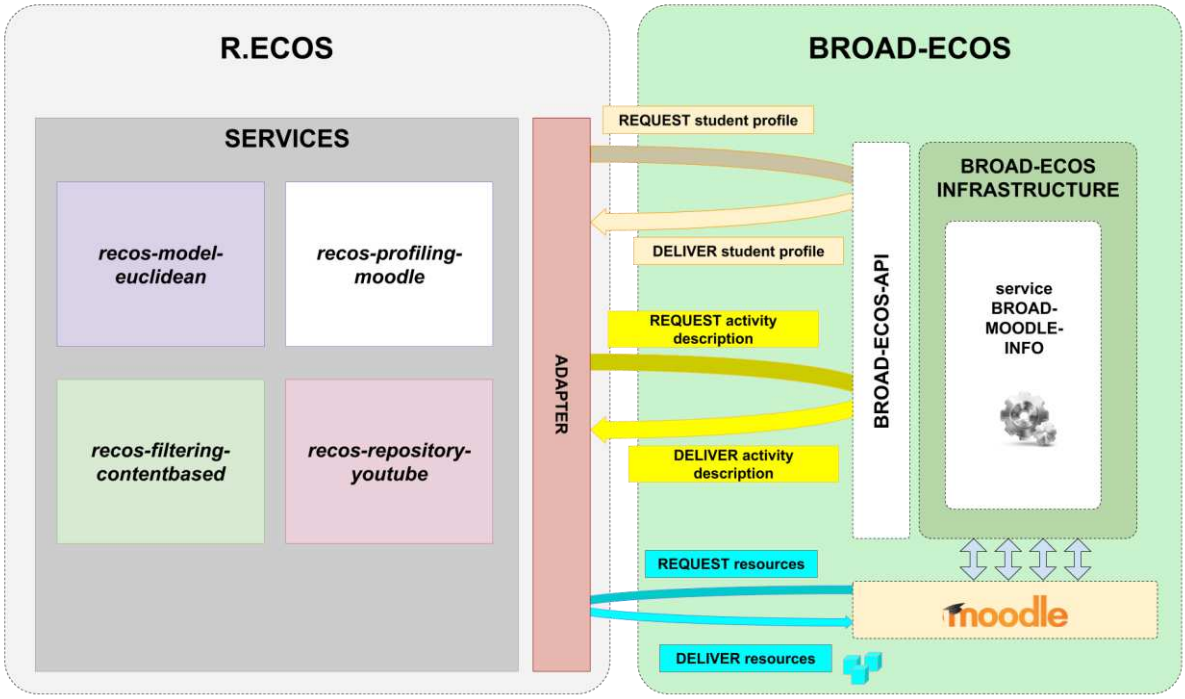

Figure 8 - Integration between R.ECOS and BROAD-ECOS.

Execution

With the view to enable students (INDIVIDUALS) to receive further recommended resources to complement the content of their class, the teacher (USER) requested that educational resources be recommended considering the profile and context of the students. Since R.ECOS can develop recommender systems as a module of another system but does not have an interface with Moodle, it was necessary to reuse the services already available in BROAD-ECOS. Thus, to reduce the development time of new services, the integration between two platforms that support Software Ecosystems, R.ECOS and BROAD-ECOS, was provided.

The INTERNAL DEVELOPER built the *recos-profiling-moodle* service to perform requests in BROAD-ECOS, through BROAD-ECOS-API, and to obtain information on the educational profile of the STUDENTS. The INTERNAL DEVELOPER also built the *recos-repository-youtube* service to enable the semantic analysis of the activities description. This service was responsible for making requests to other services outside R.ECOS and BROAD-ECOS, in particular *MeaningCloud API*[5]. This API performs the semantic analysis of texts, identifying the main concepts and entities. To use the YouTube API, the DEVELOPER had to register with Google services, access the Google APIs panel, and register the API key.

After the previous steps, information about the extracted videos was obtained, such as descriptions, titles, and access links. The *recos-repository-youtube* service filters the resources according to some STUDENTS' features to improve recommendation quality. Some used filters are: evaluations already carried out by the student that are stored in the REPOSITORIES (Moodle); access to class forums; reported questions; messages exchanged with the TEACHER. Finally, through the adapter layer, the service presents, in the VLE Moodle account of each student, a personalized list with three videos classified as most appropriate at the moment of the student's access.

Having confirmed his/her registration and login in the R.ECOS platform, the TEACHER was allowed to register a new project. After confirming the project registration, the token of the project was delivered. With

[5]https://www.meaningcloud.com/developer/apis

the project registered, an actor with a SUPPORT role needed to assist the TEACHER during the Moodle environment configuration. The SUPPORT helped the TEACHER to make an intervention in the Moodle source code to enable requests in R.ECOS-API. Figure 9 shows part of the VLE Moodle of a student with three resources recommended to him/her, along with the mechanism of recommendation evaluation. The student can choose either "liked" (stating that this recommendation was appropriate for his/her current needs) or "disliked" (saying that the resource was not relevant to him/her at the time).

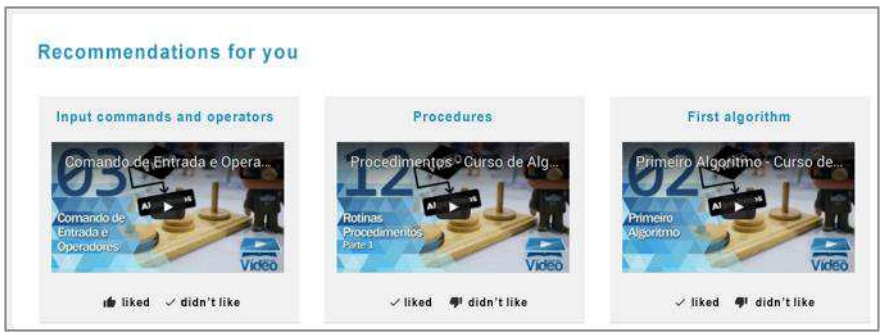

Figure 9 - Resource recommendation with the evaluation device on Moodle.

With the users' feedback and the evaluations about the R.ECOS platform' REPOSITORY, it is possible to adjust the preferences of each STUDENT according to the recommended resource features, thus leading to better recommendations.

Observed Evidence

This case study examined how R.ECOS integrates with other services, such as proprietary API and even other SECO. In addition to the integration with BROAD-ECOS, integrations were also possible with proprietary APIs, such as MeaningCloud API and YouTube API. Another evaluation carried out referred to the components present in the R.ECOS platform. Among the components and their relationships, this case study highlights the activities carried out by MAINTAINERS, INTERNAL DEVELOPER, and TEACHER (intending to recommend educational resources to his/her STUDENTS).

It was possible to evaluate one of the ways to present resources in the Moodle through the Adapter layer and to make an intervention in the VLE source code when it was necessary. This intervention took place by following the examples available in R.ECOS DOCUMENTATION.

Some limitations were observed while using the external services, as they were beyond the control of the maintainers and platform users. For example, the MeaningCloud API has a limit of up to 20,000 requests per month, with only two requests per second. Beyond these limits, the service using this external API may underperform.

As regards the actors, the platform and technologies used thereof by the TEACHER did not require much technical knowledge. They were able to register projects according to the services available. However, the TEACHER reported that the documentation, although assisting in the environment settings, was insufficient to incorporate a recommender system into Moodle. As a result, other materials were added to R.ECOS DOCUMENTATION.

From the SUPPORT perspective, knowledge about PHP programming language was required, as an intervention in the Moodle code was necessary. Concerning MAINTENANCE and INTERNAL DEVELOPER, the full understanding of the platform and the technologies involved in it was mandatory; however, it was not a barrier as the actors had participated in the design and implementation process of the R.ECOS platform.

Considering the results, the objective of this case study was achieved and showed that it is possible to integrate different software ecosystems, services, and external APIs to perform resource recommendations.

5.2. Case Study II - Recommendation for groups of students

Case Study II used a Moodle database designed for the Algorithm class, part of the curriculum of the Computer Science Program of the Federal University of Juiz de Fora (UFJF), Brazil, with 128 enrolled students. The educational profile of each student was defined by their test results, grades, and participation in discussion forums [25].

In a class, it is often necessary to organize groups of students to carry out activities or identify current levels of knowledge about a given topic. Recommender Systems can enhance learning through recommendations adherent to the profile of a group of individuals. Identifying a group of students with similar characteristics helps the teacher to select educational resources that support students learning and recap class subjects. Also, it increases the students' engagement in in-class activities.

In this second case study, an automatic resource recommendation service for groups was developed and made available on the R.ECOS platform. The clustering algorithm was performed through the integration between the Ecosystems formed in Case Study I to obtain students' information. The clustering service extracts profile and context, named *recos-profiling-moodle-group*, to form groups for the recommendation of resources to them (groups), not for users individually. Other services that made up this system were the *recos-filtering-content-based* (content-based filtering service); the *recos-model-euclidean* (memory-based model service) using Euclidean distance to calculate educational similarity among students; and a recommendation service named *recos-repository-algorithms* with resources in different formats and contents about the Algorithm class.

