UNIVERSIDAD DE CASTILLA-LA MANCHA

ESCUELA SUPERIOR DE INFORMÁTICA

BACHELOR IN COMPUTING ENGINEERING

Specialization in Computing

BACHELOR DISSERTATION

FENCE: a Fuzzy Sociotechnical Congruence Measurer

Antonio Manjavacas Lucas

July, 2020

UNIVERSIDAD DE CASTILLA-LA MANCHA
ESCUELA SUPERIOR DE INFORMÁTICA

Department of Technologies and Information Systems

Specialization in Computing

BACHELOR DISSERTATION

FENCE: a Fuzzy Sociotechnical Congruence Measurer

Author: Antonio Manjavacas Lucas

Supervisor: Aurora Vizcaíno Barceló

Co-supervisor: José Ángel Olivas Varela

July, 2020

Este texto ha sido preparado con la plantilla LaTeX de TFG para la UCLM publicada por Jesús Salido en GitHub[1] y Overleaf [2] como parte del curso *«LaTeX esencial para preparación de TFG, Tesis y otros documentos académicos»* impartido en la Escuela Superior de Informática de la Universidad de Castilla-La Mancha.

[1]https://github.com/JesusSalido/TFG_ESI_UCLM
[2]https://www.overleaf.com/latex/templates/plantilla-de-tfg-escuela-superior-de-informatica-uclm/phjgscmfqtsw

Tribunal:

Presidente: ______________________________

Vocal: ______________________________

Secretario: ______________________________

Fecha de defensa: ______________________________

Calificación: ______________________________

Presidente	Vocal	Secretario
Fdo.:	Fdo.:	Fdo.:

A Antonia, Eugenia,
Paco y Jesús

Abstract

The evolution of the software industry that has taken place in recent years has brought with it a considerable trend towards the decentralisation of projects and, therefore, of the work teams involved in their development. This software production model, known as Global Software Development (GSD), provides organisations with multiple benefits, such as a reduction in costs and time to market, innovation, process improvement, continuous development, and proximity to resources and stakeholders.

However, despite the many advantages associated with GSD, there are still many challenges associated with this software development model, most of which are related to collaboration and coordination difficulties owing to the interaction between users who are culturally, temporally, or geographically distant. These distances accentuate communication problems and pose difficulties that, if not properly addressed, can lead to significant drawbacks or even project failure.

Preventing communication problems between users or work teams is, therefore, of particular relevance in distributed environments. To achieve this task, a metric called Socio-Technical Congruence (STC) is proposed, whose objective is to measure the gap between organisations' coordination requirements and the coordination actions that are currently being performed.

This project addresses the design and development of FENCE, a tool with which to measure and improve organisations' STC levels, thus providing a new means to measure these levels by taking advantage of the benefits of fuzzy logic and making use of an expert system designed to provide solutions to existing communication problems.

Resumen

La evolución de la industria del software en los últimos años permite observar una considerable tendencia a la descentralización de los proyectos y, por tanto, de los equipos de trabajo involucrados en su desarrollo. Este modelo de producción de software, conocido como Desarrollo Global del Software (GSD), supone múltiples beneficios para las organizaciones, tales como reducción de costes y tiempos de desarrollo, innovación y mejora de procesos, desarrollo continuo, así como proximidad a recursos y *stakeholders*.

Sin embargo, a pesar de las múltiples ventajas asociadas al GSD, son aún muchos los retos asociados a este modelo de desarrollo software. La mayoría de ellos se encuentran relacionados con las dificultades de colaboración y coordinación debido a la interacción entre usuarios cultural, temporal y geográficamente distantes. Estas distancias acentúan los problemas de comunicación y suponen dificultes que, en caso de no ser debidamente abordadas, pueden conducir a notables inconvenientes o incluso al fracaso de los proyectos.

De esta forma, la anticipación a los problemas de comunicación entre usuarios o equipos de trabajo se plantea como una acción de especial relevancia en entornos distribuidos. Para lograr este cometido, existe una métrica denominada congruencia sociotécnica (STC) destinada a medir la diferencia entre los requisitos de coordinación de una organización y las medidas de coordinación que actualmente están siendo llevadas a cabo.

Este proyecto aborda el diseño y desarrollo de FENCE, una herramienta orientada a la medición y mejora de los niveles de STC de las organizaciones, ofreciendo una forma novedosa de medirla aprovechando los beneficios de la lógica borrosa y haciendo uso de un sistema experto destinado a ofrecer soluciones a los problemas de comunicación existentes.

AGRADECIMIENTOS

A mis padres, Antonio y Criptana, por enseñarme desde la humildad y el trabajo a valorar el esfuerzo por hacer bien las cosas. Es infinito mi agradecimiento a vuestro sacrificio y entrega por hacer realidad este sueño. Este trabajo es tanto mío como vuestro.

A mi hermano Jesús, testigo de mi esfuerzo y empeño, ya sea desde la cercanía como desde la distancia.

A mis amigos, siempre presentes, tanto en mejores como en peores momentos, por ayudarme a mantener la cordura y hacerme tan feliz.

A Miguel y Rubén, porque un camino como este no debería recorrerse solo. Gracias por aguantarme con paciencia y humor, sois una parte fundamental de esta etapa.

A mis compañeros del grupo Alarcos. Gracias a Javier y Julio, por todo lo que me han enseñado, por ayudarme a dar mis primeros pasos en la investigación, así como por los buenos momentos vividos en el laboratorio. Gracias a Mario Piattini, por su confianza y apuesta en mí, por introducirme de lleno en este mundo tan sacrificado pero apasionante. Gracias, en general, a todos los que han formado parte de esta gran experiencia. No tengo dudas de que el esfuerzo que ha supuesto compaginar estudio y trabajo ha merecido la pena.

A José Ángel Olivas, una verdadera inspiración y ejemplo a seguir. Gracias por toda la ayuda prestada, por enseñar desde la experiencia y, ante todo, por tu buen humor.

A Aurora Vizcaíno, por confiar en mí, por enseñarme tanto, por tu cercanía y tus buenos consejos. Gracias por guiarme con templanza durante todos estos años, es mucho lo que te debo.

Antonio Manjavacas Lucas
Campo de Criptana, 3 de Junio de 2020

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

LIST OF ALGORITHMS

CHAPTER 1

INTRODUCTION

This project presents FENCE, a tool with which to measure and improve the socio-technical congruence levels of organisations by means of fuzzy logic. In this first chapter we shall specifically address the theoretical foundations set out in literature on which the justification for this project rests. Finally, once the context on which FENCE is based has been presented, the structure of this document will be detailed, providing a brief explanation of the aspects that will be dealt in each of the following chapters.

1.1. GLOBAL SOFTWARE DEVELOPMENT

When looking at how the software industry has progressed in recent years, a major trend towards decentralisation and the adoption of distributed development models will be observed. This new approach to software production, known as Global Software Development (GSD), is based on the development of projects on distant sites, usually located in different countries around the world. GSD consequently provides organisations with multiple advantages, such as cost savings, time to market reduction, access to larger multi-skilled workforces or proximity to resources and stakeholders [1, 10].

These are the best-known benefits for organisations, but there are also other implicit advantages, as GSD also implies organisational benefits associated with greater innovation and the adoption of best practices, improved task modularisation, increased team autonomy, better documentation, or more accurate and traceable asynchronous communication [1].

If we look beyond organisations, then the greatest expression of GSD is currently taking place in the open-source community [38] and in any crowdsourcing environment, in which hundreds or even thousands of people remotely collaborate in software development.

There are not, however, only advantages, since GSD also implies multiple challenges associated with temporal, geographical and socio-cultural distances [42], given its distributed and multicultural essence. Some of these challenges are related to communication and collaboration among sites, a lack of awareness of the organisational structure or project state, knowledge dissemination difficulties, requirements elicitation, trust improvement, or the need to adapt agile methodologies to distributed environments [29, 30, 37, 45].

Literature generally places considerable emphasis on the importance of communication and coordination in GSD, as they are the key aspects as regards ensuring the success of projects on which multiple virtual teams have to distantly collaborate together [29, 61]. Being able to know and evaluate the state of an organisation in terms of its coordination and communication will consequently result in a significant advantage when it comes to mitigating risks and preventing major problems, hence the importance of tools and means that allow this task to be undertaken.

1.2. SOCIOTECHNICAL CONGRUENCE

In 1968, Melvin Conway was responsible for "Conway's Law" [11, p. 31], which states that "*organizations which design systems [...] are constrained to produce designs which are copies of the communication structures of these organizations*". This principle is still in force [15, 22], and its interpretation suggests that communication has a significant influence on the development of any physical system. Moreover, software systems can be considered a special case that also applies to this law, since in this case the product structure corresponds to the software architecture, and the organisational structure with the organisational chart [22].

As explained in [47], in 2006, a new term closely related to Conway's law and known as Socio-Technical Congruence (hereinafter STC) was introduced by Cataldo and his colleagues [7]. It was initially defined as "*a technique to measure task dependencies among people, and the 'fit' between these task dependencies and the coordination activities performed by individuals*" [7, p. 353]. This has been the basis of the many other definitions of STC that have emerged since the initial one [5, 6, 23, 25, 44, 56], always emphasising that STC describes the degree of matching between an organisation's coordination requirements and its current coordination activities.

The main properties of STC are those shown in [44], in which it is defined as a representation of organisational status in terms of coordination, also emphasizing its multidimensionality and interpretability from the different levels of the organisation. The authors also highlight other aspects, such as the fact that STC is a dynamic and descriptive metric or that it involves advantages and disadvantages that must be balanced in order for it to be truly useful for the organisation as a whole.

Some of the other proven effects of STC are its impact on software quality [5–7, 12, 25, 31, 48] and development time [6, 7, 25]. Its relevance for GSD projects has also been studied [5–7, 12], including open source environments [4, 50–52, 56] in which work is still pending. As can be seen, the range of influence of STC covers all types of projects and dimensions, thus proving the relevance of this metric and its proper measurement in real contexts.

If we focus on how to measure STC, there is a problem to be considered, since there is no standard measurement method, but rather multiple proposals based on the same idea with visible differences. As explained in [47], the most generic proposal by Cataldo et al. [6, 7] is usually that taken as a reference and to which adaptations and improvements have been made. Despite the fact that multiple proposals with which to measure STC have appeared in literature [6, 7, 27, 40, 55, 56], in this project we shall take as a reference that provided by Kwan et al. [24, 25], given its adaptability, scalability and precision when compared to other alternatives. An in-depth study of its implementation will be carried out in chapter 4.

Finally, another relevant statement made in [47] is that no specific ranges have been defined in literature that will make it possible to state if the STC levels are good or bad. In this respect, STC is flexible and, its measurement relative to the circumstances of projects and organisations.

1.3. KNOWLEDGE-BASED SYSTEMS

Artificial intelligence (AI) can be defined as a branch of information science that deals with the implantation of a restricted but defined part of human intelligence [39] in computers. There have been many definitions of AI since its origin, which can be grouped according to 4 perspectives, as shown in Table 1.1.

AI is a very broad field, and many disciplines are directly or indirectly involved in its definition, from mathematics and programming to psychology, philosophy, or linguistics. This interaction with multiple disciplines has given rise to a great variety of branches within AI, of which we shall focus on that known as Knowledge Engineering.

Table 1.1: Definitions of AI

	Human perspective	**Rational perspective**
Mental processes and reasoning	Systems that think like humans	Systems that think rationally
Behaviour and performance	Systems that act like humans	Systems that act rationally

Knowledge Engineering is a discipline of artificial intelligence focused on the analysis and proposal of methods so as to acquire, represent, store and use knowledge in order to emulate the intelligent reasoning capacities of human beings. One of the main tasks in Knowledge Engineering is to provide systems with the ability to reason on the basis of how this is done by human beings, resulting in Expert Systems.

An Expert System is a computer programme that attempts to emulate the behaviour of an expert in a specific domain when confronted with a certain problem, along with the procedure followed to attain a solution [32, 57]. Knowledge engineering, therefore, addresses the specification, analysis and development of expert systems.

The origin of expert systems goes back to the end of the 1960s and beginning of the 1970s, during which time systems such as DENDRAL [26], MYCIN [46] or PROSPECTOR [13] were some of their most representative examples. These examples make it possible to appreciate the many areas of application of expert systems, of which medicine, teaching, consulting, mathematics, decision making, training, control or planning are just some.

The main objective of expert systems is consequently to help people solve problems, regardless of their experience as regards mastering the problem, in addition to safeguarding knowledge in order to facilitate further training and learning in its domain.

With regard to how these systems perform their tasks, expert systems have different ways of emulating human reasoning, with rule-based reasoning being one of the most widespread. These rule-based systems make use of logical inference (usually supported by fuzzy logic [20, 36]) to derive conclusions from a knowledge base and a set of input perceptions. As can be assumed, the use of logical inference in reasoning will involve a prior formalization of expert knowledge by using a logical representation.

As explained in [33], the hybrid combination of fuzzy logic and expert systems is simple and natural, as both have features that complement each other. This has, therefore, led to the appearance of fuzzy expert systems, which, as will be shown in the following section, make it possible to reflect reality in a more human and precise way when compared to conventional rule systems.

1.4. FUZZY LOGIC

Fuzzy logic has its origins in the first definition of a fuzzy set provided by Lofti A. Zadeh in 1965. As he explains in [59, p. 339]: "*a fuzzy set A in [a space of points] X is characterized by a membership function $f_A(x)$ which associates with each point in X a real number in the interval [0, 1], with the value of $f_A(x)$ at x representing the 'grade of membership' of x in A*". It is, therefore, possible to extend traditional set theory to a more extensive domain in which the membership functions of sets are no longer bivalued (0 or 1), but represent a degree of membership between 0 and 1, as shown in Equation 1.1:

$$f_A : X \longrightarrow [0, 1] \tag{1.1}$$

where $f_A(x) = 1$ if x is totally in A, $f_A(x) = 0$ if x is not in A, and $0 < f_A(x) < 1$ if x is partially in A.

On this basis, one the main contributions of fuzzy logic is the ability to represent knowledge by means of qualitative quantifiers. This makes it possible to bring the inferences closer to a more humane approach, in which linguistic labels such as "high" or "cold" do not express precise values but are ambiguous and relative to the context in which they are being applied [3, 35, 60].

Furthermore, after its inception, the applications of fuzzy logic underwent an overwhelming growth, and began to appear in countless fields such as fuzzy control systems, fuzzy databases, expert systems, optimisation problems, and any other domain in which approximate reasoning is required.

In the field of fuzzy control, systems are based on knowledge inferred by an expert or learned autonomously in order to make decisions by means of inference. The inference engines of expert systems that make use of approximate reasoning are similarly able to exploit the advantages of fuzzy logic in order to provide reasoning and solutions that are closer to the human perspective.

With regard to how fuzzy inference systems work, the process carried out is as follows [58]:

- First, the fuzzification of the input values into fuzzy membership functions is performed.
- The system then executes those applicable rules, resulting in output values.
- Finally, the defuzzification process is carried out on the outputs in order to obtain the resulting "crisp" (conventional set) values.

As will be noted, fuzzy sets provide smoother transitions between the limits of a crisp set. It is, in turn, possible to employ many kinds of fuzzy membership functions (triangular, trapezoidal, gaussian, sigmoidal, etc. [21], and choosing between them will, therefore, depend on the problem to be solved and the decisions made on the basis of experience.