The service developed uses the homogeneity approach to identify the groups whose members have similar characteristics. The clustering process is automatic, categorized as ***auto clustering detection***. Data were collected implicitly in Moodle using data mining algorithms for group creation.

The TEACHER wanted to group STUDENTS according to the syllabus learning objectives and recommend resources to the group in which these students were distributed. In this way, s/he expected to improve the STUDENTS' performance and identify students that were about to drop out for their poor performance and who had not interacted with the virtual environment.

The actors of this case study were the USER (teacher), INDIVIDUALS (students), GROUPS of students, and the INTERNAL DEVELOPER. The technological components were PLATFORM, SERVICE, REPOSITORIES, and SOFTWARE (such as MARKETPLACE and VLE).

Objective

The objective of this case study is **to analyze** the R.ECOS platform and its services **in order to** develop and provide a clustering service **with respect to** the similarity of students' educational profile **from the point of view of** the teacher **in the context of** reuse and sharing of educational resources recommendation services for groups.

Planning

When recommending resources for groups of individuals with similar profiles, it is necessary to consider which information could be used and analyzed. In VLE Moodle, several kinds of information are available, such as grades on assignments and evaluations, participation in forums and chats, number of accesses, among others. For this study, information about the results of class evaluations and the level of student interactivity in VLE were considered. To obtain these data, the services described in Case Study I were used, configuring integrations with BROAD-ECOS based on the VLE Moodle used in the Algorithms class.

Execution

Before using the R.ECOS MARKETPLACE services, actors with a DEVELOPER role need to implement these SERVICES. For the current case study, an INTERNAL DEVELOPER had to implement a *group profile extraction* (*recos-profiling-moodle-group*) SERVICE. After testing it, the service was available on MARKETPLACE for R.ECOS' users.

The use of the *recos-profiling-moodle-group* service – an adaptation of the *recos-profiling-moodle* service – allowed obtaining the educational profile of all the students enrolled in the Algorithm class. It was possible then to analyze and perform similarity calculations among all students, getting STUDENT GROUPS with similar educational profiles.

Based on state of the art, different strategies were identified to create users' groups. In this case study, the proximity between the values of their attributes was analyzed, which is a common clustering strategy. After calculating the similarities among the students, the service disregards relationships between students with lower values, i.e., it only considers users to be similar once they have a high similarity value. Figure 10(a) shows the graph of a student with his/her similarities calculated concerning all the other students, where edges are the similarity values and nodes are students. Figure 10(b) presents the same student with the most relevant similarities. Following this reasoning, graphs with all relationships among students of the class were defined and, later, edges with low similarity value were disregarded and removed. In this process, we consider the average and standards deviation of the edges to set the lower threshold. The tool used to build the graphs was the Neo4J[6] database.

[6] www.neo4j.com

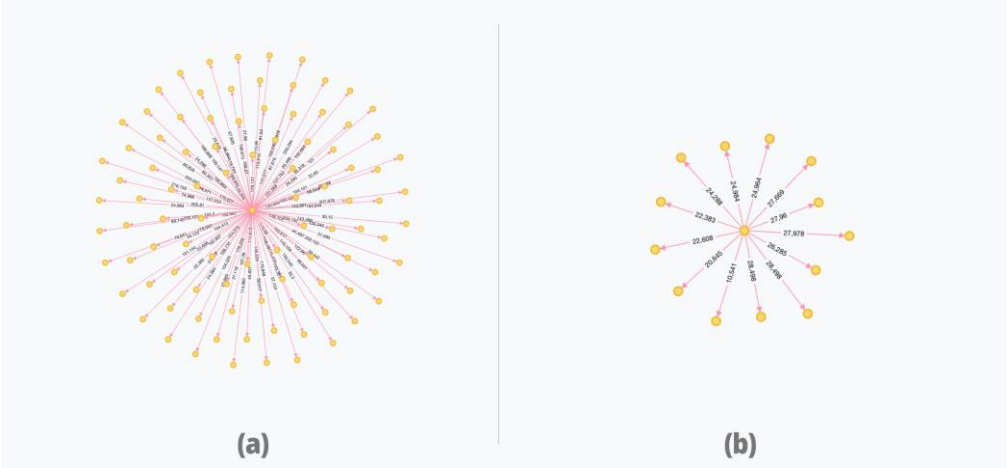

Figure 10 - (a) The student with similarities calculated among all the other students. (b) The same student with the most relevant similarities to the group.

Following this criterion, the *recos-profiling-moodle-group* performs the clustering of these students and returns student groups along with their profiles, allowing recommending educational resources appropriate for the groups' profiles. Representation of students and their similarities is displayed in Figure 11, where students are grouped according to their most similar peers.

In the class analyzed, three groups of students were identified by the service. The **first group** consists of students who have poor academic performance (below the class average minus standard deviation) and do not actively participate in the VLE, i.e., students who are likely to drop out. The **second group** consists of students with average academic performance and moderate participation in forums and activities. These students need attention but may receive recommendations for more advanced educational resources. Finally, the **third group** consists of students with good academic performance (above the class average plus standard deviation). Although many teachers believe that third-group students do not need special attention, it is essential to keep them motivated. The recommendation of educational resources addressing advanced subjects can motivate them.

In order to recommend resources to each group, the *recos-repository-algorithms* recommendation service was used, where resources were obtained according to the needs of the groups, also considering different media, such as videos, scientific articles, and educational games, following the class plan defined by the TEACHER.

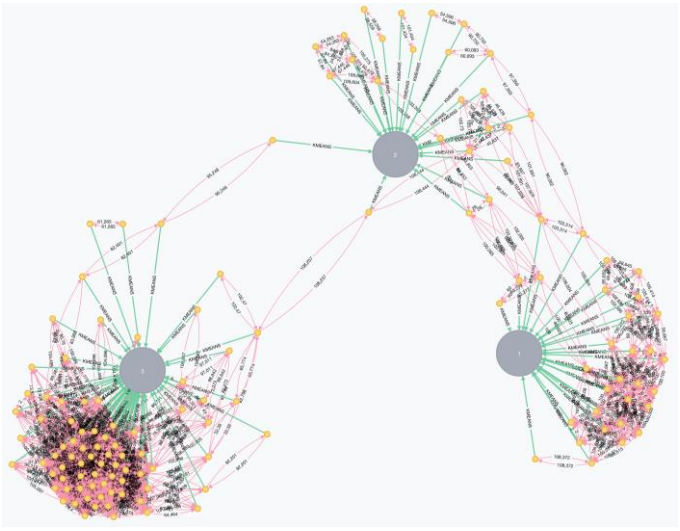
Figure 11 - Graph representing groups of similar students.

The **first group** received basic resources, and to encourage student participation, some resources such as educational games were provided. The use of resources with gamification characteristics promotes the involvement of students in VLE activities [41].

The **second group** received intermediate and advanced resources because they are students who have already demonstrated basic knowledge about the topic. The resources selected were videos and scientific articles.

The **third group** received resources with advanced information in an attempt to improve their knowledge and maintain students motivated. The teacher wanted to deliver these recommendations according to

the students' profiles in the group, showing that they were prepared for other topics beyond those previously presented in class. Recommended resources include videos with advanced content and scientific articles.

The recommendations were evaluated by the teacher by tracking his/her students learning progress throughout the course: (i) students at risk of dropping out of the class continued studying, (ii) performance of students in groups 1 and 2 improved, and (iii) students in the third group enjoyed the advanced content suggested.

Observed Evidence

In this case study, it was possible to group students attending Algorithm class considering their educational profile similarities, such as grades on tests and activities and their behavior in the VLE. Students' behavior was characterized by their access to both VLE and available materials and their participation in discussion forums.

A novelty here concerning the previous case study is the possibility of extracting profiles of individuals and grouping them using the same profile/context extraction service. This service identifies groups of students along with their profiles, rather than returning only individual profiles.

When recommendations were addressed to groups of students, it was possible to overcome one of the limitations encountered when using an external API, such as MeaningCloud API. Since it is a proprietary API, and after some requests, it becomes a paid service, the clustering strategy is attractive because fewer requests are made for the resources to be returned. For example, in a group of 30 students using an individual recommendation strategy, 30 requests are made to the API. In comparison, a group of 30 students with similar profiles requires only one request to the same API.

The actors involved in the case study were INDIVIDUALS (students) and GROUPS formed by students, whose profiles were extracted from specific information about their behavior. This service can be reformulated if necessary, but in this case, only the actors with an INTERNAL DEVELOPER role are authorized to do so.

The case study showed that it is possible to group individuals using the R.ECOS platform and its components and recommend resources to groups of users.

5.3. Discussion

In order to verify whether the aims were achieved, it is worth recalling the research question previously established, which was divided into four secondary questions. Each of them will be resumed below.

Research Question: "*How do the proposal of a recommender system approach, from a software ecosystem perspective, favor the development, reuse, and sharing of recommendation solutions, allowing for integration with other systems and platforms?*".

Again, to address this question, four secondary ones were proposed, each of them evaluating a particular aspect.

RQ1: Is it possible to allow the reuse of recommender system solutions, considering internal and external users?

Through the R.ECOS platform, it was possible to enable users to share and develop their recommender system solutions but also to reuse others. The **existence of a recommendation services marketplace** was decisive during this process because the recommendation services designed during Case Study I were made available for reuse. In Case Study II, a clustering algorithm service was developed and made available on Marketplace, while other recommendation services were reused to design the Recommender System for groups of students. In this vein, we can consider that the Marketplace allows users to both share and reuse solutions.

RQ2: Is it possible to develop a recommender system using the R.ECOS platform?

As observed in both case studies, it is possible to develop recommender systems using R.ECOS. As seen during the evaluation, some particularities had to be contemplated by the platform, but users were able to create and use their recommender systems.

RQ3: Is it possible to allow the integration with external solutions enhancing reuse and sharing of RS solutions?

A characteristic raised in the present proposal was the **non-restriction to technology** for solutions generated in the R.ECOS platform. This requirement was possible by using communication standards in the JSON format, where systems with the most diverse programming languages can read and process the information. In the Case Study I, the return of all requests made for R.ECOS-API use JSON notation.

Another characteristic evaluated was the **integration with external services already used by the community**, encouraging the use and also the implementation of new solutions within the R.ECOS platform. Case Study I presented the integration between R.ECOS and BROAD-ECOS – a software ecosystem for e-learning environments –, along with integrations between R.ECOS services and free YouTube API as well as with MeaningCloud API – another free API but with a limited number of requests.

RQ4: Is it possible to recommend personalized resources to both users and a group of users?

From Case Studies I and II, it was possible to verify the R.ECOS performance in building recommender system solutions in a real-world context. In all the cases, the main goal of building a recommender system was to present personalized resources to end-users. Notably, Case Study II demonstrated that it is possible to recommend to both individuals and groups of individuals using the R.ECOS platform.

In Case Study II, a clustering service was developed and made available in the R.ECOS platform, so other users can reuse it when developing recommender systems for groups of users. In this vein, R.ECOS can be regarded as a platform that facilitates the development, reuse, and sharing of solutions.

Considering the users' feedback, we can affirm that using the systems developed with the support of the R.ECOS platform recommending personalized resources to both users individually and to a group of users was enhanced.

With the presentation of feasibility studies, case studies, and the answers to the research questions, we have evidence that the R.ECOS' technological components can promote interaction between actors and the platform, in addition to facilitating the use of the components available for building recommender systems.

6. FINAL REMARKS

Dresch, Lacerda, and Antunes [42] have defined seven fundamental characteristics that must exist in a scientific research project: (I) creation of an artifact to (II) solve a specific problem, (III) whose utility must be explained through an appropriate evaluation of its applicability, and (IV) contributions and results of the study should be shared among stakeholders. To ensure its validity, (V) investigations must be carried out with rigor and (VI) possible forms of solution analyzed, and (VII) results should be disclosed and communicated to interested parties.

This study was therefore structured to follow the seven characteristics defined by the above authors, proposing (I) the R.ECOS platform to (II) centralize and systematize the development of Recommender Systems, favoring solutions reuse and sharing (III) to evaluate the technological platform by validating the mandatory social and technological components in a Software Ecosystem for Recommender Systems through literature reviews and an exploratory study with researchers in the relevant areas, and (IV) presenting the obtained results to the community through this paper and also making the platform available on the internet. We have maintained (V) methodological rigor during the platform development, (VI) analyzed possible solutions of libraries and frameworks for Recommender Systems, and finally (VII) published our research results.

Considering the research question that guided this study, we can affirm that there are signs that the idea and spread of a recommender system approach from a software ecosystem perspective do favor the reuse and sharing of solutions, allowing for integration with other systems and ecosystems.

The main objective of this study was achieved with R.ECOS, which allowed integrating solutions and encouraging the development, reuse, and sharing thereof. Specific goals, as the platform architecture definition with its technological and social components, and the development and availability of this platform were also fulfilled. The R.ECOS platform is online, working, and available at *http://www.diplomaster.com/recos*, along with its technological and social components.

Defining a recommendation services marketplace has proved of great value for the advance of recommender system research as it promotes the reuse and sharing of existing solutions, given that many users and researchers can benefit from them. Another contribution was the fact that the platform helps users with diverse roles, such as "*people who want to recommend resources on their system*," "*those who want to advance in RS research*," "*those who want to disseminate their solutions*," among others.

Although some frameworks and libraries facilitate recommender system research, proposals similar to R.ECOS were not found in recent literature. Therefore, another significant contribution of this paper is the fact that it promotes an advance in the *Software Ecosystem* and *Recommender System* areas.

Concerning the solutions implemented, the platform and its technological components are still prototypes. Regarding information security, no devices were defined to protect the platform and its information against malicious attacks. Neither were backup routines created for files and databases, although the platform was implemented following good development practices and is hosted on known and trusted hosting services.

Tests concerning platform interface and usability were not addressed in this study, so improvements are still necessary. No tests were carried out with many simultaneous requests and accesses, and this information is

not available in this study. At the current stage of the platform, there is not yet a significant number of users required to perform such analyses.

As future work, the proposed technological platform shall be continued, implementing improvements and making new technological components available. Specific components for integration with R.ECOS solutions can also be presented, such as plug-ins, to perform requests in R.ECOS-API in a more straightforward manner than that of the code intervention required in the recommendation interface.

Some other projects are being conducted by researchers at the NEnC, and these can be incorporated into the R.ECOS project, for instance, using sentiment analysis in user profile and context definition, using predictive models to improve recommendation performance, among others.

In future versions of R.ECOS, it will be possible to add new indicators to the platform and to the recommender systems built through it, which have not been previously added as the platform is at its initial stage.

7. DECLARATIONS

7.1. Ethics approval and consent to participate
Not applicable

7.2. Consent for publication
Not applicable

7.3. Availability of data and material
The source code and the dataset supporting the conclusions of this article are not available in an online repository yet, but all data used could be shared when asked by email **victor.stroele@ice.ufjf.br**.

7.4. Competing interests
The authors declare that they have no competing interests.

7.5. Funding
The authors declare that they have no competing financial interests

7.6. Authors' contributions
Some specific author's contributions are:

- *André Abdalla*: data extraction, models development, result analysis, and article writing
- *Victor Ströele*: models development, result analysis, and article writing
- *Fernanda Campos*: students guideline, models development, result analysis, and article writing
- *Regina Braga*: students guideline, result analysis, and article writing
- *José Maria N. David*: students guideline, result analysis, and article writing

AUTHORS' BIOGRAPHY:

André Abdalla: Bachelor (B.Sc.) in Computer Science at Federal University of Juiz de Fora. Master (M.Sc.) in Computer Systems and Technologies, Software Engineering from Federal University of Juiz de Fora (2018).

Victor Ströele: BS in Computer Science from the Federal University of Juiz de Fora (2005), a Master's Degree (2007) and Ph.D. (2012) in Systems Engineering and Computer Science Program from the Federal University of Rio de Janeiro. He is currently Associate Professor II at the Federal University of Juiz de Fora. He has experience in Computer Science, with emphasis on Data Mining and Complex Network, working mainly on the following themes: Clustering Algorithm, Social Network Analysis, Recommender Systems, and Computers in Education.

Fernanda Campos: Graduated in Mathematics from Federal University de Juiz de Fora (1978). Received her master degree in Systems Engineering and Computer Science from Federal University of Rio de Janeiro in 1994. She is Ph.D. in Systems Engineering and Computer Science from the Federal University of Rio de Janeiro (1999). Professor Campos is currently a senior professor at the Federal University of Juiz de Fora, working on the undergraduate program in Computer Science and is a permanent member of the Master Program of Computer Science. Campos has experience in computer science, with emphasis on software engineering, specifically in the following topics: e-Science, e-Learning, Recommender Systems, and Ecosystems.

Regina Braga: Graduated in Computer Science from Federal University de Juiz de Fora (1991). Received her master degree in Systems Engineering and Computer Science from Federal University of Rio de Janeiro in 1995.

She is Ph.D. in Systems Engineering and Computer Science from the Federal University of Rio de Janeiro (2000). Professor Braga is currently an associate professor at the Federal University of Juiz de Fora, working on the undergraduate program in Computer Science and is a permanent member of the Master Program of Computer Science. Braga has experience in computer science, with emphasis on software engineering and databases, specifically in the following topics: Software Reuse, Ontologies, Data Integration, Ecosystems, and e-Science.