1.5. DOCUMENT STRUCTURE

Having explained the motivation of the project and the state of the art, the phases undertaken to develop FENCE, a tool for the measurement and improvement of STC levels within an organisation by using fuzzy logic, will subsequently be described in the various chapters of this document. A detailed description of these chapters is provided below:

- **Chapter 1**. The present chapter addresses the motivation of the project, along with the state of the art as regards global software development, socio-technical congruence, knowledge-based systems and fuzzy logic. It has served as an introduction to the work to that will take place and to the context of the problem being confronted.
- **Chapter 2**. The second chapter provides a detailed explanation of the project objectives, both general and specific.
- **Chapter 3**. This chapter concentrates on the initial planning of the project, which involves both the justification of the working methodology adopted and the technological framework used.
- **Chapter 4**. This chapter shows details of the development of FENCE by describing its multiple phases and iterations. It covers both the planning and definition of the project and the development of its different functional modules.
- **Chapter 5**. This final chapter presents the assessments and conclusions of the project, including future work, lessons learned, and the justification of the competences acquired after its development.

For any enquiry, all of the code referenced in this document is available in the following GitHub repository: https://github.com/manjavacas/fence.

CHAPTER 2

OBJECTIVES

This chapter details the objectives to be addressed in this project. First, the main objective of the project will be presented, after which the sub-objectives into which the project is divided will be explained in detail.

2.1. MAIN OBJECTIVE

The main objective to be addressed in this project is the development of a tool for the measurement and improvement of socio-technical congruence within organisations.

As previously explained, STC is a metric that provides a perspective of the state of the organisation from the point of view of its coordination and collaboration, signifying that the greater that the STC is, the better these organisational attributes will be, and that a generally improved performance is, therefore, expected (fewer costs, better quality, less time spent, etc.).

The tool will, therefore, provide information on the state of the organisation in terms of STC, while simultaneously providing recommendations in order to improve coordination and communication between employees. The objective of the tool is consequently that of improving individual, and thus, overall STC levels within the organisation.

STC will be measured by employing an adaptation of the algorithms presented in literature in an attempt to improve and adapt them to the context of global software development and to improve them by applying fuzzy logic.

Finally, the functionality of this tool, whose objective is to provide recommendations, will also make use of a fuzzy inference system based on expert knowledge and built following the necessary knowledge engineering practices.

2.2. SPECIFIC OBJECTIVES

Having detailed the main objective of the project, we shall now subdivide it into a set of specific objectives. These objectives will be directly mapped onto the features of the tool to be developed during the project.

The specific objectives and their rationale are detailed as follows:

1. The development of a dashboard that will allow the manipulation of the different resources needed to measure STC: employees, teams, projects, tasks, task assignments, task dependencies and communications.
2. The implementation of the algorithms required to measure STC and adapted to Global Software Development contexts, along with an interface that will allow the user to perform this calculation and visualize it at user, team and project levels.

3. Improvements to the conventional algorithms presented in literature by using fuzzy logic. The objective will be to attempt to adjust the weights of the matrices required in order to measure STC by means of a fuzzy control system. The objective is, therefore, to obtain more realistic measurements adapted to the communication challenges of global software development projects.
4. The implementation of a module oriented towards the visualization of the STC measurement history for employees, teams and projects in order to evaluate its temporal evolution.
5. The creation of an adaptable recommendation system based on expert knowledge so as to improve the STC levels of the organisation. A knowledge engineering approach will be adopted in order to build the system, and its feasibility and features will be studied prior to its development.

CHAPTER 3

METHODOLOGY

This chapter will address the definition and justification of the working methodology used, as well as the planification of the project. The technological resources and tools employed will also be exposed in detail.

3.1. AGILE METHODOLOGIES

The Agile Manifesto [2] was born in 2001 as a set of four values and twelve principles associated with the search for improvements in software development over conventional methodologies. In this manifesto, the assessments of expert software practitioners were declared, stating the importance of:

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

Agile methodologies promote the continuous delivery of software in short cycles, fostering involvement, training and adjustment to stakeholders needs [34]. These methodologies mean, in turn, greater flexibility, increased productivity and, consequently, cost reductions during changing projects, where the requirements can differ throughout their development.

Nowadays, the benefits associated with agile methodologies have made them widely used by practitioners, enhancing the perceived and internal quality of software development as well as promoting their profitability.

3.2. OPENUP

Being aware of the advantages associated with the use of agile methodologies in software engineering and given the iterative and incremental nature of this project, the methodology chosen for its development was OpenUP [28].

OpenUp is a minimum and sufficient methodology, which means that it only considers the fundamental contents of software development, leaving aside aspects such as the management of large teams, technology-specific guidance or contractual situations. In spite of its simplicity, OpenUP covers in a complete and agile way the whole development process of a software project, being completely flexible to the nature of the project in which it is employed.

The main principles of OpenUP offer a direct mapping with the ones expressed in the agile manifesto and try to represent the working model to follow by using this methodology:

- **Collaborate to align interest and share understanding**, advocating for coordination and mutual understanding among stakeholders.

- **Balance competing priorities to maximize stakeholder value**, trying to maximize profits while conforming to project constraints.
- **Focus on the architecture early to minimize risks and organize development**.
- **Evolve to continuously obtain feedback and improve**, in order to have continuous communication with stakeholders and demonstrate incremental value to them.

3.3. PHASES AND PROJECT PLANNING

The use of agile, iterative, and incremental methodologies such as OpenUP facilitates the coordination and development of projects based on multiple modules that, once developed, add value to the desired final product.

As shown in Figure 3.1, the organisation of work followed by OpenUP distinguishes between three different perspectives based on personal, team and stakeholder levels:

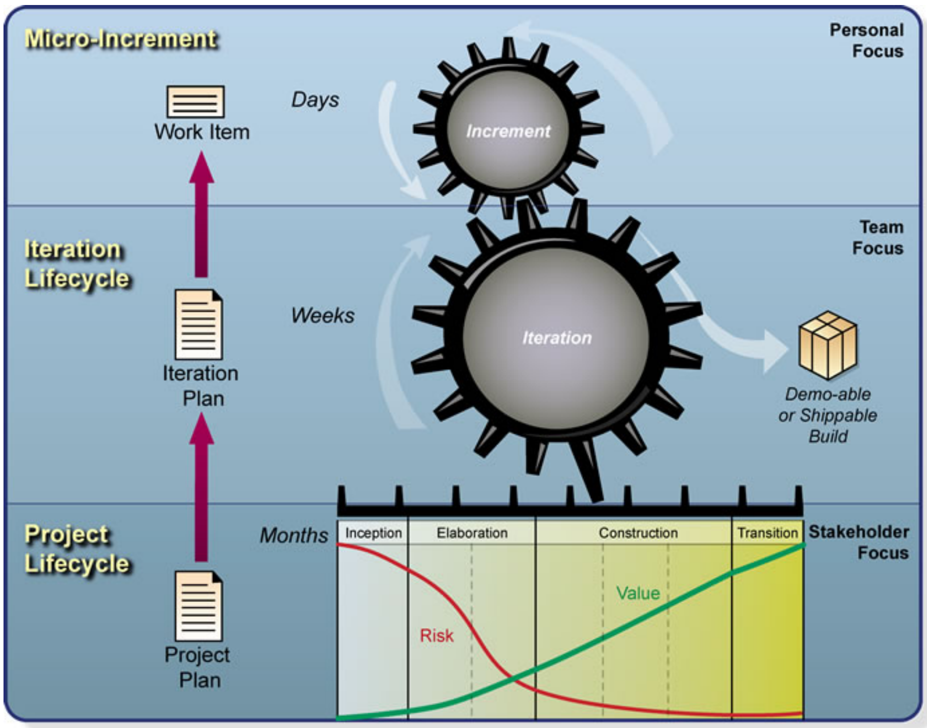


Figure 3.1: OpenUP lifecycle layers (source: https://www.ida.liu.se/~TDDD77/openup/index.htm)

- The personal effort of an OpenUP project is defined as a **micro-increment**, commonly measured in hours or days.
- From the team's perspective, an **iteration lifecycle** reflects how micro-increments are applied to obtain stable and cohesive builds of the system being produced.
- Focusing on how an overview of the project is guaranteed for the stakeholders, OpenUP structures the **project lifecycle** into four phases: *Inception*, *Elaboration*, *Construction*, and *Transition*.

Thus, each of the phases of the OpenUP lifecycle are defined as follows:

- **Inception phase**: in this phase, it is a question of understanding what is intended to be produced and what are the objectives and limitations of the system to be developed, identifying the stakeholders, and detailing their success criteria.
- **Elaboration phase**: it involves the procurement of a more detailed understanding of the system requirements; design, implementation, validation and establishment of the architecture

baseline, providing a skeleton of the system structure; mitigation of essential risks and project planning in terms of time and costs.
- **Construction phase**: iterative development of the desired product, until a tested result is achieved and ready to be offered to users. The goal pursued during this phase is to minimize costs through resource optimization and parallelization of independent tasks.
- **Transition phase**: this last phase involves validating user expectancies, obtaining stakeholders approval, and seeking to improve future projects based on well-documented lessons learned.

These phases, applied to the particular case of this project, are presented in the following way:

3.3.1. Inception phase

In this phase, the justification of the project was carried out, identifying its scope and the objectives to be pursued. Its feasibility in terms of risks, time and estimated costs was also assessed. Roles were assigned and, at the same time, an attempt was made at identifying the key functionalities of the system in accordance with the specifications defined with the identified stakeholders.

At this stage, the first meetings were held to establish the key functionalities of the system, to understand the competencies addressed and to provide an insight into how to proceed in the months ahead.

3.3.2. Elaboration phase

Once an overview of the project had been established, a more detailed understanding of the objectives pursued was sought in this phase. To perform this task, a series of interviews with the stakeholders were planned and conducted, allowing to know the essential features that the system should meet. On this basis, the elicitation and formalised documentation of the system's requirements was carried out, allowing to proceed with the analysis and design of the functional modules that would compose the system.

Other tasks inherent to this stage were addressed, such as the choice of technological resources to be used, the development process employed, the definition of the system architecture baseline or the acquisition of domain-specific knowledge. Actors and use cases were also formally defined.

Finally, considering the set of formally stated requirements as well as the resulting skeleton of the system to be developed, a project planning consisting of time and cost estimation was elaborated. Based on the identified functionalities, the iterations that constitute the lifecycle were organized according to the priorities expressed by the stakeholders.

3.3.3. Construction phase

In this stage, the implementation of the different functionalities of the system was carried out. This implementation was undertaken in an orderly manner based on the preferences agreed upon with the stakeholders in the previous phase.

Presented in the order in which they were addressed, the following system functionalities were developed:

- **Module 1**: API skeleton.
- **Module 2**: Main user interface.
- **Module 3**: STC measurement.
- **Module 4**: Data visualization.
- **Module 5**: Recommendation system.
- **Module 6**: Settings and preferences.

The implementation of each module was done in an iterative way, trying to agree with the stakeholders (the project tutors) any kind of modification in the functionality or the interface considered during this development stage. As would be reflected in the planning, each functional module was associated with an iteration during the construction phase of the project.

3.3.4. Transition phase

Finally, the transition stage was dedicated to the documentation, testing and deployment of the system, confirming compliance with the requirements with the stakeholders and elaborating the final project report.

The review of the project by the tutors as well as the preparation of the project presentation were also part of this stage.

3.4. ROLES

In this subsection, the set of basic OpenUP roles will be defined as well as the assignment of those roles for the concrete case of this project.

- **Analyst**: is the person in charge of identifying and understanding the problems and opportunities of the project, by knowing and interpreting the requirements expressed by the customer and end-users.
- **Architect**: designs and documents the system architecture, being responsible for technical decisions on the overall implementation of the system.
- **Developer**: each person in this role is responsible for the implementation of a set of parts of the system, adjusting it to the architecture and using the necessary technologies for the development.
- **Project manager**: is the person in charge of planning the project, fulfilling the objectives as well as communicating and coordinating with the stakeholders.
- **Tester**: person in charge of the identification, definition, implementation and execution of the tests over the system.
- **Stakeholders**: people that may be directly or indirectly affected by project realization. Normally a stakeholder is a person whose needs will be met after the project is completed, like end-users o customers.

The general assignment of the OpenUP roles for this project is detailed in Table 3.1.

Table 3.1: Roles assignment

Role	Person	Charge
Analyst	Antonio Manjavacas Lucas	Grade student
Architect		
Developer		
Project manager		
Tester		
Stakeholder	Aurora Vizcaíno Barceló	Project supervisor and professor
Stakeholder	José Ángel Olivas Varela	Project supervisor and professor

On the one hand, Aurora Vizcaíno was in charge of monitoring and providing feedback on the project: due to her experience in the field of GSD, she elicited the key requirements of the system, served as a guide throughout its planning and development and acted as a contact person for any queries during the project.

On the other hand, José Ángel Olivas was responsible for recommending and guiding the implementation of the modules related to the competencies of the Computing intensification, such as the underlying fuzzy logic mechanisms used to describe the communications and coordination between users by the expert system.

Finally, the tasks of analysis, design, development, testing, and documentation were carried out by Antonio Manjavacas, the grade student responsible for the project.

3.5. RESOURCES

The following sub-sections will detail the hardware and software resources used in the development of this project.

3.5.1. Hardware resources

The hardware resources used are shown below:

- HP Pavilion x360 Convertible 14-cd0xxx equipped with:
  - Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz 1.99GHz, 64 bits
  - 12 GB RAM
  - Integrated graphics Intel(R) UHD Graphics 620
  - NVIDIA GeForce MX130

3.5.2. Software resources

Likewise, these were the technologies, environments and programming languages used throughout the project:

- Java 1.8. General purpose, high level, object-oriented programming language. It is one of the most widely used programming languages by the developers community nowadays. Java stands out for its ease of learning and use, as well as its security and portability to any device capable of running Java Virtual Machine (JVM).

- Apache Maven. Tool for the management and automation of Java projects, facilitating dependencies management and software building. It is maintained by the Apache Software Foundation and is widely used by the Java community.

- Git. Distributed version control system based on repositories. Facilitates version, branch, and data integrity management in software projects.

- GitHub. Hosting system for version control using Git. It facilitates the management of repositories and projects, offering multiple facilities in their creation and maintenance.

- Spring Framework. Framework for the development of Java applications developed by Pivotal Software. It provides easy integration with Maven and other services to achieve the development of web applications.

- MongoDB. Database management system, NoSQL and document based. Chosen for its easy use and optimization for the treatment of large data sets. Provides an easy integration with the rest of the tools employed, such as Spring framework.

- MongoDB Atlas. MongoDB cloud database hosting service. It offers limited free storage as well as an easy connection and usage by applications via URI.

- Eclipse. Integrated Development Environment (IDE) widely used in Java development, among other languages. It includes utility plugins that facilitate integration with Git (EGit) or Maven (m2Eclipse).
- Visual Studio Code. Free and open source code editor, which offers a large number of plugins and language support.
- Python. Interpreted, multi-paradigm, dynamic and multi-platform programming language. It stands out for its readability and usability.
- jFuzzyLogic. Java library for handling and working with fuzzy logic [8, 9]. It is based on the standard FCL (Fuzzy Control Language) and is freely distributed.
- FCL. Fuzzy Control Language published by the International Electrotechnical Commission (IEC). It is specified in the IEC 61131-7 document and allows the standardization of fuzzy control systems.
- HTML 5. Markup language used in web development. It is a standard in charge of the World Wide Web Consortium (W3C) and is currently in its version 5.
- CSS. These cascading style sheets constitute a graphic design language used in conjunction with markup languages such as HTML. Used to enhance the appearance of user interfaces and customize their appearance.
- Bootstrap. Open source multi-platform library for frontend design. It offers design templates based on HTML, CSS, and JavaScript.
- JavaScript. Interpreted, object-oriented, prototype-based, imperative, weakly typed and dynamic programming language. Widely used in client-side web development.
- Chart.js. Open source JavaScript library for creating graphics and animations in HTML5.
- jQuery. JavaScript library aimed to simplify interaction with HTML and make client-side AJAX (Asynchronous JavaScript And XML) requests in web applications.
- Postman. A tool for the creation of API requests in a simple and fast way. It eases the testing of web applications and allows a high customization of the requests.
- Heroku. Cloud Computing Platform as a Service (PaaS). Enables deployment of cloud applications and offers limited free hosting.
- Ubuntu. Free and open-source Linux distribution distributed by Canonical Ltd. Used in the deployment procedure and part of the application development.
- Windows 10. Operating system developed and distributed by Microsoft. Used in most of the application development.
- Zotero. Free and open bibliographic reference manager. It is multi-platform and offers a browser extension that facilitates the management and collection of references.

- LaTeX. Markup language oriented to the elaboration and layout of high-quality documents. Based on TeX, initially developed by Donald Knuth. The template available in [43] has been used to elaborate this document.

- TeXstudio. Integrated writing IDE for creating LaTeXdocuments. TeXstudio is open source and available for all major operating systems.

- Microsoft Word. WYSIWYG text editor distributed by Microsoft in its Microsoft Office package. Used for the elaboration of the draft memory and basic documental tasks during the project.

- Microsoft OneDrive. Cloud data storage system offered by Microsoft. Used for the storage of documents, backups, code, and diagrams.

- Diagrams.net. Previously known as Draw.io, it is a tool for creating diagrams and charts. It is open source and offers online and desktop support.

Finally, a summary of the software resources used is shown in Figure 3.2.

Figure 3.2: Software resources used in this project

CHAPTER 4

RESULTS

In accordance with the OpenUP lifecycle described in chapter 3, the following subsections will detail the development phases of the first version of FENCE, emphasizing the results of each phase and thoroughly explaining the decisions made in the design and development of the tool.

4.1. INCEPTION PHASE

In this first phase, initial meetings with stakeholders took place, where a first contact was established and from which the project objectives would begin to be defined. The idea of the preliminary project was consolidated, and the functionalities and design of FENCE began to be roughly sketched out.

Therefore, this meetings with the stakeholders, Aurora Vizcaíno and José Ángel Olivas, focused on defining, on the one hand, the basic functionalities of the tool (STC measurement, dashboard features, settings and preferences) and, on the other hand, the features that would make FENCE an innovative tool (use of fuzzy control system to improve STC measurement or a recommending system aimed to enhance STC levels). These features were conveniently formalized as requirements, as we will see in the elaboration phase.

As previously mentioned, the initiation phase in OpenUP involves the definition of the roles of the different stakeholders. In this case, it was decided that Aurora Vizcaíno would supervise the development of the application in general terms, while José Ángel Olivas would monitor the part of the system related to the intensification competencies, such as the implementation of the expert system or the introduction of fuzzy logic into the system.

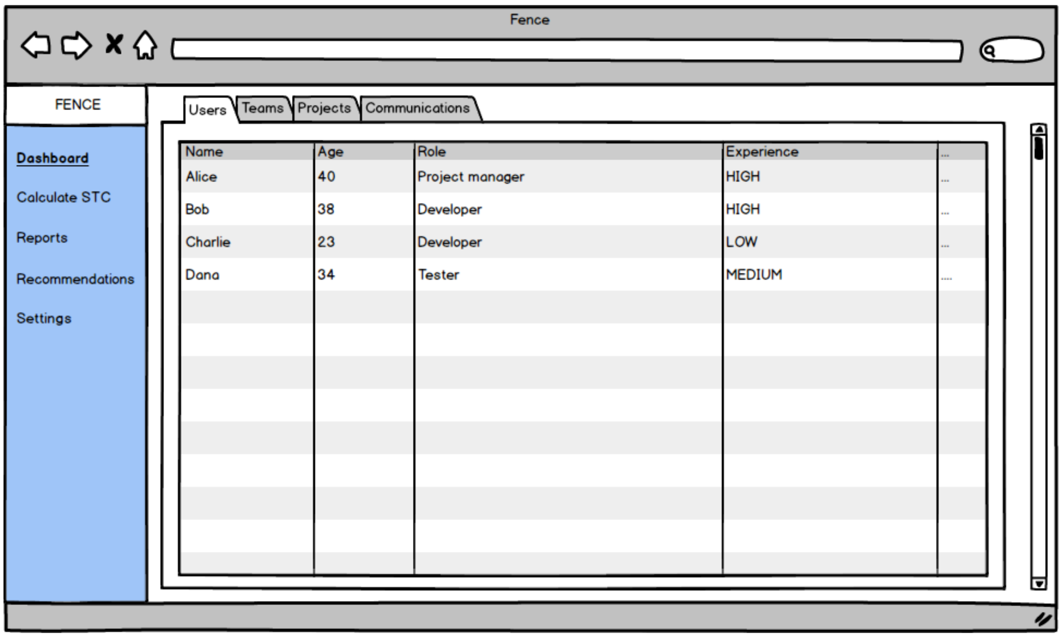
Moreover, from a technical point of view, it was also agreed that FENCE would be a web application capable of offering services locally and, if possible, online. Furthermore, ideas also began to be sketched out on what the web application would graphically look like, as shown in the sketches represented in Figure 4.1 and Figure 4.2.

With clear objectives and an approximate and viable idea of the project to be undertaken, we proceeded to the elaboration phase, where these objectives and requirements would be formalized thus establishing the foundations of FENCE.

4.2. ELABORATION PHASE

Initially, the objectives pursued in this phase are the formalization of the system requirements, the decision on the architecture to be used, as well as its design and data model. On this basis, it would be possible to establish a project planning, as well as a cost estimation adjusted to the development time and resources used.

Moreover, some key learning aspects were carried out at this stage, such as training in the Spring framework or MongoDB. Thus, further research in these technologies in the early project stages



Name	Age	Role	Experience	...
Alice	40	Project manager	HIGH	...
Bob	38	Developer	HIGH	...
Charlie	23	Developer	LOW	...
Dana	34	Tester	MEDIUM	....

Figure 4.1: Dashboard mockup

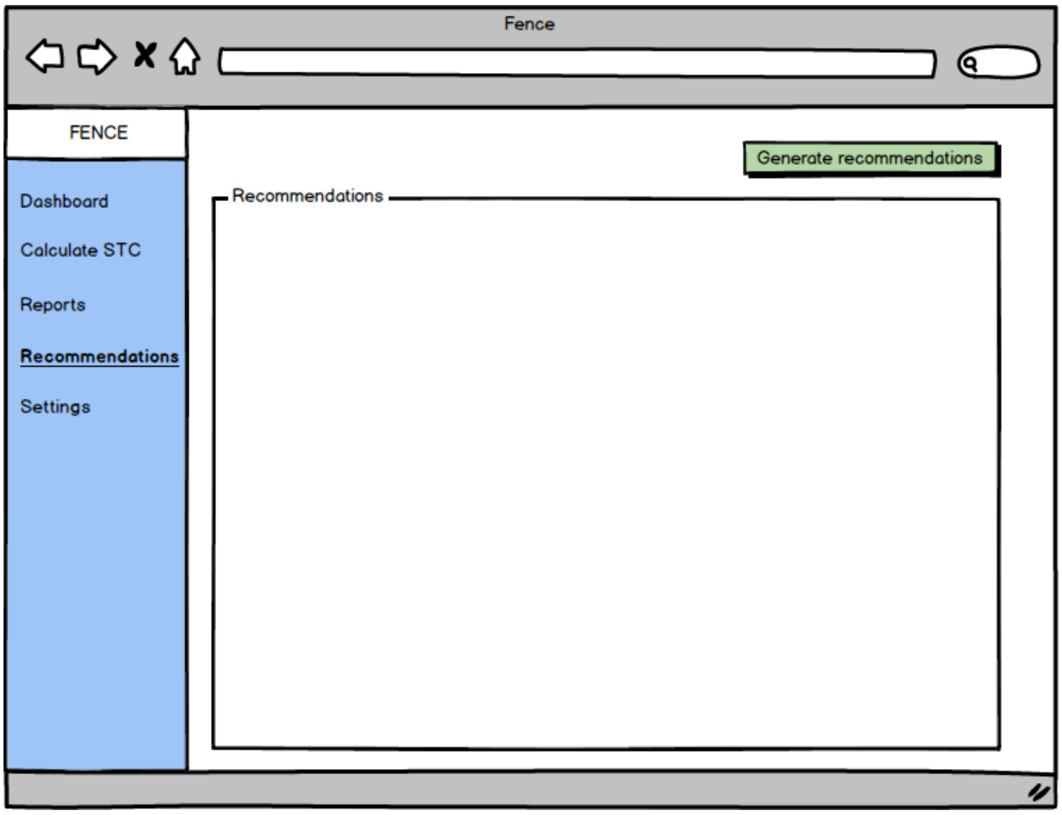


Figure 4.2: Recommendations window mockup

would facilitate future work and ensure more consistent progress.

4.2.1. Requirements

The requirements elicitation is the process in which the description of the properties that the system to be developed must meet is performed. These features are expressed by the client, while the interviewer is in charge of collecting the necessary information and expressing it in a precise and unambiguous document.

In the case of this project, the process of requirements elicitation was carried out following the structured interview method, as detailed in [54]. These interviews are divided in three phases: preparation, realization and analysis, and they are a close and direct way to directly understand interests of the stakeholders. Therefore, the interview was approached as follows:

- First the questions to be asked were decided, a structured script was drawn up with the questions to be asked and the order to be followed.
- Subsequently, a first meeting was held with the main stakeholder, Aurora Vizcaíno, where the functional and non-functional requirements of the tool were broadly agreed upon.
- Regular follow-up meetings would be held throughout the project, where these requirements would be not only evaluated but also adjusted to suggested changes and improvements.

The initial requirements were expressed in precise and unambiguous language, as will be shown in the following subsections. Thus, after their validation by the client, the use cases and modelling of FENCE would proceed.

FUNCTIONAL REQUIREMENTS

Functional requirements represent which functionalities should be covered by the system to be developed, considering inputs and outputs as well as the behaviour of the software. After the interviews with stakeholders, these functional requirements were agreed upon and duly expressed, giving rise to the set of functionalities represented in Table 4.1.

NON FUNCTIONAL REQUIREMENTS

Non-functional requirements are complemented by functional requirements to fully define the expected characteristics of the system. These requirements focus on representing quality and technical features as well as conditions and constraints expressed by the customer. Thus, the non-functional requirements for this project are shown in the Table 4.2.

4.2.2. Use cases and modules

Once the system requirements had been compiled, the definition of the use cases was undertaken. The use case model is shown in Figure 4.3, which represents the activities carried out by the entities (actors) according to the expected system functionalities.

Once the use cases were identified, they were grouped into different functional modules in order to organize the development of the subsystems that compose the application. These are listed in Table 4.3, together with their definitions and priority levels.

The decomposition of the application in functional modules would facilitate the way to proceed in the construction phase of the life cycle, performing micro-increments that could be tested and evaluated by the stakeholders. This modular division would also facilitate the planning and estimation of development times, as it will be seen in the following sub-section.

Table 4.1: Functional requirements

ID	Description	Objectives	Priority
FR1	Users STC measurement	The tool must calculate and display the STC levels of the users in the organization.	High
FR2	Teams STC measurement	The tool must calculate and display the STC levels of the teams in the organization.	High
FR3	Projects STC measurement	The tool must calculate the STC levels of the projects in the organization.	High
FR4	Recommendations	The system must provide recommendations based on the existing coordination gaps to improve the STC levels of the organization.	High
FR5	Dashboard: employees	The tool should allow to perform CRUD operations over employees as resources.	Medium
FR6	Dashboard: teams	The tool should allow to perform CRUD operations over teams as resources.	Medium
FR7	Dashboard: projects	The tool should allow to perform CRUD operations over projects as resources.	Medium
FR8	Dashboard: tasks	The tool should allow to perform CRUD operations over tasks as resources.	Medium
FR9	Dashboard: task assignments	The tool should allow to perform CRUD operations over tasks assignments as resources.	Medium
FR10	Dashboard: task dependencies	The tool should allow to perform CRUD operations over tasks dependencies as resources.	Medium
FR11	Dashboards: communications	The tool should allow to visualize communications within the organization.	Medium
FR12	Users STC visualization	The tool should display visual information about the users' STC levels along time.	Medium
FR13	Teams STC visualization	The tool should display visual information about the teams' STC levels along time.	Medium
FR14	Projects STC visualization	The tool should display visual information about the projects' STC levels along time.	Medium
FR15	Custom project's lack of coordination threshold	The tool will allow to customize from which threshold of lack of coordination recommendations should be provided for a given project.	Low
FR16	Custom users' lack of coordination threshold	The tool will allow to customize from which threshold of lack of coordination recommendations should be provided for a given pair of employees within a project.	Low

Table 4.2: Non-functional requirements

ID	Description	Objectives	Type
NFR1	Labels	Linguistic labels will be used to represent ambiguous terms, such as user experience, English level, or dependency weight between tasks.	Usability
NFR2	Web application	The application will be deployed as a web service, preferably at a local level, but with the possibility of public offering.	Usability
NFR3	Browser compatibility	The application must be able to run on desktop devices using conventional browsers such as Google Chrome or Mozilla Firefox.	Compatibility
NFR4	Cloud database	A database management system available in the cloud will be used, in order to facilitate the integration of the tool with remote systems.	Compatibility
NFR5	Performance	The application will run fluently in the interaction with its different components, offering an adequate performance and user experience.	Performance

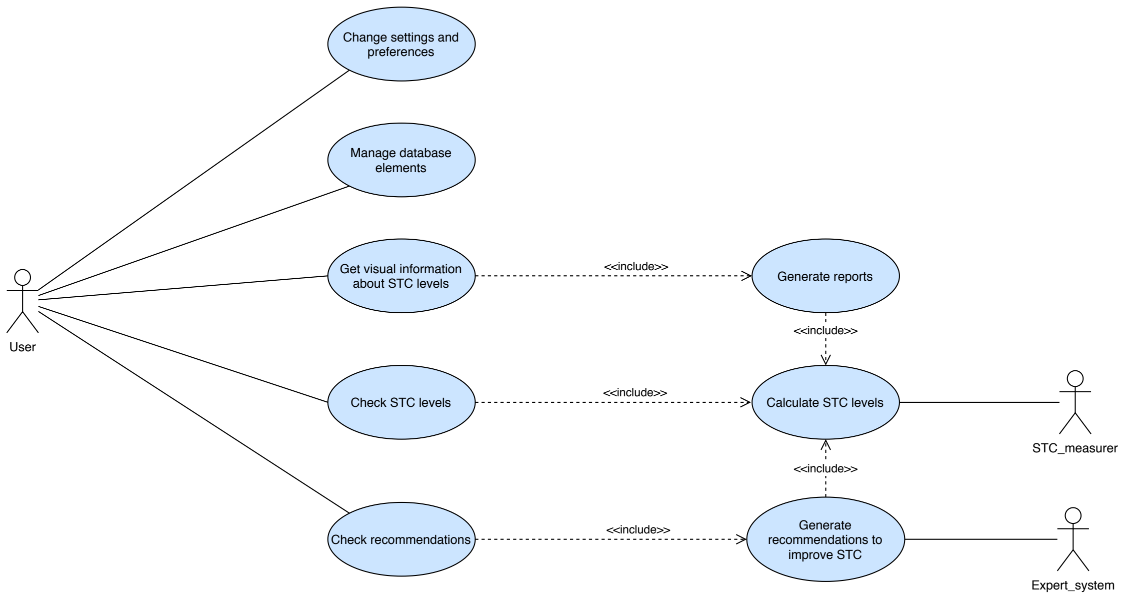


Figure 4.3: Use cases model

Table 4.3: Functional modules

ID	Module description	Objectives	Priority
M1	API skeleton	Development of the application skeleton, creation of the API resources as well as connection and operations on the database.	High
M2	Main user interface	Implementation of the dashboard functionalities and elaboration of the main windows of the web application.	High
M3	STC measurement	Extraction of assignments, dependencies, and current coordination, as well as coordination requirements and gaps. Computation of STC at employees, teams, and projects levels.	High
M4	Data visualization	Graphic representation of the STC levels history, allowing to see comparisons and the variation of these levels along time.	Medium
M5	Recommendation system	System for the detection of coordination gaps and recommendation of solutions to improve the levels of STC based on expert knowledge.	Medium
M6	Settings and preferences	Customization and tool settings configuration.	Low