José Maria N. David: Bachelor's degree in Electrical Engineering from Military Institute of Engineering, IME, (1983), Master degree (M.Sc.) in Systems Engineering and Computer Science at Federal University of Rio de Janeiro (1991) and Doctorate (D.Sc.) at Systems Engineering and Computer Science at Federal University of Rio de Janeiro (2004). Professor David is currently an associate professor at the Federal University of Juiz de Fora, working on the undergraduate Program in Computer Science and is a permanent member of the Master Program of Computer Science. Has experience in Computer Science, focusing on Software Engineering, acting on the following subjects: Groupware, CSCW, CSCL, Software Ecosystems, and Middleware.

Acknowledgments

We would like to thank all the people who participated in the experiments. This study was financed in part by CAPES Foundation (*Coordenação de Aperfeiçoamento de Pessoal de Nível Superior*, Brazil) – Finance Code 001, UFJF, CNPq, and FAPEMIG.

REFERENCES

1. Internet Live Stats (2017) Internet usage and social media statistics. In: Website. http://www.internetlivestats.com/
2. Beel J, Gipp B, Langer S, Breitinger C (2016) Research-paper recommender systems: a literature survey. International Journal on Digital Libraries 17:305–338 . doi: 10.1007/s00799-015-0156-0
3. Jansen S, Brinkkemper S, Cusumano MA (2013) Software ecosystems: Analyzing and managing business networks in the software industry. Software Ecosystems: Analyzing and Managing Business Networks in the Software Industry 368 . doi: 10.4337/9781781955635
4. Felfernig A, Polat-Erdeniz S, Uran C, Reiterer S, Atas M, Tran TNT, Azzoni P, Kiraly C, Dolui K (2019) An overview of recommender systems in the internet of things. Journal of Intelligent Information Systems 52:285–309 . doi: 10.1007/s10844-018-0530-7
5. Karimi M, Jannach D, Jugovac M (2018) News recommender systems – Survey and roads ahead. Information Processing & Management 54:1203–1227 . doi: 10.1016/j.ipm.2018.04.008
6. Franco-Bedoya O, Ameller D, Costal D, Franch X (2017) Open source software ecosystems: A Systematic mapping. Information and Software Technology 91:160–185 . doi: 10.1016/j.infsof.2017.07.007
7. Albrecht CC, Dean DL, Hansen J V. (2005) Marketplace and technology standards for B2B e-commerce: Progress, challenges, and the state of the art. Information and Management 42:865–875 . doi: 10.1016/j.im.2004.09.003
8. Stroele V, Campos F, David JMN, Braga R, Abdalla A, Lancellotta PI, Zimbrao G, Souza J (2017) Data abstraction and centrality measures to scientific social network analysis. 2017 IEEE 21st International Conference on Computer Supported Cooperative Work in Design (CSCWD) 281–286 . doi: 10.1109/CSCWD.2017.8066708
9. Pereira CK, Campos F, Ströele V, David JMN, Braga R (2018) BROAD-RSI – educational recommender system using social networks interactions and linked data. Journal of Internet Services and Applications 9:7 . doi: 10.1186/s13174-018-0076-5
10. Horta V, Ströele V, Braga R, David JMN, Campos F (2018) Analyzing scientific context of researchers and communities by using complex network and semantic technologies. Future Generation Computer Systems 89:584–605 . doi: 10.1016/j.future.2018.07.012
11. Goldberg D, Nichols D, Oki BM, Terry D (1992) Using collaborative filtering to weave an information tapestry. Communications of the ACM 35:61–70 . doi: 10.1145/138859.138867
12. Schafer J Ben, Konstan J, Riedi J (1999) Recommender systems in e-commerce. In: Proceedings of the 1st ACM conference on Electronic commerce - EC '99. ACM Press, New York, New York, USA, pp

158–166

13. Resnick P, Iacovou N, Suchak M, Bergstrom P, Riedl J (1994) GroupLens : An Open Architecture for Collaborative Filtering of Netnews. Proceedings of the 1994 ACM conference on Computer supported cooperative work 175–186 . doi: 10.1145/192844.192905

14. Resnick P, Varian HR (1997) Recommender systems. Communications of the ACM 40:56–58 . doi: 10.1145/245108.245121

15. Hernando A, Bobadilla J, Ortega F, Tejedor J (2013) Incorporating reliability measurements into the predictions of a recommender system. Information Sciences 218:1–16 . doi: 10.1016/j.ins.2012.06.027

16. Bobadilla J, Ortega F, Hernando A, Gutiérrez A (2013) Recommender systems survey. Knowledge-Based Systems 46:109–132 . doi: 10.1016/j.knosys.2013.03.012

17. Kardan AA, Ebrahimi M (2013) A novel approach to hybrid recommendation systems based on association rules mining for content recommendation in asynchronous discussion groups. Information Sciences 219:93–110 . doi: 10.1016/j.ins.2012.07.011

18. Balabanović M, Shoham Y (1997) Fab: content-based, collaborative recommendation. Communications of the ACM 40:66–72 . doi: 10.1145/245108.245124

19. Pazzani MJ (1999) A framework for collaborative, content-based and demographic filtering. Artificial Intelligence Review 13:393–408 . doi: 10.1023/A:1006544522159

20. Dell'Amico M, Capra L (2008) SOFIA: Social filtering for robust recommendations. IFIP International Federation for Information Processing 263:135–150 . doi: 10.1007/978-0-387-09428-1_9

21. Deshpande M, Karypis G, Karypis G (2004) Item-Based Top-N Recommendation Algorithms. ACM Transactions on Information Systems 22:143–177 . doi: http://doi.acm.org/10.1145/963770.963776

22. Bosch J (2009) From Software Product Lines to Software Ecosystems. Proceedings of the 13th International Software Product Line Conference (SPLC 2009), San Francisco 111–119

23. Manikas K (2016) Revisiting software ecosystems Research: A longitudinal literature study. Journal of Systems and Software. doi: 10.1016/j.jss.2016.02.003

24. Campbell PRJ, Ahmed F (2010) A three-dimensional view of software ecosystems. Proceedings of the Fourth European Conference on Software Architecture Companion Volume - ECSA '10 81 . doi: 10.1145/1842752.1842774

25. Abdalla A, Ströele V, Veiga W, Simões L, Campos F, Braga R, David JMN (2017) R . ECOS – Educational Recommender Ecosystem. In: IEEE/ACM Joint 5th International Workshop on Software Engineering for Systems-of-Systems and 11th Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems. pp 48–54

26. Kitchenham B, Pretorius R, Budgen D, Pearl Brereton O, Turner M, Niazi M, Linkman S (2010) Systematic literature reviews in software engineering – A tertiary study. Information and Software Technology 52:792–805 . doi: 10.1016/j.infsof.2010.03.006

27. Gantner Z, Rendle S (2011) MyMediaLite: A free recommender system library. Proceedings of the fifth ACM conference on Recommender systems 305–308 . doi: 10.1145/2043932.2043989

28. Vengroff D (2011) RecLab: a system for eCommerce recommender research with real data, context and feedback. … Context-awareness in Retrieval and Recommendation 31–38 . doi: 10.1145/1961634.1961641

29. Garcin F, Faltings B (2013) PEN recsys: a personalized news recommender systems framework. Nrs 3–9 . doi: 10.1145/2516641.2516642

30. Schmedding M, Fuchs M, Klas CP, Engel F, Brock H, Heutelbeck D, Hemmje M (2016) Recalot.com: Towards a reusable, modular, and RESTFul social recommender system. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9679:402–406 . doi: 10.1007/978-3-319-35122-3_28

31. Simões L, Campos F, Ströele V, Braga R (2017) Sistema de Recomendação de Serviços Baseado em uma Arquitetura Aberta para um Ecossistema de Software. In: Anais do XIII Simpósio Brasileiro de Sistemas de Informação. SBC, Lavras, pp 340--347

32. Yang L, Bagdasaryan E, Gruenstein J, Hsieh C-K, Estrin D (2018) OpenRec: A modular framework for

extensible and adaptable recommendation algorithms. [WSDM2018]Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining 664–672 . doi: 10.1145/3159652.3159681

33. Veiga W, Campos F, Braga R, David JM (2016) A Software Ecosystem Approach to e-Learning Domain. In: Proceedings of the XII Brazilian Symposium on Information Systems on Brazilian Symposium on Information Systems: Information Systems in the Cloud Computing Era - Volume 1. Brazilian Computer Society, Porto Alegre, Brazil, Brazil, pp 75:574--75:581

34. Santos RP dos, Werner CML (2012) ReuseECOS: An Approach to Support Global Software Development through Software Ecosystems. 2012 IEEE Seventh International Conference on Global Software Engineering Workshops 60–65 . doi: 10.1109/ICGSEW.2012.16

35. Bosch-Sijtsema PM, Bosch J (2015) Plays nice with others? Multiple ecosystems, various roles and divergent engagement models. Technology Analysis and Strategic Management 27:960–974 . doi: 10.1080/09537325.2015.1038231