Table 4.4: Project iterations

Iteration	Summary	Estimated time
IT0	Definition of general objectives, first draft and preliminary meetings.	1 week
IT1	Planning, requirements elicitation, role assignment, usecase modelling and design.	2 weeks
IT2	Module 1 implementation	1 week
IT3	Module 2 implementation	2 weeks
IT4	Module 3 implementation	4 weeks
IT5	Module 4 implementation	1 week
IT6	Module 5 implementation	3 weeks
IT7	Module 6 implementation	1 week
IT8	Documentation, tests and deployment	3 weeks
IT9	Review and submission preparation	2 weeks

Table 4.5: Project scheduling

Inception phase	Elaboration phase	Construction phase						Transition phase	
IT0	*IT1*	*IT2*	*IT3*	*IT4*	*IT5*	*IT6*	*IT7*	*IT8*	*IT9*
From: 7th January - To: 31st January	From: 1st February - To: 15th February	From: 16th February - To: 23th February	From: 24th February - To: 8th March	From: 9th March - To: 5th April	From: 6th April - To: 12th April	From: 13th April - To: 3rd May	From: 4th May - To: 10th May	From: 11th May - To: 31st May	From: 1st June - To: 14th June
1 week	2 weeks	1 week	2 weeks	4 weeks	1 week	3 weeks	1 week	3 weeks	2 weeks
1 week	2 weeks	12 weeks						3 weeks	
Total: 18 weeks									

4.2.3. Planification

Having identified the functionalities to be implemented and the general project decomposition, it was divided into the iterations shown in Table 4.4. Each one, with an estimated duration, would represent an evaluable and revisable micro-increment of the application.

Based on these iterations, the project schedule was established (see Table 4.5). This was largely achieved, apart from some slight fluctuations which did not affect the project, as they were due to improvements in the different functional modules, rather than delays.

4.2.4. Cost estimation

In this section, the cost estimation of the project, calculated from the working time and resources used, is broken down. For this estimation, the variables shown in Table 4.6 have been considered, giving rise to the total cost of 7206.6 €.

The salary per hour of the analyst-programmer is reflected in [53], with data updated up to May 2020. At the same time, the price of software resources has been omitted because only free software or software covered by academic licenses has been used.

Table 4.6: Project cost estimation

Variable	Value
Total working hours	$\approx$ 630 hours $*$ 11,22 €/hour = 7068 €
Indirect investments (resources, electricity consume, etc.)	138.6 €
	7206.6 €

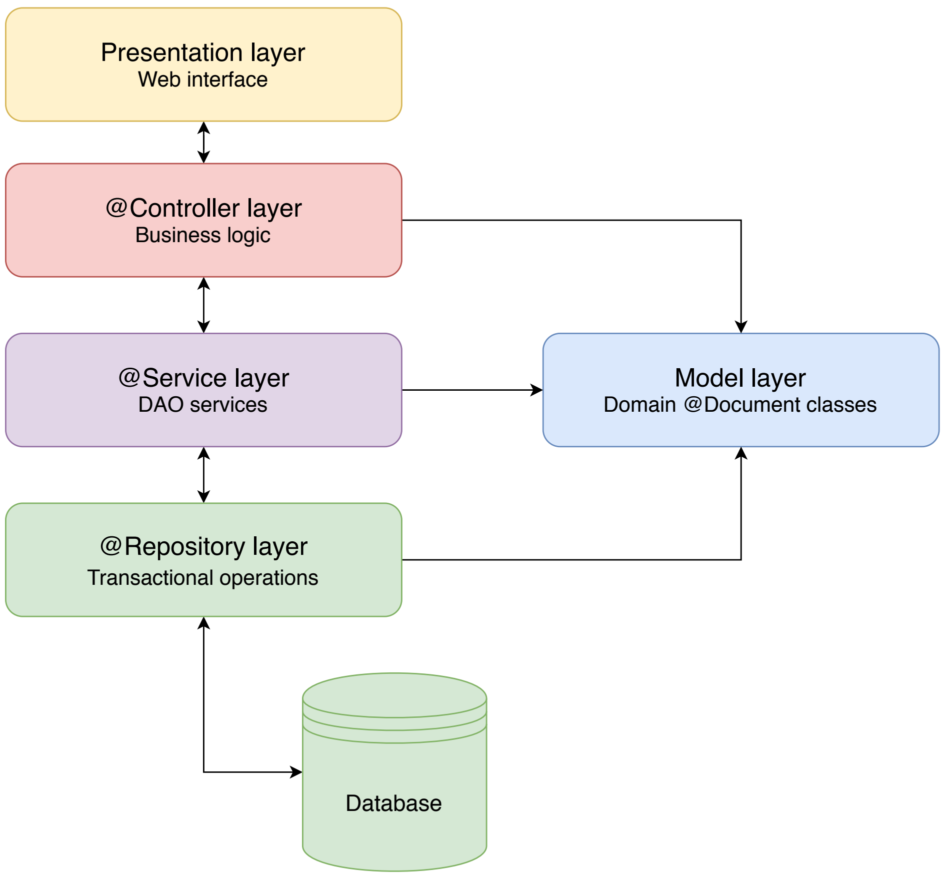


Figure 4.4: Layered architecture

4.2.5. Architecture and design

The elaboration phase of the OpenUp lifecycle includes the selection of the architecture to be employed. So, at this stage it was decided that FENCE would be a client-server application based on a layered architecture, whose characteristics are described below. The choice of client-server as the architecture of FENCE was agreed in order to be able to deploy the tool as a web service if necessary. In this way, this would be a portable application, which would allow the user to run the tool from any browser without the need for additional software.

Moreover, a preliminary data model was also established, establishing the main collections and documents that would comprise it, as well as their attributes and possible relationships.

LAYERED ARCHITECTURE

Layer-based architecture [41] is one of the most common design patterns in the development of client-server web applications. In this type of architecture, the different modules that comprise the system (layers) are organized horizontally, so that each layer of the application fulfils a specific role.

Spring Boot [49] provides a great simplicity when implementing this type of RESTful layered services [14], so it is on this basis on which the FENCE architecture is based (see Figure 4.4 and Figure 4.5):

- **Presentation**. It is the layer in charge of the interaction with the user. In this case, it includes the static resources (HTML views and CSS code), as well as the underlying JavaScript code. This layer, conceived as the client side, communicates with the server through GET, POST, PUT or DELETE Ajax requests.
- **Model**. This is the layer where the classes that model real-world entities are found. These classes, making use of the Spring framework integration with MongoDB, are labelled as

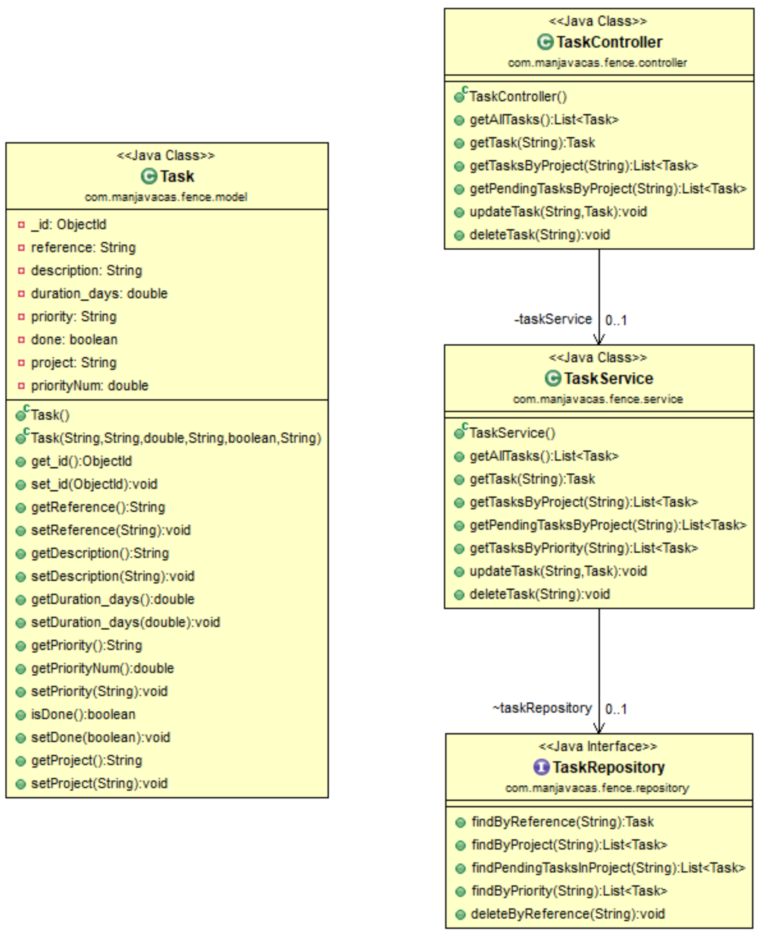

<<Java Class>>
TaskController
com.manjavacas.fence.controller
TaskController()
getAllTasks():List<Task>
getTask(String):Task
getTasksByProject(String):List<Task>
getPendingTasksByProject(String):List<Task>
updateTask(String,Task):void
deleteTask(String):void
-taskService
0..1
<<Java Class>>
TaskService
com.manjavacas.fence.service
TaskService()
getAllTasks():List<Task>
getTask(String):Task
getTasksByProject(String):List<Task>
getPendingTasksByProject(String):List<Task>
getTasksByPriority(String):List<Task>
updateTask(String,Task):void
deleteTask(String):void
~taskRepository
0..1
<<Java Interface>>
TaskRepository
com.manjavacas.fence.repository
findByReference(String):Task
findByProject(String):List<Task>
findPendingTasksInProject(String):List<Task>
findByPriority(String):List<Task>
deleteByReference(String):void
<<Java Class>>
Task
com.manjavacas.fence.model
_id: ObjectId
reference: String
description: String
duration_days: double
priority: String
done: boolean
project: String
priorityNum: double
Task()
Task(String,String,double,String,boolean,String)
get_id():ObjectId
set_id(ObjectId):void
getReference():String
setReference(String):void
getDescription():String
setDescription(String):void
getDuration_days():double
setDuration_days(double):void
getPriority():String
getPriorityNum():double
setPriority(String):void
isDone():boolean
setDone(boolean):void
getProject():String
setProject(String):void

Figure 4.5: Sample connection between layers

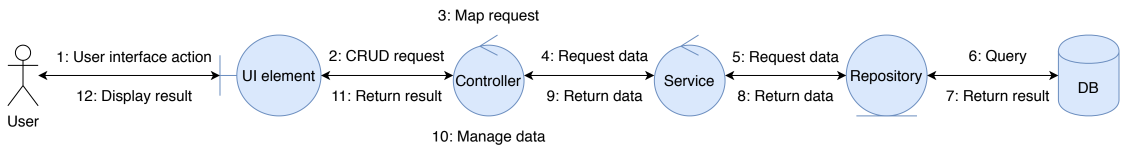


Figure 4.6: Layered interaction

documents (*@Document*), so that they represent the basic structure of the documents included in the database's collections.

- **Controller**. The controllers are the classes in charge of handling the CRUD requests received from the presentation layer. They make use of the classes defined in the Model layer, are labelled as *@Controller*, and establish, for each of the CRUD operations, a mapping with their methods. In general, this layer covers business logic operations, so we can also find controller classes in charge of performing other transactions besides attending to requests from the presentation layer. Furthermore, when any of these operations requires access to the database, communication with the underlying DAO (Data Access Only) layers will be necessary.
- **Service**. This layer includes the set of DAO classes that interact with the database. They are labelled as *@Service* and include the logic needed to access repositories and handle controller requests.
- **Repository**. Repository. MongoDB repositories are a type of interface included in Spring that provides easy and flexible interaction with MongoDB collections. They allow database queries to be abstracted and customized and are labelled as *@Repository*.

If we go deeper into the FENCE architecture at a class level, we will always find a similar pattern: the class controller attends to CRUD requests from the interface and, in case of requiring stored data, on request to the service class, the data is retrieved through the corresponding repository interface. This general procedure is summarized in Figure 4.6.

DATA MODEL

The reasons for using a noSQL database system like MongoDB are the advantages that this type of environments provide:

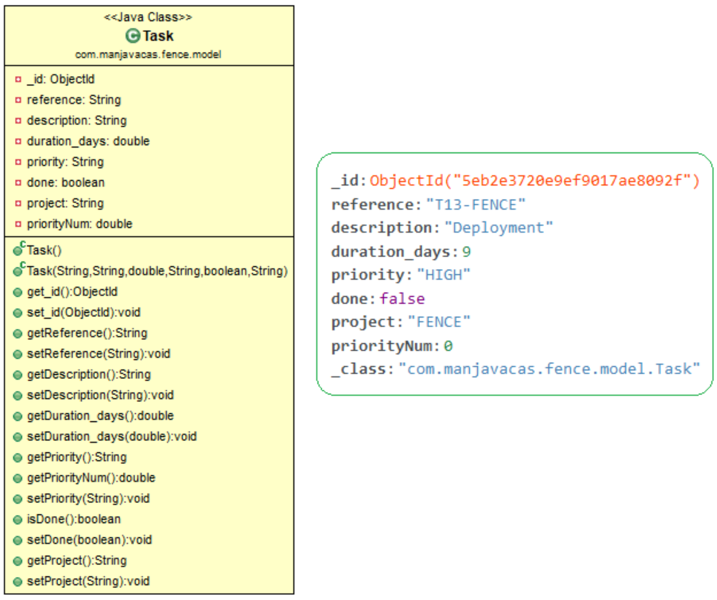
- On the one hand, they offer great flexibility during development, especially if we consider the easy integration of MongoDB with Spring framework.
- On the other hand, the modification of noSQL schemes is very simple, fast, and scalable.
- In addition, MongoDB databases also integrate SQL features and queries, so we can have the best of both types of databases in the same system.

However, although there are disadvantages when using noSQL, such as manual control of data consistency or operation atomicity, the data model of FENCE was not considered complex enough to make a SQL system more convenient.

As indicated, Spring's integration with MongoDB is relatively simple. We only need to label the classes we want to represent as database documents with *@Document* in order to establish their schema. This will make it indexable through the *@Repository* interfaces and easily accessible. An of the correspondence between a class labelled as *@Document* and its instance in the database is shown in Figure 4.7.

On this basis, the scheme shown in Figure 4.8 represents the data model of FENCE, where the *@Document* classes and their relationships can be appreciated.