36. Manikas K, Hansen KM (2013) Software ecosystems - A systematic literature review. Journal of Systems and Software 86:1294–1306 . doi: 10.1016/j.jss.2012.12.026

37. Shull F, Mendoncça MG, Basili V, Carver J, Maldonado JC, Fabbri S, Travassos GH, Ferreira MC (2004) Knowledge-Sharing Issues in Experimental Software Engineering. Empirical Software Engineering 9:111–137 . doi: 10.1023/b:emse.0000013516.80487.33

38. Wohlin C, Runeson P, Höst M, Ohlsson MC, Regnell B, Wesslén A (2012) Case Studies. Experimentation in Software Engineering 55–72 . doi: 10.1007/978-3-642-29044-2_5

39. Basili VR (1992) Software modeling and measurement: the Goal/Question/Metric paradigm. University of Maryland at College Park

40. Moodle Trust (2013) Moodle.org: Moodle Statistics. In: Moodle Statistics. https://moodle.org/stats

41. Hamari J, Koivisto J, Sarsa H (2014) Does Gamification Work? -- A Literature Review of Empirical Studies on Gamification. 2014 47th Hawaii International Conference on System Sciences 3025–3034 . doi: 10.1109/HICSS.2014.377

42. Dresch A, Lacerda DP, Antunes Jr JAV (2015) Design Science Research. Design science research: A method for science and technology advancement. doi: 10.1007/978-3-319-07374-3

Figures

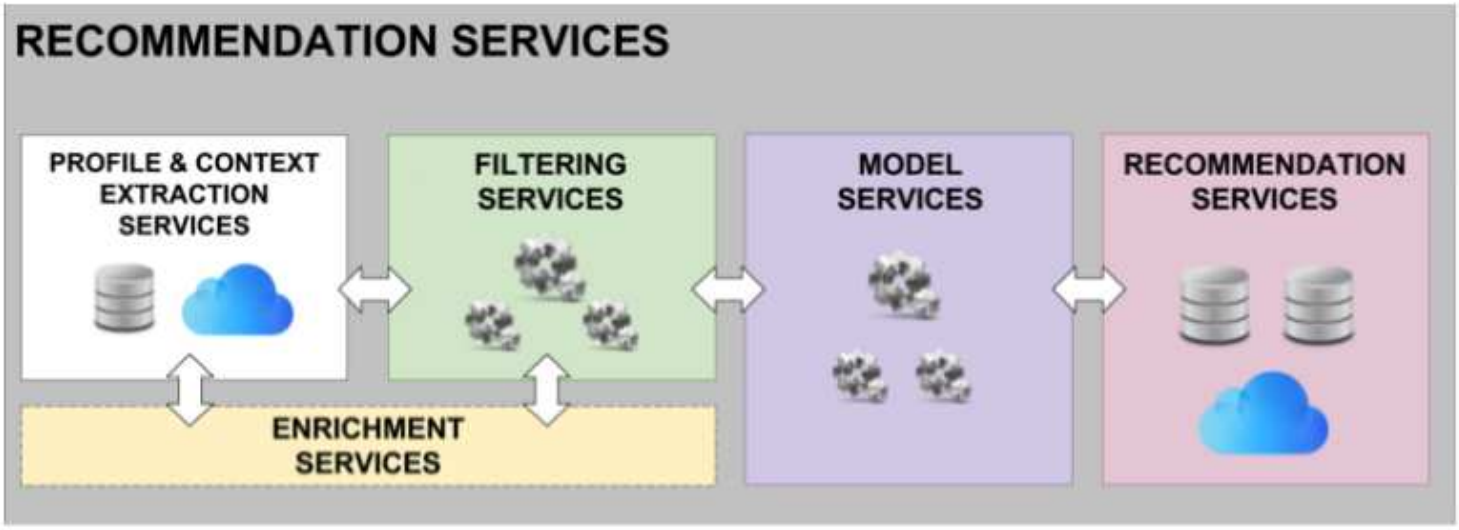

Figure 1

R.ECOS services

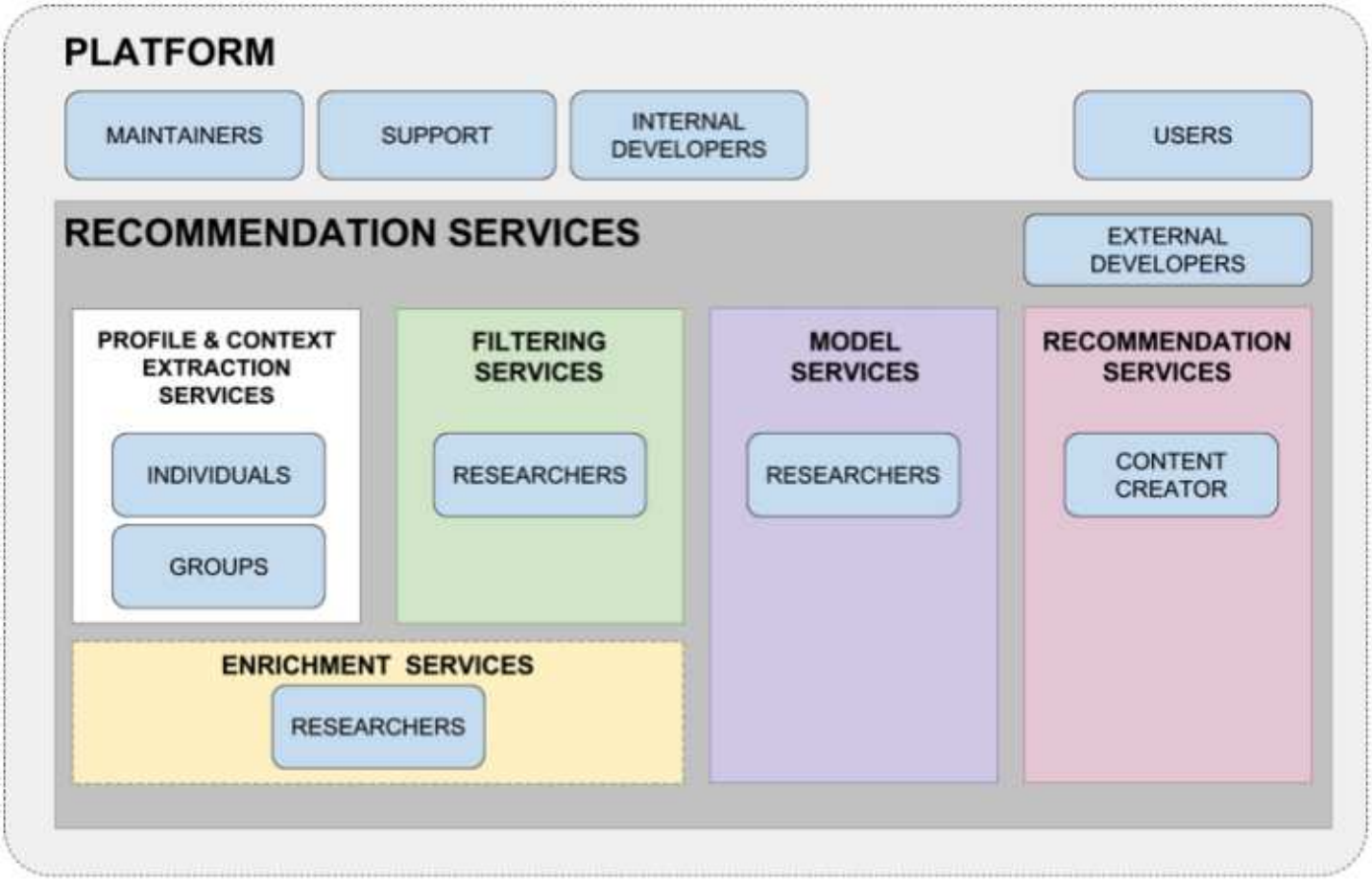

Figure 2

Social components and their interactions in R.ECOS.

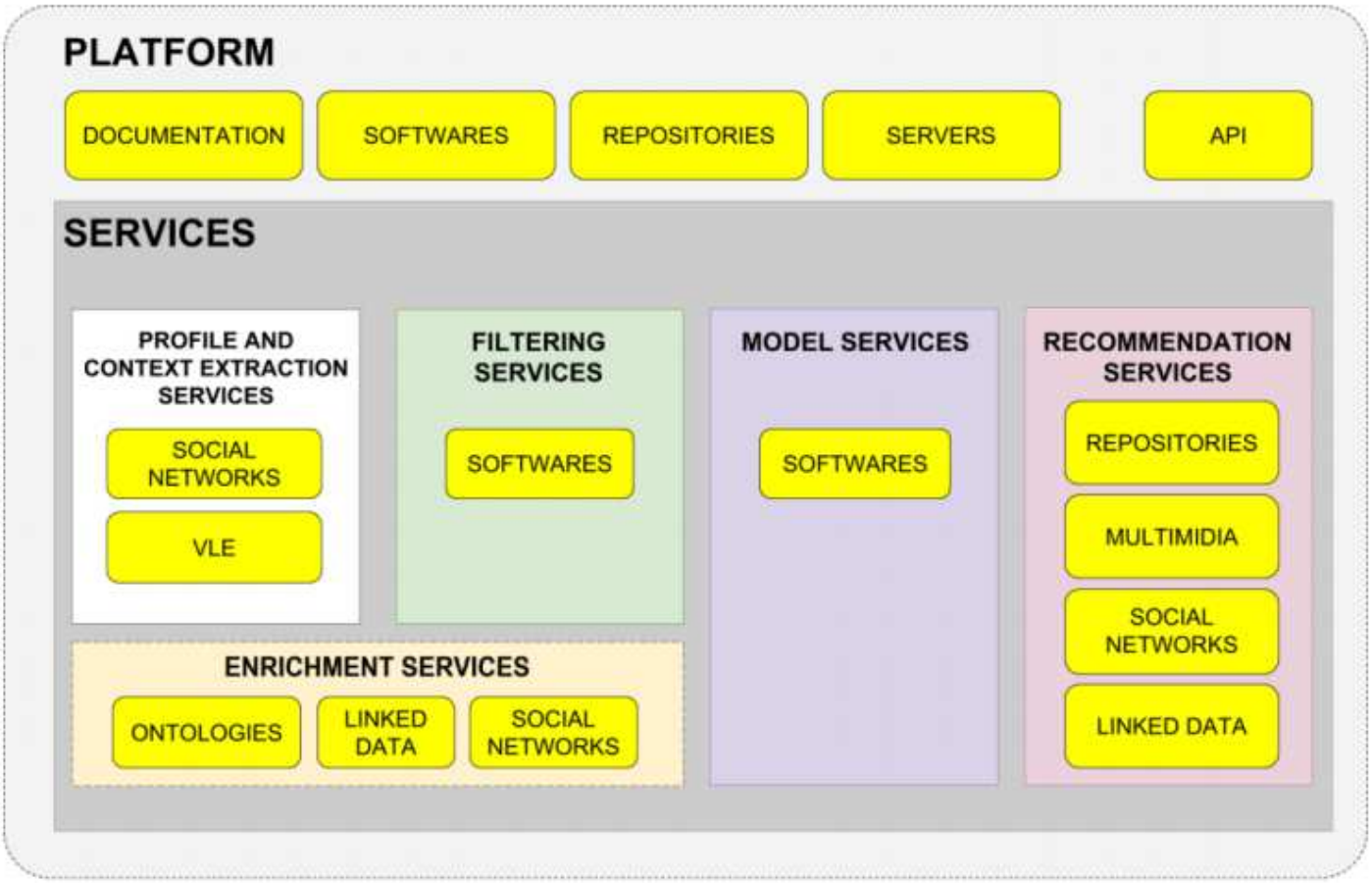

Figure 3

Technological components in R.ECOS

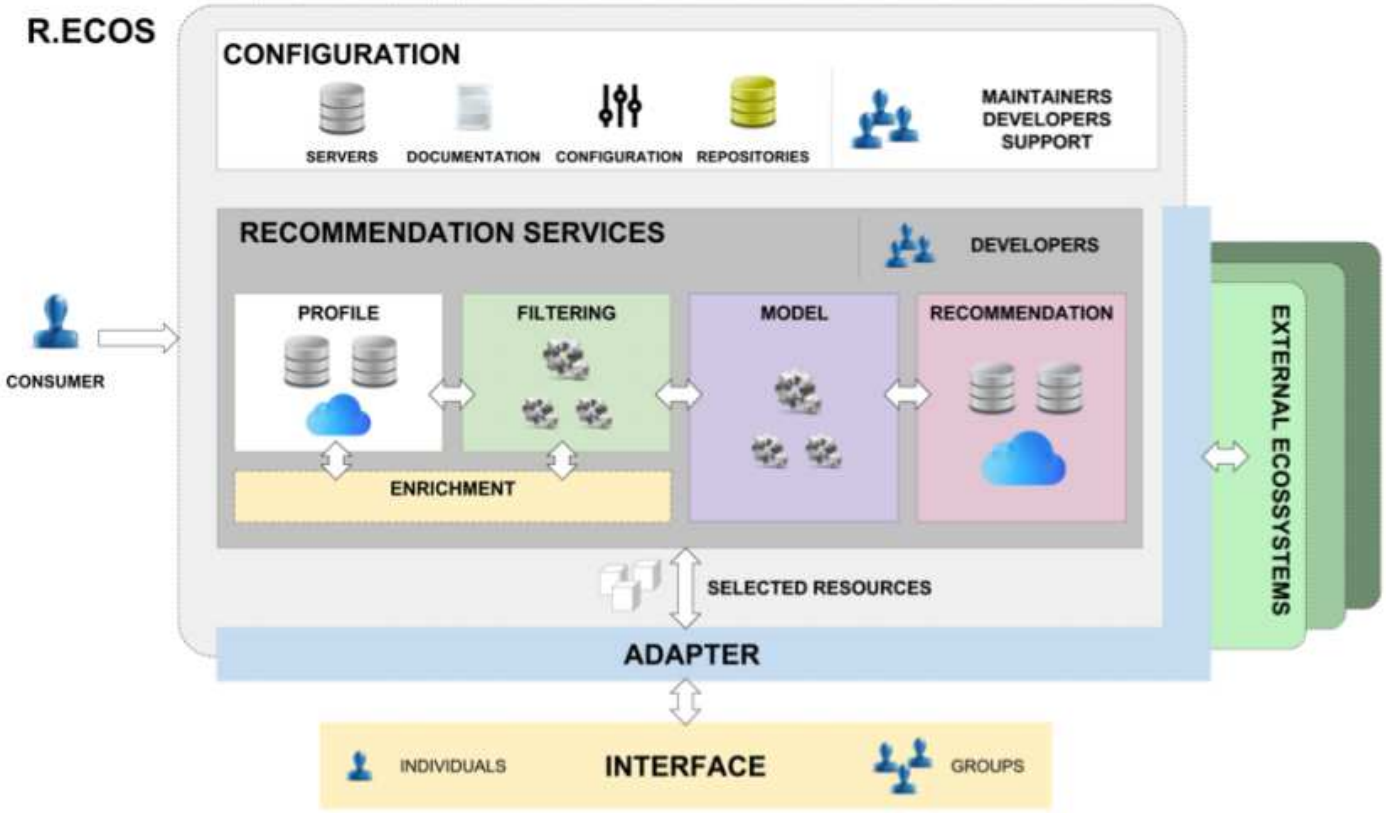

Figure 4

The R.ECOS architecture, with its layers and social and technological components

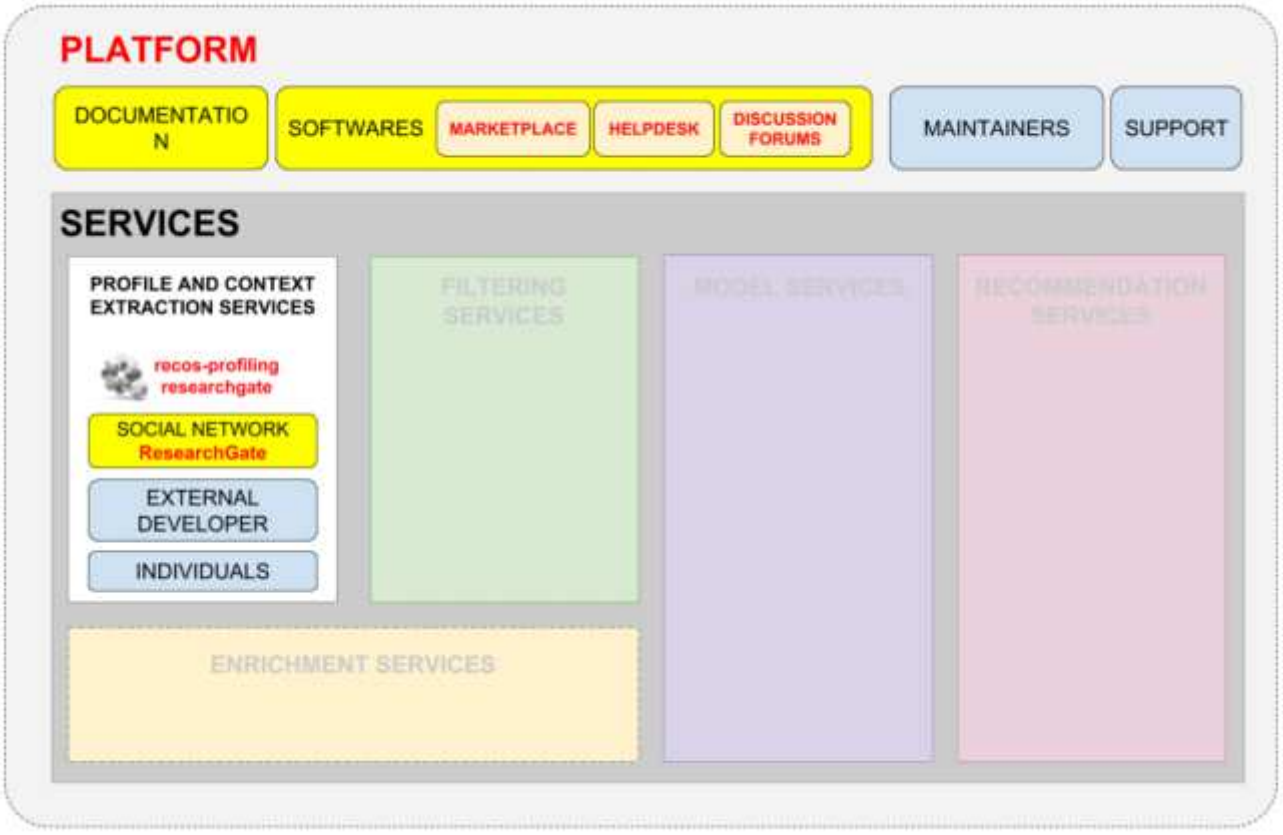

Figure 5