Therefore, once the architecture and data model of the system had been defined, the implementation of the different functional modules of FENCE was undertaken.



```
_id: ObjectId("5eb2e3720e9ef9017ae8092f")
reference: "T13-FENCE"
description: "Deployment"
duration_days: 9
priority: "HIGH"
done: false
project: "FENCE"
priorityNum: 0
_class: "com.manjavacas.fence.model.Task"
```

Figure 4.7: Relation between Java class and MongoDB document

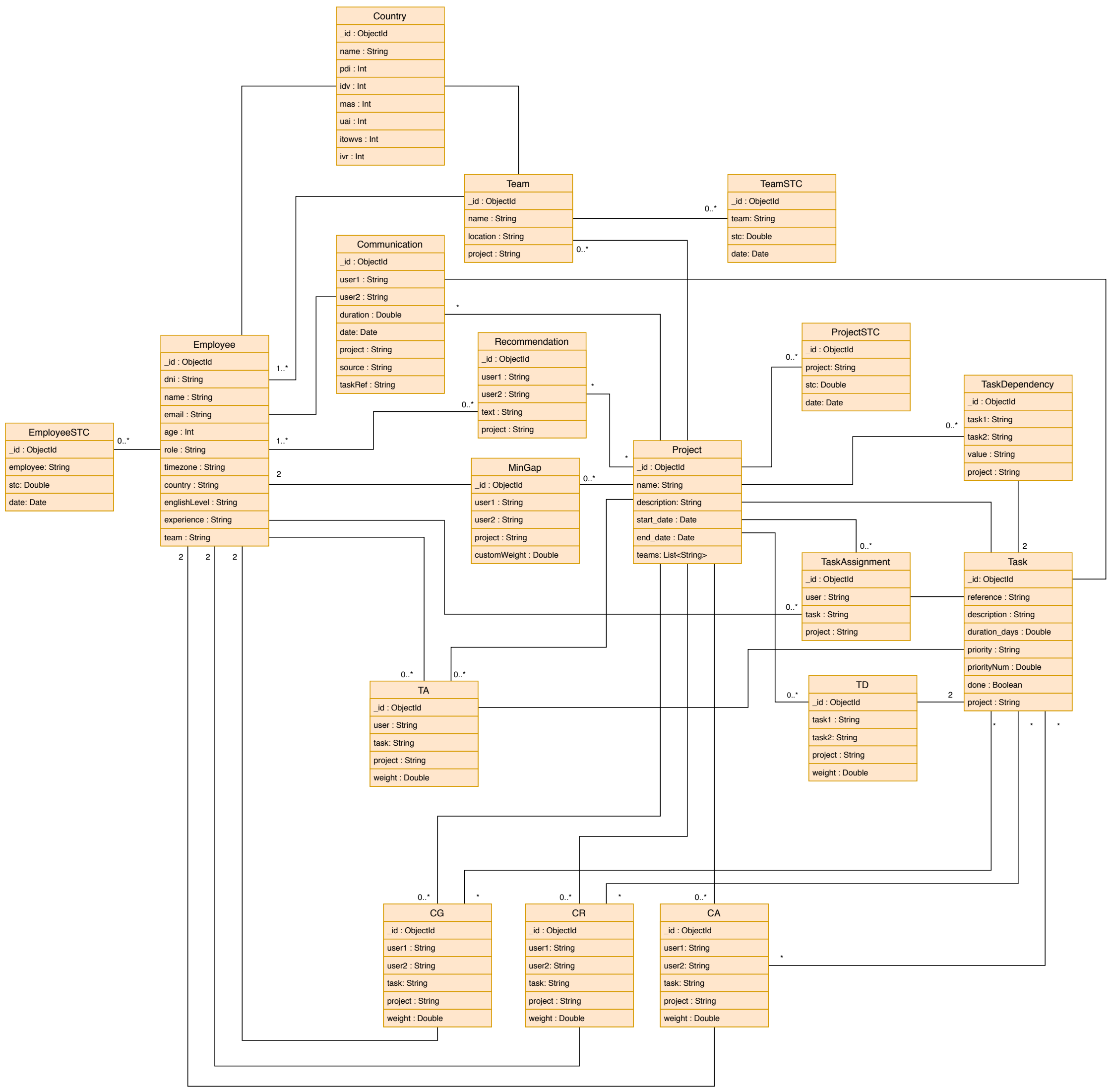
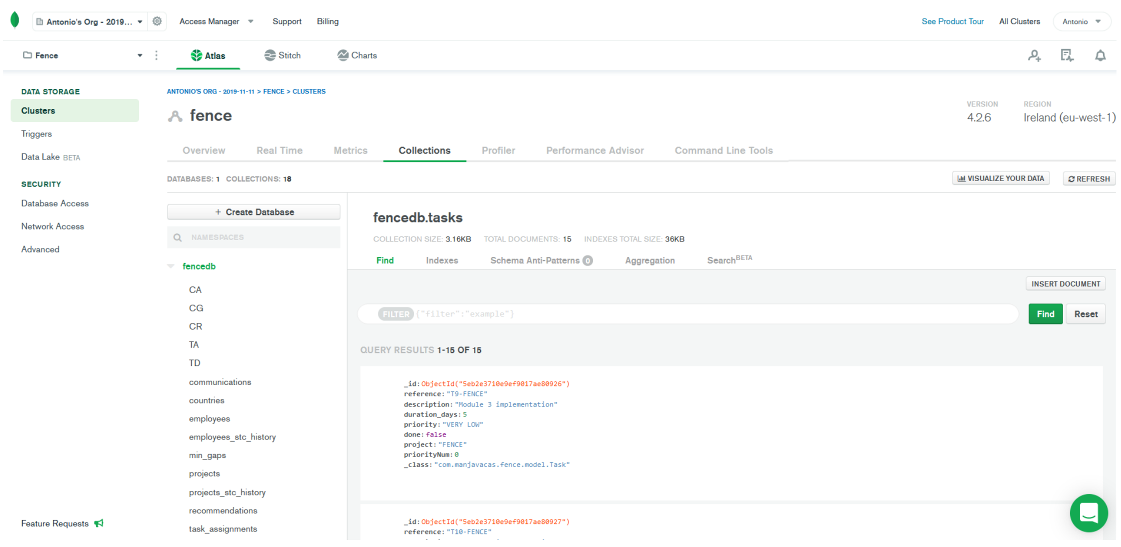


Figure 4.8: FENCE data model



```
_id: ObjectId("5eb2e3710e9ef9017ae80926")
reference: "T9-FENCE"
description: "Module 3 implementation"
duration_days: 5
priority: "VERY LOW"
done: false
project: "FENCE"
priorityNum: 0
_class: "com.manjavacas.fence.model.Task"

_id: ObjectId("5eb2e3710e9ef9017ae80927")
reference: "T10-FENCE"
```

Figure 4.9: MongoDB Atlas cluster

4.3. CONSTRUCTION PHASE

The following subsections will address in detail the process of developing the functionalities of FENCE, as well as the difficulties and decisions faced in its implementation. The implementation of each of these modules meant a micro-increment in the functionalities of FENCE, giving as a result the first version of the application.

4.3.1. Module 1. API skeleton

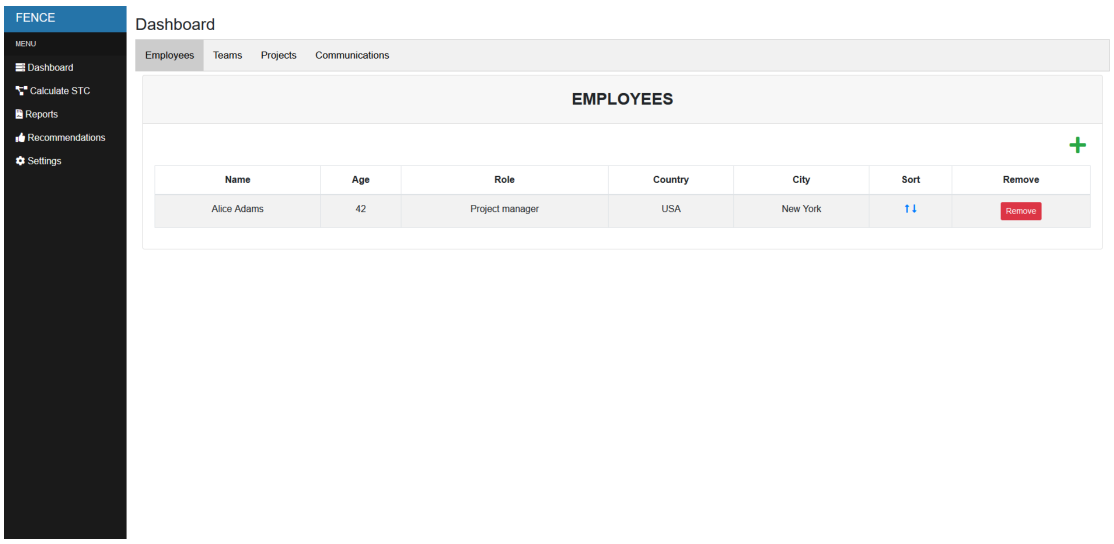
The first objective pursued in this phase was the creation of the initial set of resources that compose the application, as well as the main windows that would define the structure of the system's interface. Throughout this stage, the main structure of the data model was established, which would be progressively expanded throughout the project.

Thus, taking as a reference this initial data model and the previously designed class diagram, the different classes, URIs and query methods were defined. Likewise, as shown in Figure 4.9, the MongoDB cluster hosted in the MongoDB Atlas cloud service was created, establishing access permissions, integrating the MongoDB Java Driver into the Maven project, and making queries operational through Spring.

Once the cluster's database was created and correctly integrated into the system, the main CRUD operations were added for each of the resources, defining the necessary methods in Controllers, Services and Repositories. Following the model provided by Spring framework, each application resource is represented by a class from the Model package, as previously explained. On the other hand, controllers are in charge of the main operations and communication with the interface, while service classes act as DAO entities communicating with the MongoDB repositories (as shown in Figure 4.4).

Therefore, when the logical skeleton of the application had been created, the two main windows of the system were elaborated. On the one hand, the main control panel that would be gradually completed in the different iterations of the project, whose first version is shown in Figure 4.10.

On the other hand, a home page was developed with information about the utility, version, and authorship of FENCE, whose final version is shown in Figure 4.11.



Name	Age	Role	Country	City	Sort	Remove
Alice Adams	42	Project manager	USA	New York	↑↓	Remove

Figure 4.10: First dashboard version

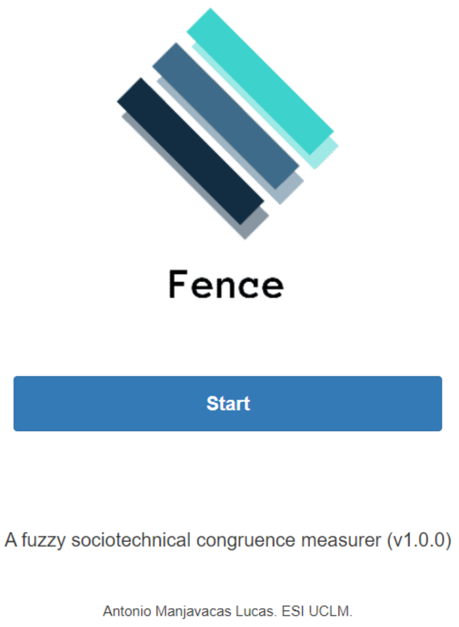
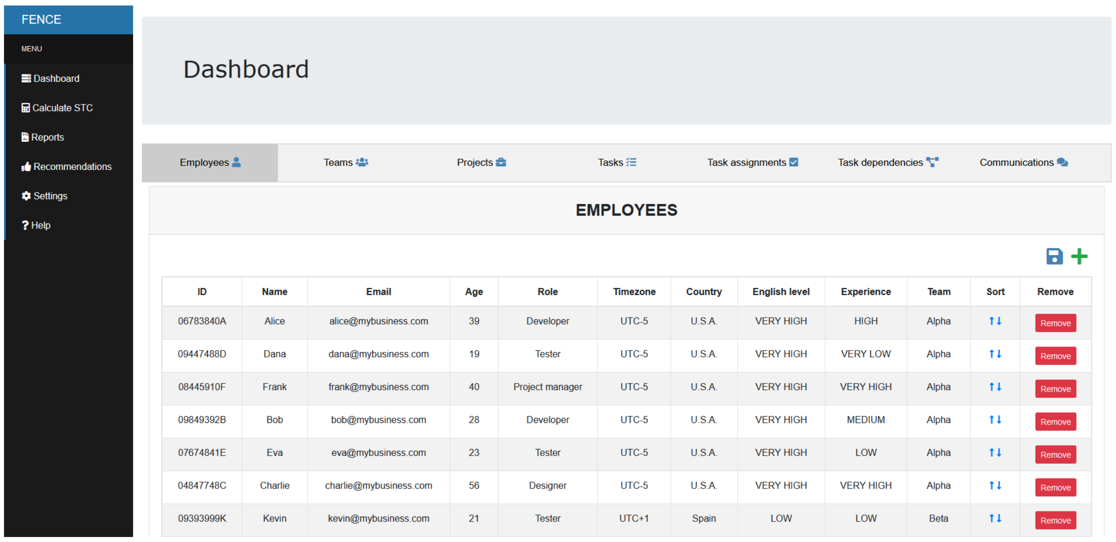

Fence
Start
A fuzzy sociotechnical congruence measurer (v1.0.0)
Antonio Manjavacas Lucas. ESI UCLM.

Figure 4.11: FENCE home page



EMPLOYEES

ID	Name	Email	Age	Role	Timezone	Country	English level	Experience	Team	Sort	Remove
06783840A	Alice	alice@mybusiness.com	39	Developer	UTC-5	U.S.A.	VERY HIGH	HIGH	Alpha	↑↓	Remove
09447488D	Dana	dana@mybusiness.com	19	Tester	UTC-5	U.S.A.	VERY HIGH	VERY LOW	Alpha	↑↓	Remove
08445910F	Frank	frank@mybusiness.com	40	Project manager	UTC-5	U.S.A.	VERY HIGH	VERY HIGH	Alpha	↑↓	Remove
09849392B	Bob	bob@mybusiness.com	28	Developer	UTC-5	U.S.A.	VERY HIGH	MEDIUM	Alpha	↑↓	Remove
07674841E	Eva	eva@mybusiness.com	23	Tester	UTC-5	U.S.A.	VERY HIGH	LOW	Alpha	↑↓	Remove
04847748C	Charlie	charlie@mybusiness.com	56	Designer	UTC-5	U.S.A.	VERY HIGH	VERY HIGH	Alpha	↑↓	Remove
09393999K	Kevin	kevin@mybusiness.com	21	Tester	UTC+1	Spain	LOW	LOW	Beta	↑↓	Remove

Figure 4.12: FENCE dashboard

4.3.2. Module 2. Main user interface

In this second construction stage, efforts were focused on improving and completing the interface outlined in the previous module. This implied a greater deepening in HTML, CSS and JavaScript solutions, making use of Bootstrap and jQuery as nexus to implement the different functionalities of the interface. Initially, the use of these technologies never employed was an individual challenge, and therefore much of the effort invested at this stage was related with learning them.

An objective pursued in this stage was the elaboration of the different tabs and panels that compose the system's dashboard, as well as the communication between the interface and the application controllers via jQuery's Ajax requests. These requests were carried out according to the controllers' URIs defined in the previous construction stage, following the communication model expressed in Figure 4.6.

Furthermore, as can be seen in the Figure 4.12, these were the tabs contained the dashboard in its final version:

- **Employees**: includes information on the users that are part of the organisation, used to measure individuals' STC, such as age, experience, English level, country, etc.
- **Teams**: includes the name, location, and project membership of the different teams in the organization.
- **Projects**: active projects in the organization.
- **Tasks**: tasks to be carried out in each of the projects, together with their description, duration, priority, and completion.
- **Task assignments**: information about task assignment to users.
- **Task dependencies**: information on dependencies between tasks and their reliance weight.
- **Communications**: communications history between users. We will see in future sections their utility and how they are gathered.