Technological and social components of Feasibility Study I

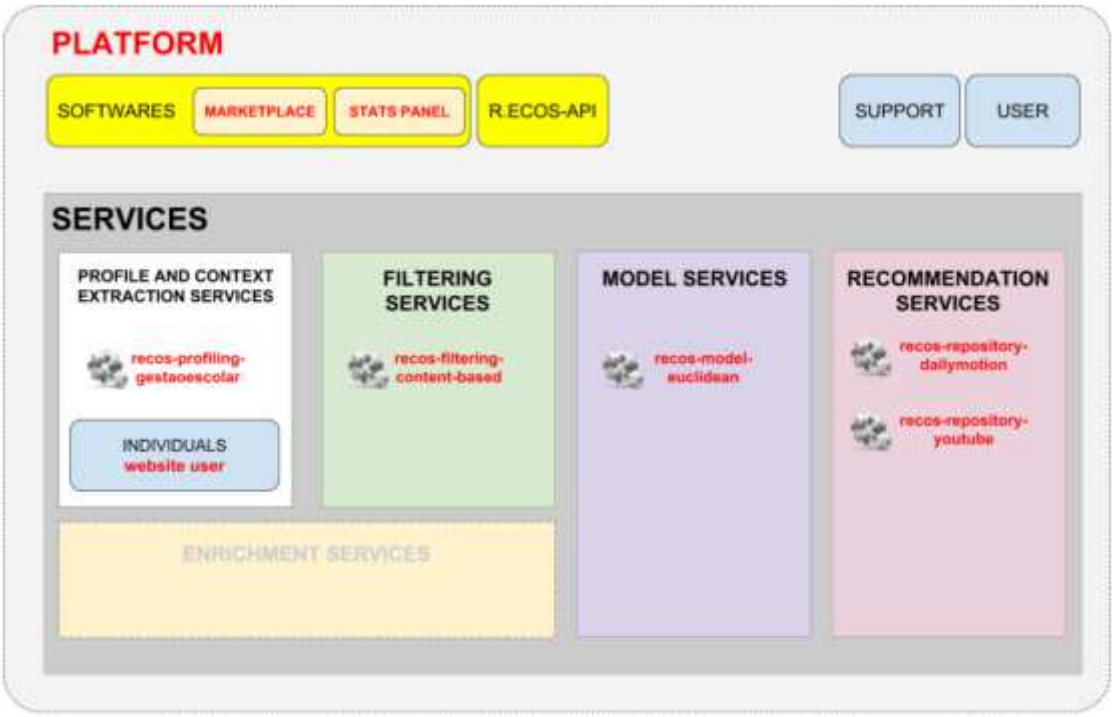

Figure 6

Technological and social components of Feasibility Study II

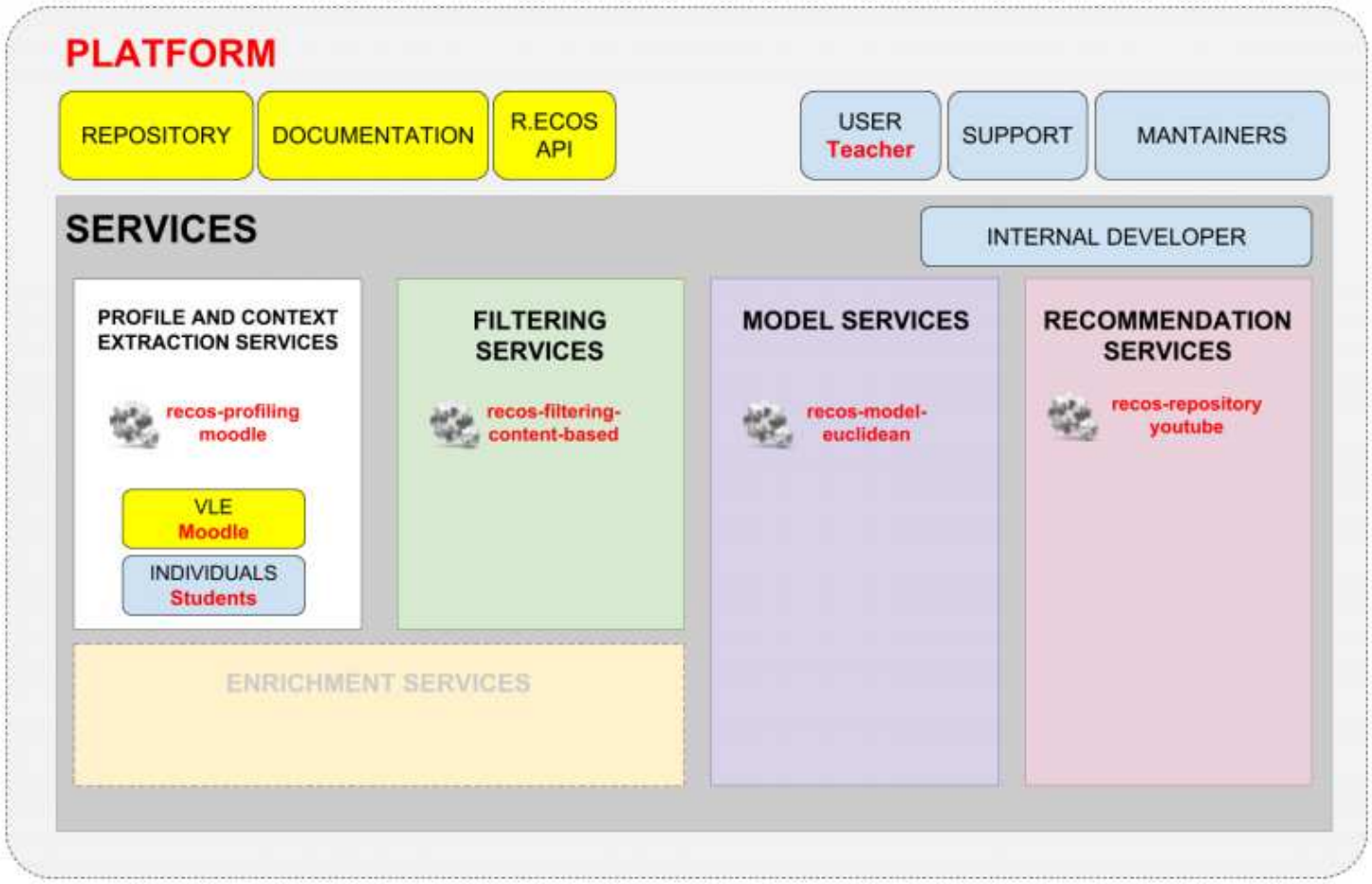

Figure 7

Technological and social components of Case Study I.