Once the dashboard was implemented and the main application windows were set, some test data was loaded, while the proper performance of methods and queries were tested.

4.3.3. Module 3. STC measurement

Up to now, the creation of the main resources of the tool, their storage, and the CRUD interaction with them through the dashboard have been presented. In this stage, the objective pursued was to use these resources to carry out the measurement of the STC at the different levels of the organization

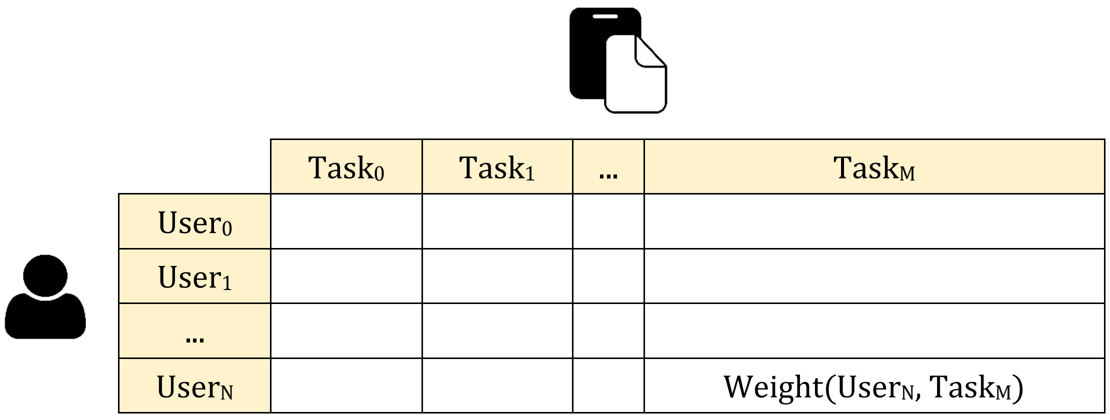


Figure 4.13: T_A matrix

under consideration: users, teams, and projects.

As previously explained in chapter 1, the procedure taken as a reference to measure STC was the one exposed by Kwan et al. [24, 25], given its facility to be adapted to the features and requirements of FENCE. Therefore, in the following subsections we will see gradually how the STC is calculated through the different algorithms implemented in this module:

TASK ASSIGNMENT MATRIX (T_A)

The first step in computing STC levels is to construct the task assignment matrix (T_A). This matrix matches tasks with users, representing the degree of user involvement in the tasks assigned to them (see Figure 4.13).

Both for this case and for the rest of the matrices detailed in the literature, in order to reduce the complexity and computational cost of the algorithms, these matrices were replaced by lists of objects stored in the database. In this case, the class T_A represents a concrete cell in the T_A matrix, as shown in Figure 4.14. Moreover, the process carried out to fill in this matrix consists on the one shown in Algorithm 1.

Finally, the generated records are stored in the database replacing older ones, always keeping the most up-to-date T_A matrix in the database.

TASK DEPENDENCY MATRIX (T_D)

The task dependency matrix (T_D) is very similar to the previous one, except that in this case the matrix does not relate users to tasks but tasks to each other. Specifically, what this matrix reflects is the degree of dependency between tasks in a project (see Figure 4.15). As this dependency level is flexible, it can be adapted to the user's understanding of "dependency". For example, the weight of the dependency between two tasks may reflect the number of functionalities of one task that must be previously implemented to perform the other.

As in the previous case, the different cells of the T_D matrix are stored as objects in the database (see Figure 4.16). The process of their creation is the one shown in Algorithm 2.

Both the calculation of T_D and T_A is relatively quick, since the operations performed consist mainly on entering the information provided by the user in objects that are stored in the database. However, this is not the case of the C_R and C_A matrices, which will be presented in the following subsections.

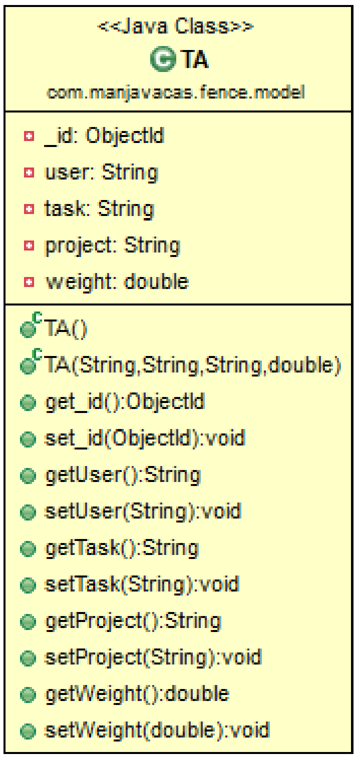
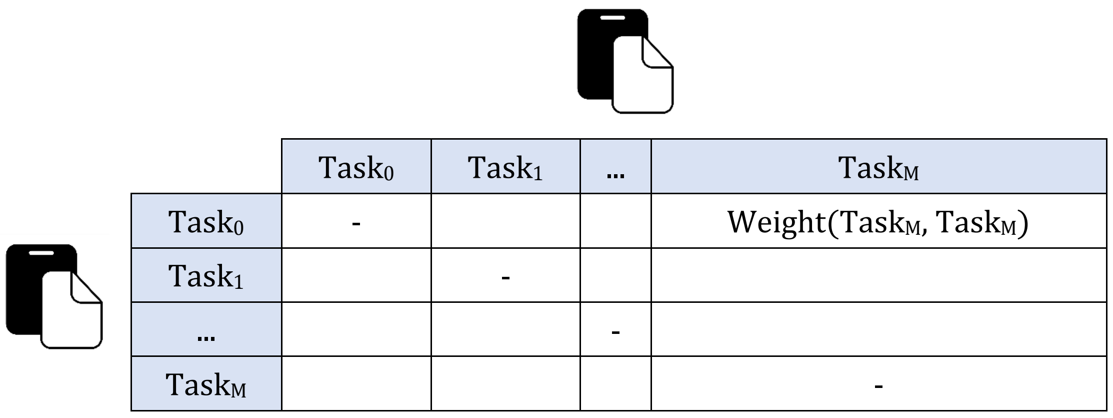


Figure 4.14: T_A class

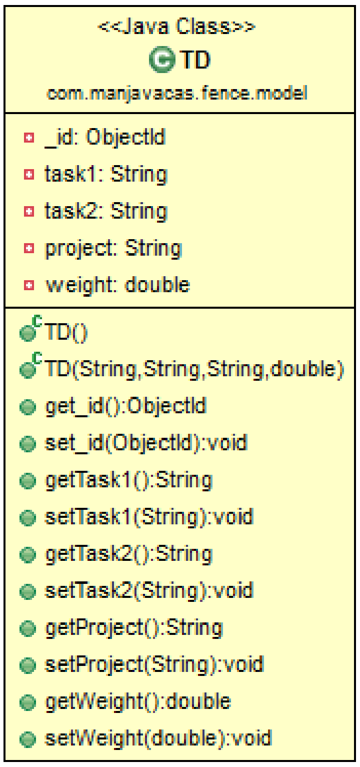
Algorithm 1: T_A calculation

```
input  : project
output : $T_A$ matrix
1  $taMatrix \longleftarrow \emptyset$
2  $tasks \longleftarrow getPendingTasks(project)$
3  foreach $task$ in $tasks$ do
4      $users \longleftarrow getUsersAssignedTo(task)$
5      if $users \neq \emptyset$ then
6          $sumExp \longleftarrow sumExperiences(users)$
7          foreach $user$ td in $users$ do
8              $weight \longleftarrow (user.experience/sumExp) * task.priority$
9              if $weight > 1$ then
10                 weight $\longleftarrow$ 1
11             end
12             $taMatrix \cup T_A(user, task, weight, project)$
13         end
14     end
15 end
```



	$Task_0$	$Task_1$	...	$Task_M$
$Task_0$	-			Weight($Task_M$, $Task_M$)
$Task_1$		-		
...			-	
$Task_M$				-

Figure 4.15: T_D matrix

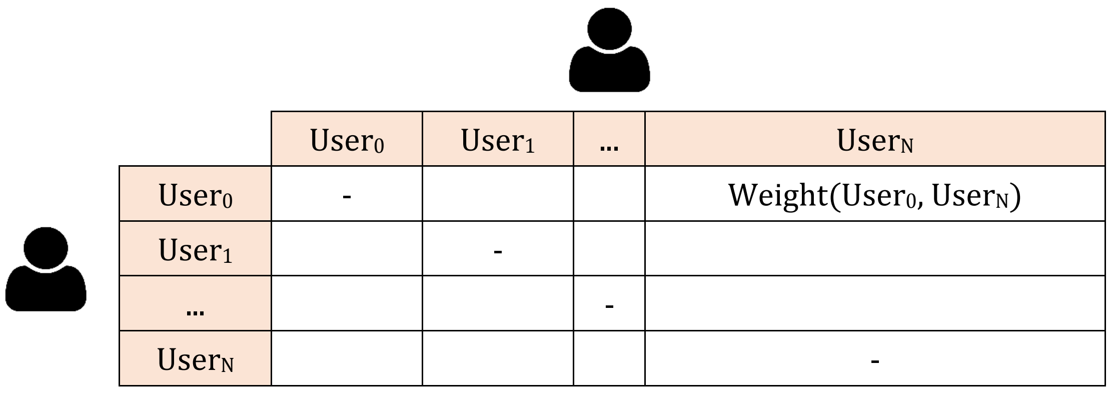


```
<<Java Class>>
TD
com.manjavacas.fence.model

_id: ObjectId
task1: String
task2: String
project: String
weight: double

TD()
TD(String,String,String,double)
get_id():ObjectId
set_id(ObjectId):void
getTask1():String
setTask1(String):void
getTask2():String
setTask2(String):void
getProject():String
setProject(String):void
getWeight():double
setWeight(double):void
```

Figure 4.16: T_D class

Algorithm 2: T_D calculation

```
input  : project
output : $T_D$ matrix
$tdMatrix \longleftarrow \emptyset$
$tasks \longleftarrow getPendingTasksOf(project)$
foreach $task$ in $tasks$ do
    $taskDependencies \longleftarrow getDependenciesOf(task)$
    if $taskDependencies \neq \emptyset$ then
        $sumValues \longleftarrow sumWeights(taskDependencies)$
        foreach $taskDependency$ in $taskDependencies$ do
            $weight \longleftarrow taskDependency.weight/sumValues$
            $tdMatrix \cup T_D(task, taskDependency, weight, project)$
        end
    end
end
```



	User0	User1	...	UserN
User0	-			Weight(User0, UserN)
User1		-		
...			-	
UserN				-

Figure 4.17: C_R matrix

COMMUNICATION REQUIREMENTS MATRIX (C_R)

Once the T_A and T_D matrices are obtained, we can compute the coordination requirements between users represented in the C_R matrix (see Figure 4.17 and Figure 4.18). This matrix represents the communication needs between user pairs based on the tasks assigned to them and their dependencies. In this case, the weights of the different matrix cells represent how much the users involved must coordinate.

Initially calculated by means of the operations shown in Equation 4.1, in this case we will follow an iterative procedure to calculate these coordination requirements, as no matrices are being used:

$$C_R = T_A * T_D * T_A^T \tag{4.1}$$

As can be shown in Algorithm 3, in the calculation of the weights for each coordination requirement, this weight is subsequently adjusted according to different distances present in Global Software Development projects. Thus, a fuzzy control system is used to carry out this adjustment, whose objective is none other than trying to reflect the difficulties or "penalties" to communication that may be caused by socio-cultural, geographical, or temporal distances among users.

The entries taken by the fuzzy control system are the following:

- **Common English level**. Level of English shared by users, assuming that communication

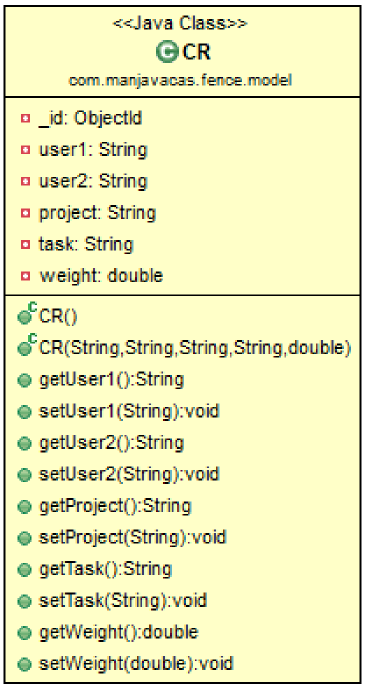


Figure 4.18: C_R class

Algorithm 3: C_R calculation

```
input  : project, $T_A$ matrix, $T_D$ matrix
output : $C_R$
$crMatrix \longleftarrow \emptyset$
foreach $taskAssignment\ ta\ in\ taMatrix$ do
    $user_1 \longleftarrow ta.user$
    $dependencies \longleftarrow getDependenciesOf(ta.task)$
    foreach $taskDependency\ td\ in\ dependencies$ do
        $responsibles \longleftarrow getUsersAssignedTo(td.task_2)$
        foreach $user\ user_2\ in\ responsibles$ do
            if $user_2 \neq user1$ then
                $weight_{T_A}1 \longleftarrow ta.weight$
                $weight_{T_A}2 \longleftarrow T_A(td.task_2, user_2).weight$
                $weight_{T_D} \longleftarrow T_D(ta.task, td.task_2).weight$

                $weight \longleftarrow weight_{T_A}1 * weight_{T_A}2 * weight_{T_D}$
                $weight \mathrel{+}= computeGlobalFactors(user_1, user_2)$

                if $weight > 1$ then
                    weight $\longleftarrow$ 1
                end

                $crMatrix \cup C_R(user_1, user_2, ta.task, weight, project)$
            end
        end
    end
end
```

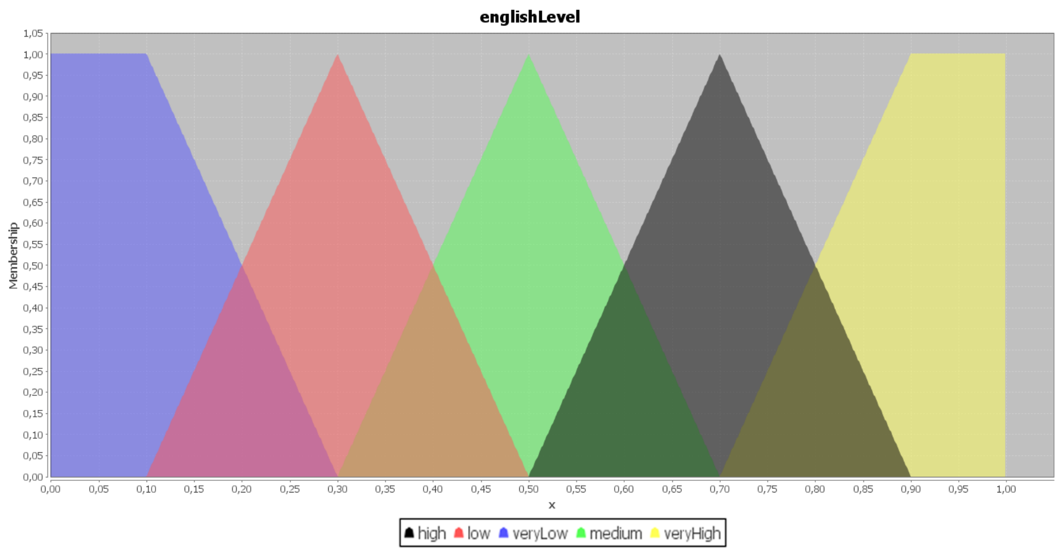


Figure 4.19: English level fuzzy sets

between them is always in this language despite their country provenance. It can be assumed that even if two users communicate in their native language, the level of English must be considered if the information disseminated by public chats (i.e. Jira, GitHub, Teams, Slack, etc.) should be interpretable by other users. Depending on the individual level of each user, the following weights are assessed:

- If both users have a high level of English, the common level of English is 1.
- If one user has a high level and the other has a medium level, the weight is 0.7.
- If both have a medium level of English, the common level of English is 0.5.
- On the other hand, if one user has a medium level of English and the other has a low level, the resulting weight is 0.3.
- Finally, if both users have a low level of English the common weight is 0.