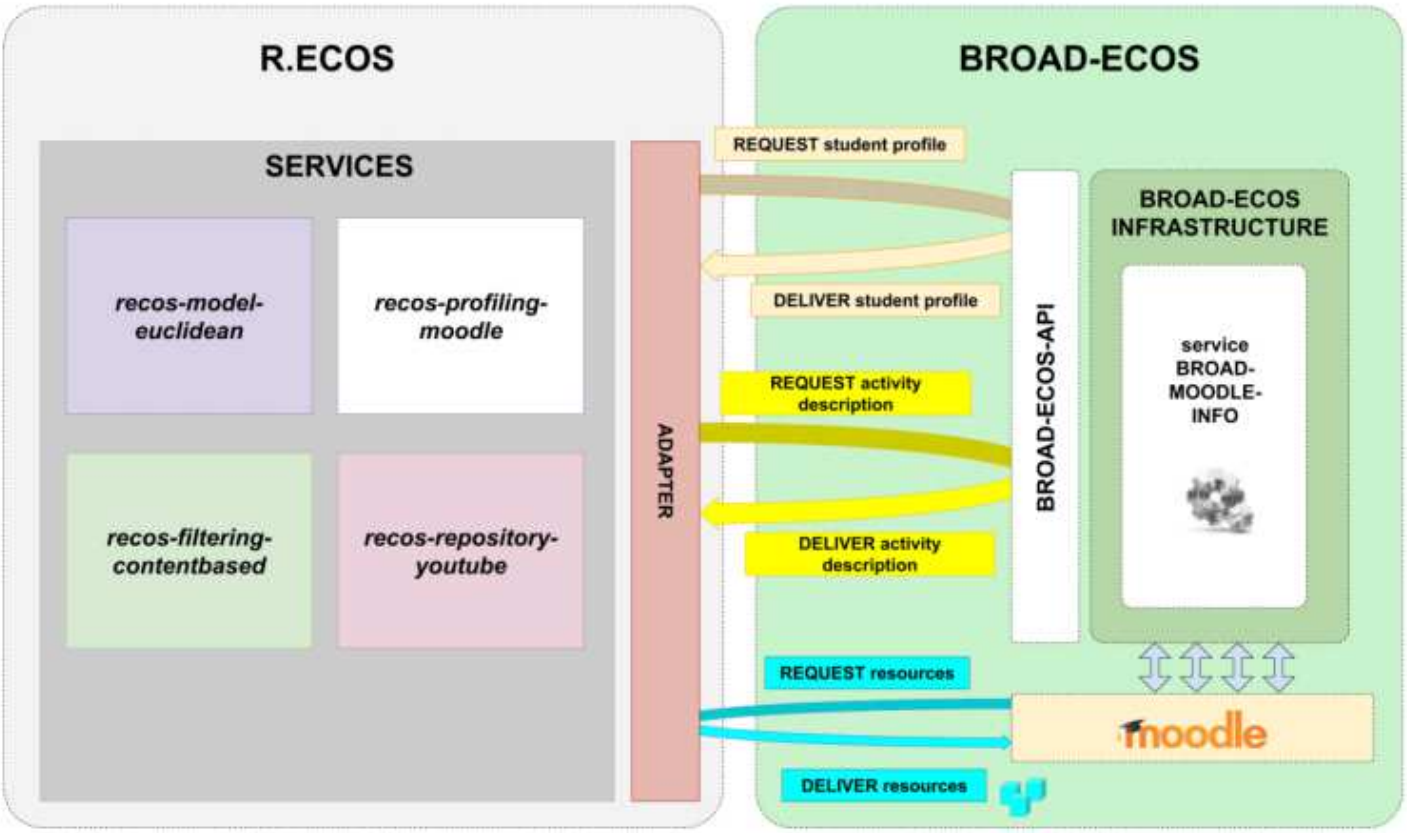

Figure 8

Integration between R.ECOS and BROAD-ECOS.

Figure 9

Resource recommendation with the evaluation device on Moodle.

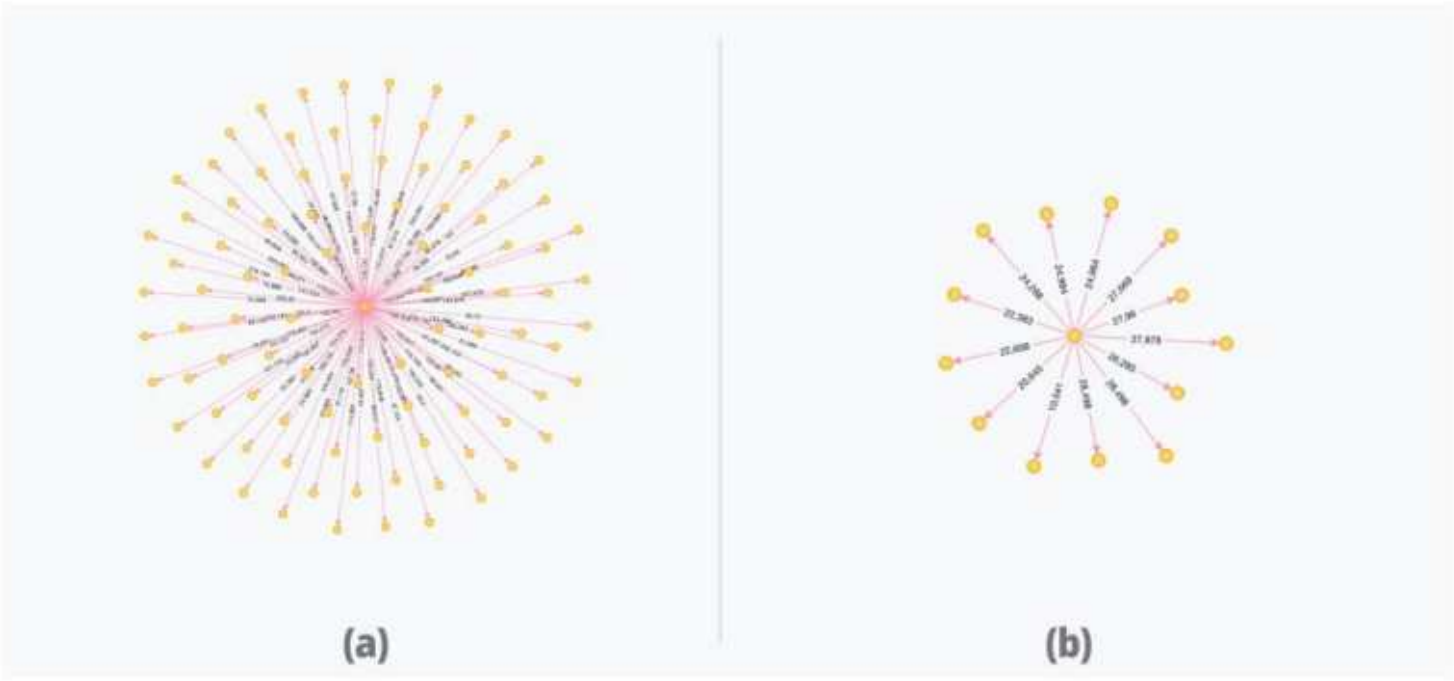

Figure 10

(a) The student with similarities calculated among all the other students. (b) The same student with the most relevant similarities to the group.

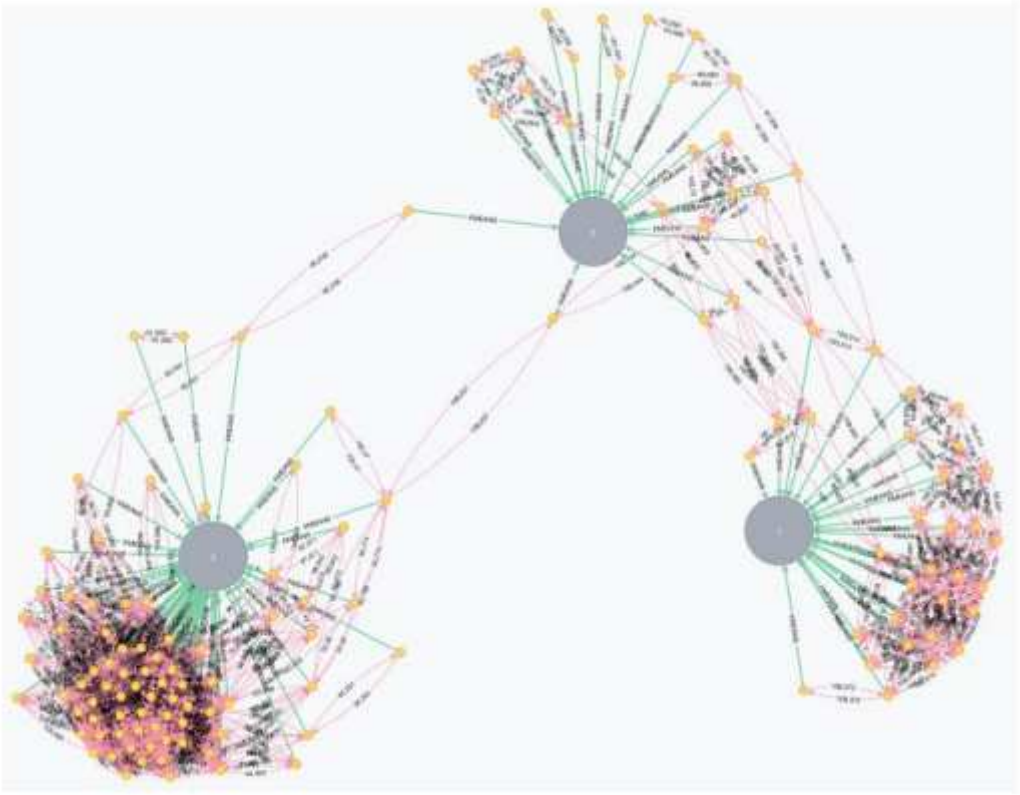

Figure 11

Graph representing groups of similar students.