Moreover, the fuzzy sets associated with this variable are shown in the Figure 4.19.

- **Cultural distance**. Average of the distances between socio-cultural factors identified by G. Hofstede [16], according to the employees' nationality[1]. These are the ones deeply explained in [18], [17]:

  - **Power distance** (PDI): degree of acceptance of power differences or equality in society. In societies with higher PDI, higher social classes are more questioned, while in cultures with lower PDI, there is more equal power among people.
  - **Individualism** (IDV): defines the level of integration of individuals into society and the development of a sense of group membership. Compared to more individualistic countries we find others with a collectivist orientation.
  - **Masculinity** (MAS): defines a culture's tendency towards more masculine or feminine patterns of behaviour. According to Hofstede, male societies are more assertive, competitive and focused on outcomes than the generally more empathetic female societies.
  - **Uncertainty avoidance** (UAI): acceptance of situations of uncertainty or where there is no absolute truth. It represents the degree of relativism of societies.
  - **Long term orientation** (LTO): this cultural factor differentiates between more tradi-

[1]These socio-cultural factors have been quantified by experts and are freely available for academic research purposes at https://geerthofstede.com/research-and-vsm/dimension-data-matrix/.

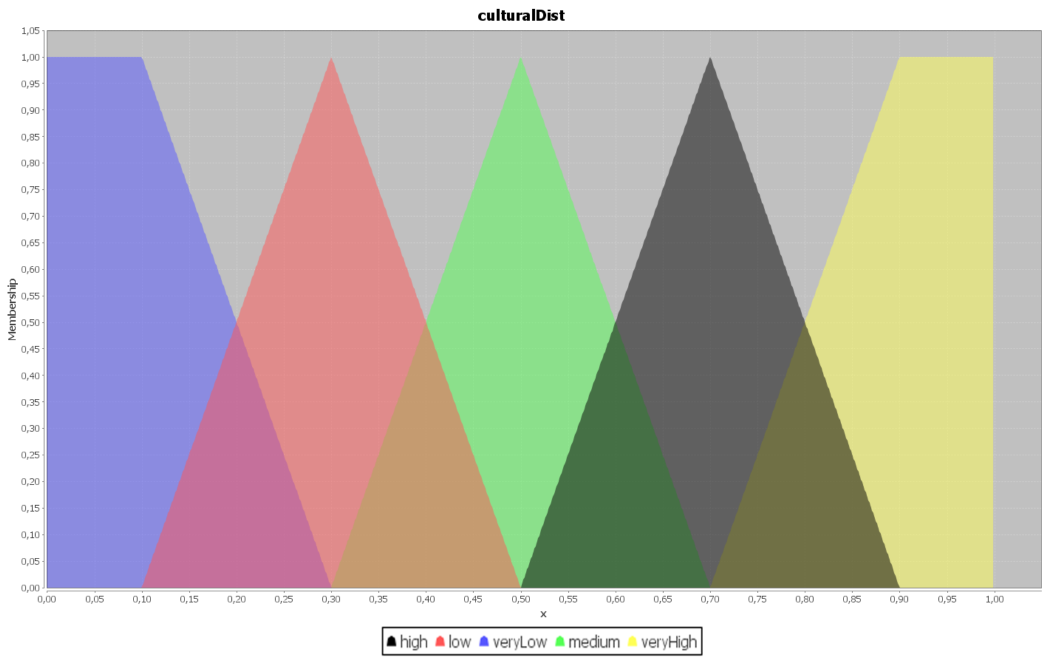


Figure 4.20: Cultural distance fuzzy sets

tionalist societies, with long-term aspirations, and those with short-term orientations where social relations are seen as a means to gain some kind of benefit.

- **Indulgence** (IVR): explains the vision of life that every culture has. On the one hand, indulgent cultures tend to be more optimistic and positive than contained cultures. On the other hand, contained cultures consider a more reserved enjoyment.

Thus, all these factors were inserted in a collection where each document represents a country with the cultural features differentiated in the mentioned study.

On this basis, the cultural distance (CD) is calculated by following the Equation 4.2.

$$CD = \frac{|pdi_1 - pdi_2| + |idv_1 - idv_2| + |mas_1 - mas_2| + |uai_1 - uai_2| + |lto_1 - lto_2| + |ivr_1 - ivr_2|}{6} \tag{4.2}$$

Similarly, the fuzzy sets considered for this variable are shown in the Figure 4.20.

- **Time overlap**. Coincidence between employees' time zones (UTC), calculated as the inverse of their time difference. A greater time overlap will facilitate the use of synchronous means of communication, while the impossibility of temporal coincidence in communication will result in greater use of asynchronous means. The computation of the time overlap is carried out following the Equation 4.3, while the considered fuzzy sets for this input variable are shown in the Figure 4.21.

$$TimeDiff = \begin{cases} 1 - \frac{UTC_1 - UTC_2}{26}, UTC_1 \geq UTC_2 \\ \\ 1 - \frac{UTC_2 - UTC_1}{26}, UTC_1 < UTC_2 \end{cases} \tag{4.3}$$

In turn, the set of rules defined using FCL and used by the system is displayed in Table 4.7.

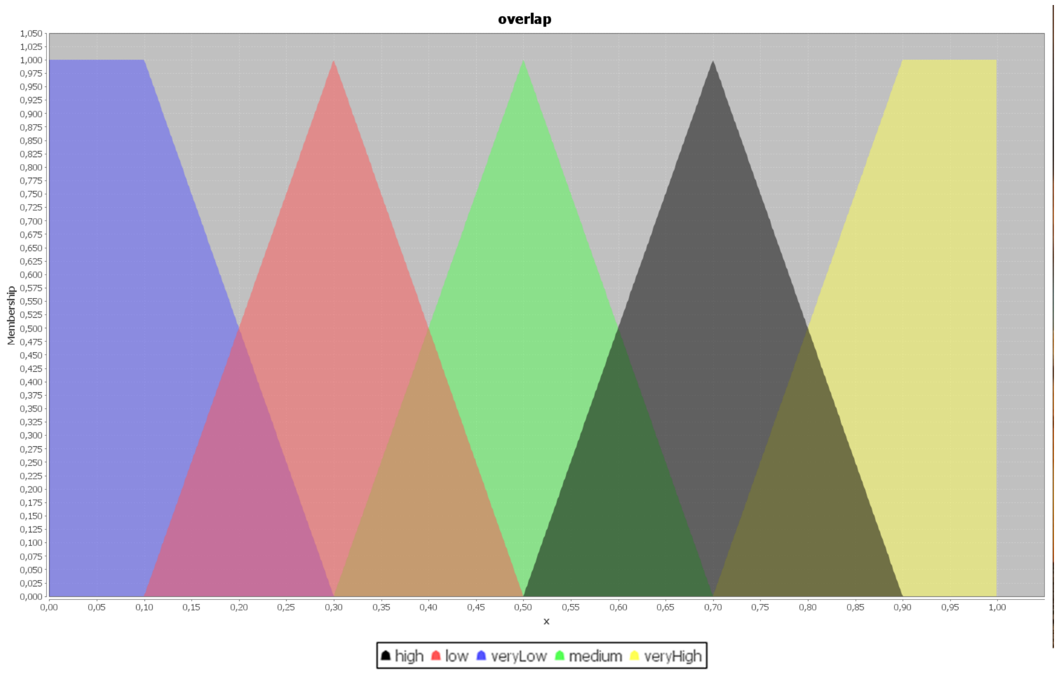
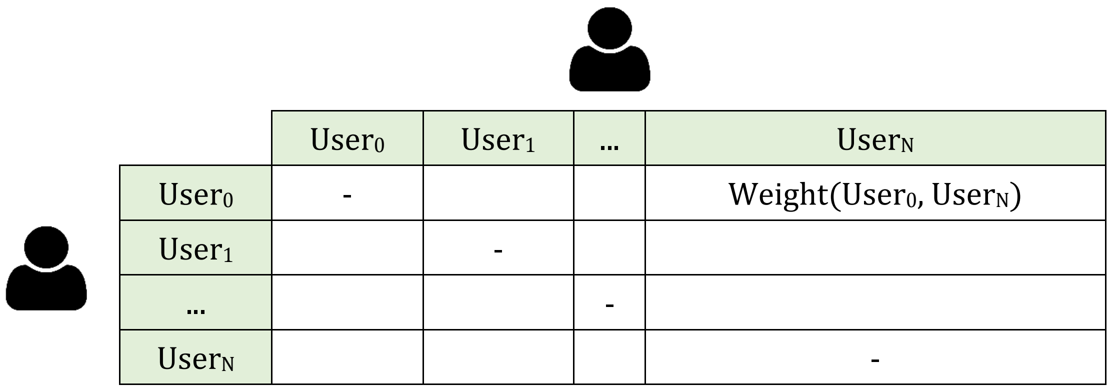


Figure 4.21: Time overlap fuzzy sets

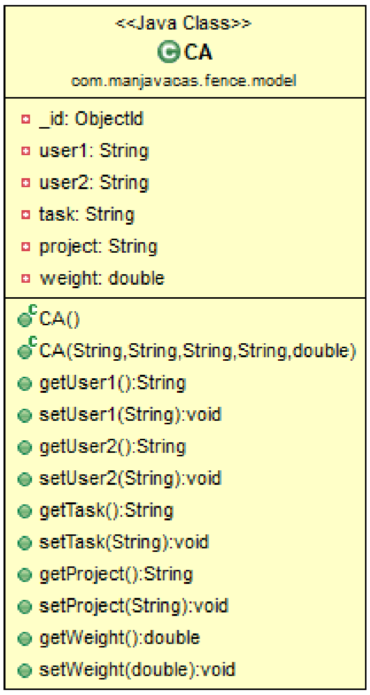
Table 4.7: C_R fuzzy control system rules

N	FCL Rule
1	IF (culturalDist IS low OR culturalDist IS veryLow) AND (overlap IS low OR overlap IS veryLow) AND (englishLevel IS low OR englishLevel IS veryLow) THEN modifier IS medium;
2	IF (culturalDist IS low OR culturalDist IS veryLow) AND (overlap IS low OR overlap IS veryLow) AND (englishLevel IS medium OR englishLevel IS high OR englishLevel IS veryHigh) THEN modifier IS medium;
3	IF (culturalDist IS low OR culturalDist IS veryLow) AND (overlap IS medium) AND (englishLevel IS low OR englishLevel IS veryLow) THEN modifier IS low;
4	IF (culturalDist IS low OR culturalDist IS veryLow) AND (overlap IS medium) AND (englishLevel IS medium OR englishLevel IS high OR englishLevel IS veryHigh) THEN modifier IS medium;
5	IF (culturalDist IS low OR culturalDist IS veryLow) AND (overlap IS high OR overlap IS veryHigh) AND (englishLevel IS low OR englishLevel IS veryLow) THEN modifier IS veryLow;
6	IF (culturalDist IS low OR culturalDist IS veryLow) AND (overlap IS high OR overlap IS veryHigh) AND (englishLevel IS medium OR englishLevel IS high OR englishLevel IS veryHigh) THEN modifier IS veryLow;
7	IF (culturalDist IS medium OR culturalDist IS high OR culturalDist IS veryHigh) AND (overlap IS low OR overlap IS veryLow) AND (englishLevel IS low OR englishLevel is veryLow) THEN modifier IS veryHigh;
8	IF (culturalDist IS medium OR culturalDist IS high OR culturalDist IS veryHigh) AND (overlap IS low OR overlap IS veryLow) AND (englishLevel IS medium OR englishLevel IS high OR englishLevel IS veryHigh) THEN modifier IS high;
9	IF (culturalDist IS medium OR culturalDist IS high OR culturalDist IS veryHigh) AND (overlap IS medium) AND (englishLevel IS low OR englishLevel is veryLow) THEN modifier IS veryHigh;
10	IF (culturalDist IS medium OR culturalDist IS high OR culturalDist IS veryHigh) AND (overlap IS medium) AND (englishLevel IS medium OR englishLevel IS high OR englishLevel IS veryHigh) THEN modifier IS medium;
11	IF (culturalDist IS medium OR culturalDist IS high OR culturalDist IS veryHigh) AND (overlap IS high OR overlap IS veryHigh) AND (englishLevel IS low OR englishLevel is veryLow) THEN modifier IS high;
12	IF (culturalDist IS medium OR culturalDist IS high OR culturalDist IS veryHigh) AND (overlap IS high OR overlap IS veryHigh) AND (englishLevel IS medium OR englishLevel IS high OR englishLevel IS veryHigh) THEN modifier IS medium;



	$User_0$	$User_1$	...	$User_N$
$User_0$	-			Weight($User_0$, $User_N$)
$User_1$		-		
...			-	
$User_N$				-

Figure 4.22: C_A matrix



```
<<Java Class>>
CA
com.manjavacas.fence.model
_id: ObjectId
user1: String
user2: String
task: String
project: String
weight: double
CA()
CA(String,String,String,String,double)
getUser1():String
setUser1(String):void
getUser2():String
setUser2(String):void
getTask():String
setTask(String):void
getProject():String
setProject(String):void
getWeight():double
setWeight(double):void
```

Figure 4.23: C_A class

After evaluating the rules, the output produced and conveniently defuzzified by the system will be the value to be added to the original C_R weight. This penalty will attempt to represent the additional communication efforts that must be made between users due to the temporal, geographical and cultural distances between them.

Having studied how this fuzzy inference system works, it will be easily noted how the calculation of the C_A matrix will make use of it in a very similar manner.

ACTUAL COMMUNICATION MATRIX (C_A)

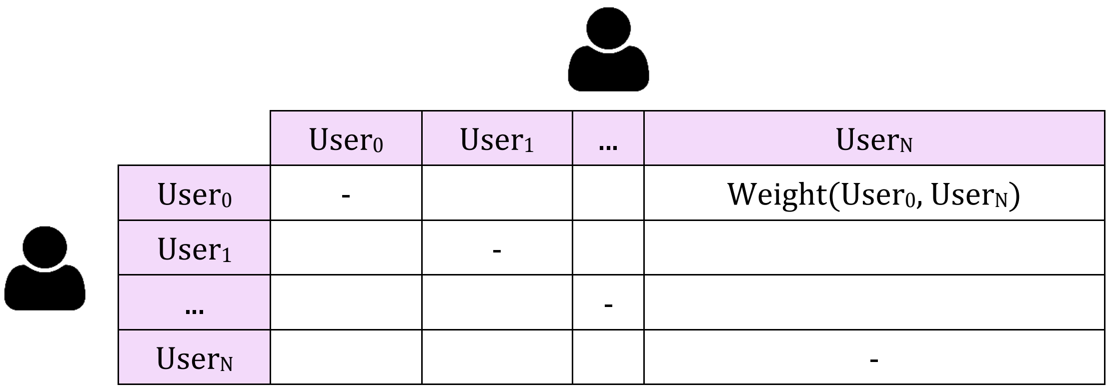
When representing how users are trying to coordinate, it is necessary to take into account the communications between them. The actual coordination matrix (C_A) makes use of the communications within the organization to represent those pairs of users who have communicated, as shown in Figure 4.22. Thus, the different cells of the C_A matrix will represent with their weights how much communication has existed between each pair of users in a given project, being these communications related to concrete tasks (see Figure 4.23).

Before addressing the calculation of the C_A objects stored in the database, it is necessary to clarify how the communications have been obtained to test the algorithm. In order to roughly simulate communications between users in the organization, a Python script was initially used to generate completely random conversations for a given project. However, after multiple iterations it was determined that the generated conversations hardly represented realistic communications, as the number of really useful communications to be generated was simply anecdotal. As a result, the script was consciously modified, giving rise to the one shown in Appendix A, which in this case did provide more realistic and useful results when testing the tool. As can be seen, the script makes use of the existing coordination requirements in C_R matrix and tries to fulfil them with a random amount of conversations between users. These communications would be stored in the database and used by the C_A matrix algorithm detailed in Algorithm 4.

Algorithm 4: C_A calculation

```
input  : project, $C_R$ matrix
output : $C_A$ matrix
caMatrix $\longleftarrow \emptyset$
users $\longleftarrow getUsersC_R()$
foreach user $user_1$ in users do
    $weight_{C_A} \longleftarrow 0$
    $userTasks \longleftarrow getTasksAssignedTo(user_1)$
    $userCommunications \longleftarrow getCommunicationsOf(user_1)$
    foreach communication com in userCommunications do
        $user_2 \longleftarrow getUser_2(com)$
        $numDependencies \longleftarrow totalDependencies(user_1, user_2)$
        if numDependencies > 0 then
            $communications \longleftarrow getCommunicationsBetween(user_1, user_2)$
            $totalCommunications \longleftarrow |communications|$
            $taskCommunications \longleftarrow getTaskCommunications(com.task, communications)$

            if taskCommunications > 0 then
                if taskCommunications $\geq$ 5 then
                    $frequency \longleftarrow 1$
                else if taskCommunications $\geq$ 3 then
                    $frequency \longleftarrow 0.6$
                else
                    $frequency \longleftarrow 0.3$
            end

            $weight_{C_A}$ += $(frequency * totalCommunications / numDependencies)$
            $weight_{C_A}$ −= $computeGlobalFactors(user_1, user_2)$

            if $weight_{C_A}$ > 1 then
                $weight_{C_A} \longleftarrow 1$
            end

            $caMatrix \cup C_A(user_1, user_2, com.task, weight_{C_A}, project)$
        end
    end
end
```

In this case, the fuzzy inference system is used to adjust the weight obtained. As can be noted, the



	$User_0$	$User_1$	...	$User_N$
$User_0$	-			Weight($User_0$, $User_N$)
$User_1$		-		
...			-	
$User_N$				-

Figure 4.24: C_G matrix

penalty for temporal, geographical and cultural distances will result in the decrease of the weight, subtracting part of the value of the established communications.

It can also be observed that the calculation of this matrix is the most time-consuming comparing to the other matrices, and that is because the execution of the algorithm depends on relatively large input data, such as users in C_R, communications and tasks. However, by following this process, we will have calculated the C_A matrix and will be ready to compare the current communication with the existing coordination requirements.

COORDINATION GAPS MATRIX (C_G)

The last matrix to be calculated, which will lead us to compute STC, is the coordination gaps matrix (C_G). This matrix represents in its cells the difference between the communication that each pair of users must achieve and the communication that is actually taking place (see Figure 4.22 and Figure 4.23). In other words, each cell of the C_G array for users $user_i$ and $user_j$ represents the difference in weight between $C_R(user_i, user_j)$ and $C_A(user_i, user_j)$. As can be observed, the weights of the C_G matrix clearly represent the coordination requirements that are being covered as well as those communication gaps that are still present between users.

The procedure followed to fill in this matrix is relatively simple compared to the previous ones, as can be observed in Algorithm 5.

With the C_G matrix representing the lack of coordination between users in a given project, the calculation of the organisation's STC levels will then be undertaken.

SOCIOTECHNICAL CONGRUENCE MEASUREMENT

As we have already indicated, the requirements to be met by FENCE involve the calculation of STC levels from three different points of view: employees, teams, and projects. As we shall see below, knowing the STC level of the organisation's teams and projects is relatively simple, being the obtainment of the employees' STC levels the basis of this calculations.

- **Employees' STC**

To calculate the individual STC of each user we will follow the process detailed in Algorithm 6, where Equation 4.4 will be used to calculate the individual level of STC for each employee.

$$STC = 1 - \frac{\sum_{i=0}^{n} \sum_{j=0}^{n} C_{Gij}}{\sum_{i=0}^{n} \sum_{j=0}^{n} C_{Rij}} \tag{4.4}$$

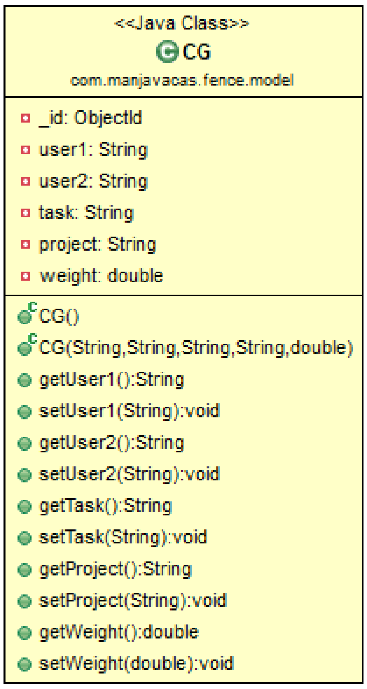


Figure 4.25: C_G class

Algorithm 5: C_G calculation

```
input  : $C_A$ matrix, $C_R$ matrix
output : $C_G$ matrix
$cgMatrix \longleftarrow \emptyset$
foreach $cr$ in $crMatrix$ do
    $caList \longleftarrow getCA(cr.user_1, cr.user_2, cr.project, cr.task)$
    if $caList \neq \emptyset$ then
        $weight_{C_A} \longleftarrow sumWeights(caList)$
        $weight_{C_R} \longleftarrow cr.weight$
        $weight_{C_G} \longleftarrow weight_{C_R} - weight_{C_A}$
        if $weight_{C_G} > 0$ then
            $cgMatrix \cup C_G(cr.user_1, cr.user_2, cr.task, weight_{C_G}, cr.project)$
        end
    else
        $cgMatrix \cup C_G(cr.user_1, cr.user_2, cr.task, cr.weight, cr.project)$
    end
end
```

Algorithm 6: User STC calculation

```
input  : project, $C_R$ Matrix, $C_G$ matrix
output : users STC
1  $stcUsers \longleftarrow \emptyset$
2  $users \longleftarrow getProjectUsers(project)$
3  foreach $user$ in $users$ do
4      $userCRs \longleftarrow getUserCRs(user)$
5      $userCGs \longleftarrow getUserCGs(user)$
6      $userCRsum \longleftarrow sumWeights(userCRs)$
7      $userCGsum \longleftarrow sumWeights(userCGs)$
8      if $userCGsum \leq 0$ then
9          $stcUsers \cup UserSTC(user, 100.0, getCurrentDate())$
10     else
11         $stcUser \longleftarrow 1 - (userCGsum/userCRsum)$
12         $stcUsers \cup UserSTC(user, stcUser * 100, getCurrentDate())$
13     end
14 end
```

At the same time, note how each measurement will be dated. This will allow us to have a history of the STC levels for each user, team, and project, whose usefulness will be seen in the next module.

- **Teams' STC**

In this case, as shown in Algorithm 7, the calculation of the STC at team level is very similar to the previous one, except that the users considered are those belonging to the chosen team.

As in the previous case, we will keep a history of the STC measurements for teams.

- **Project STC**

Having calculated the STC at user and team level, it only remains to measure it at project level by following the process detailed in Algorithm 8.

Once seen in detail the implementation of the algorithms and their adaptation to the information handled by FENCE, it only remains to illustrate how the information is requested by the user and displayed in the interface (see Figure 4.26).

- Initially, when the user accesses the STC tab, the last measurements made by the tool will be loaded into the different tables: users, teams, and project tables.
- If the user wants to recalculate the STC levels for a given project, he or she must select it and directly click on the "Calculate STC" button.
- This will cause the STC levels to be recalculated, along with the values from the different matrices, being also updated in the database.
- Finally, once this process is completed, the tables will be filled with the most updated version of the STC levels.

As can be noted, new resources such as classes to model STC measurements and matrices were included and implemented in this phase, while others previously defined were expanded or adjusted to the needs of the STC meter.

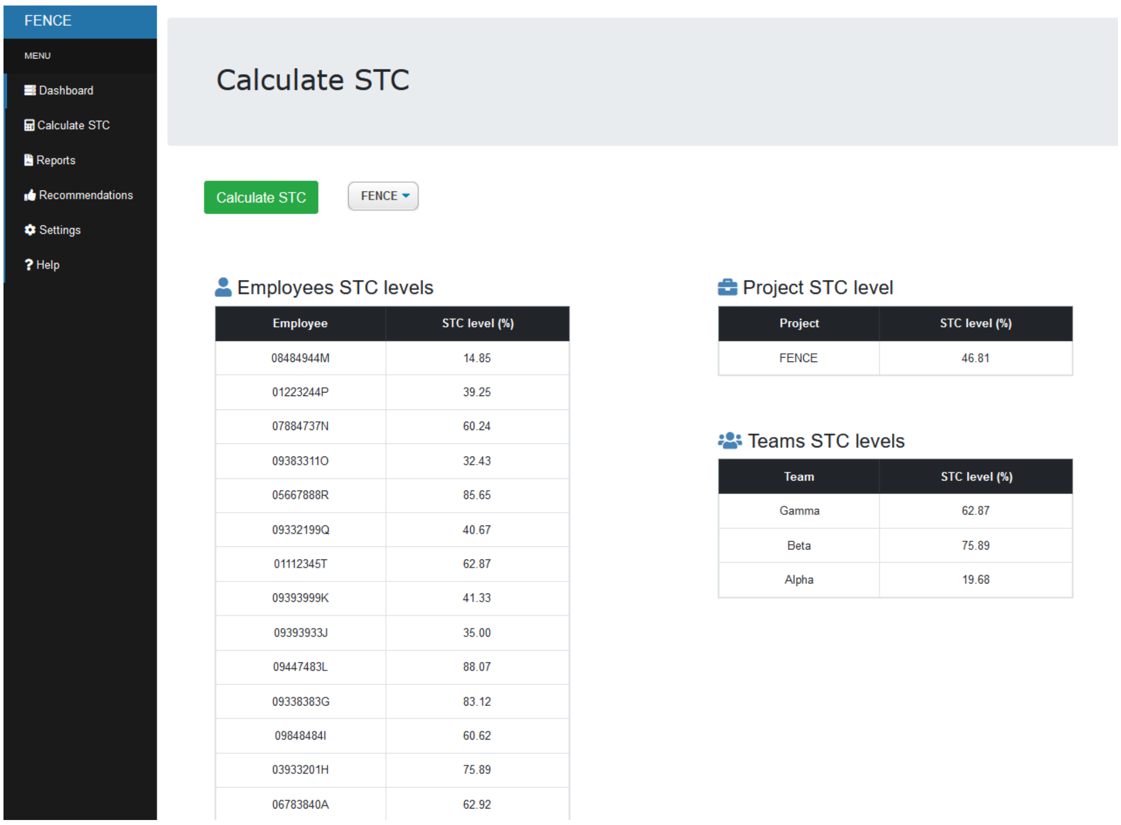
It also should be considered that the entire execution of the STC calculation algorithm is quite computational costly. Although this complexity was iteratively reduced, resulting in significantly better performance, the amount of time required for execution remained considerable given the complexity of the calculations to be made and the number of inputs (users, tasks, assignments,

Algorithm 7: Teams STC calculation

```
input  : project, $C_R$ Matrix, $C_G$ matrix
output : teams STC
1  $stcTeams \longleftarrow \emptyset$
2  $teams \longleftarrow getProjectTeams(project)$
3  foreach $team$ in $teams$ do
4      $teamCRsum \longleftarrow 0$
5      $teamCGsum \longleftarrow 0$
6      $teamUsers \longleftarrow team.users$
7      foreach $user$ in $teamUsers$ do
8          $userCRs \longleftarrow getUserCRs(user)$
9          $userCGs \longleftarrow getUserCGs(user)$
10         $teamCRsum$ += $sumWeights(userCRs)$
11         $teamCGsum$ += $sumWeights(userCGs)$
12     end
13     if $teamCGsum \leq 0$ then
14         $stcTeams \cup TeamSTC(team, 100.0, getCurrentDate())$
15     else
16         $stcTeam \longleftarrow 1 - (teamCGsum/teamCRsum)$
17         $stcTeams \cup TeamSTC(team, stcTeam * 100, getCurrentDate())$
18     end
19 end
```

Algorithm 8: Project STC calculation

```
input  : project, $C_R$ Matrix, $C_G$ matrix
output : project STC
1  $projectCRsum \longleftarrow sumWeights(C_R)$
2  $projectCGsum \longleftarrow sumWeights(C_G)$
3  if $projectCGsum \leq 0$ then
4      $stcProject \cup ProjectsSTC(team, 100.0, getCurrentDate())$
5  else
6      $stcProject \longleftarrow 1 - (projectCGsum/projectCRsum)$
7      $stcProject \cup ProjectSTC(project, stcProject * 100, getCurrentDate())$
8  end
```



Employees STC levels

Employee	STC level (%)
08484944M	14.85
01223244P	39.25
07884737N	60.24
09383311O	32.43
05667888R	85.65
09332199Q	40.67
01112345T	62.87
09393999K	41.33
09393933J	35.00
09447483L	88.07
09338383G	83.12
09848484I	60.62
03933201H	75.89
06783840A	62.92

Project STC level

Project	STC level (%)
FENCE	46.81

Teams STC levels

Team	STC level (%)
Gamma	62.87
Beta	75.89
Alpha	19.68

Figure 4.26: STC measurer interface

dependencies, communications) to be handled.

Nevertheless, the results provided by the system developed in this module would serve as a basis for the development and use of the following modules, focused on visualizing and improving the organization's STC levels.

4.3.4. Module 4. Data visualization

The objective of the work carried out in this module was to allow the visualization of time series with information on STC measurements in employees, teams, and projects. Based on this, the implementation of the graphics was carried out using the JavaScript library *Chart.js*.

This library offers a flexible and simple way to generate many types of charts. In this case, the input data would be displayed in a timeline, corresponding to the latest STC measurements for each employee, team, and project.

Looking at the FENCE interface, as can be seen in Figure 4.27, the user will find three canvas with which he or she can interact by selecting at convenience the specific employee, team or project from which it is desired to know its STC history levels.

If we go deeper into the construction and operation of the charts beyond the interface, initially the application makes GET-type Ajax requests to the corresponding controllers. Each controller shown in Figure 4.28 receives the request and invokes the corresponding method in the associated service object (see Figure 4.29), which retrieves and returns all the measurements stored in the database. When each Ajax request receives the result of the query, the last 20 measurements performed are extracted and entered as input data to be represented by the *Chart.js* plot.

This is a fast and simple process, which provides FENCE with more analytical capabilities for the analysis of STC levels evolution. Visualising this progression can help project managers, or even

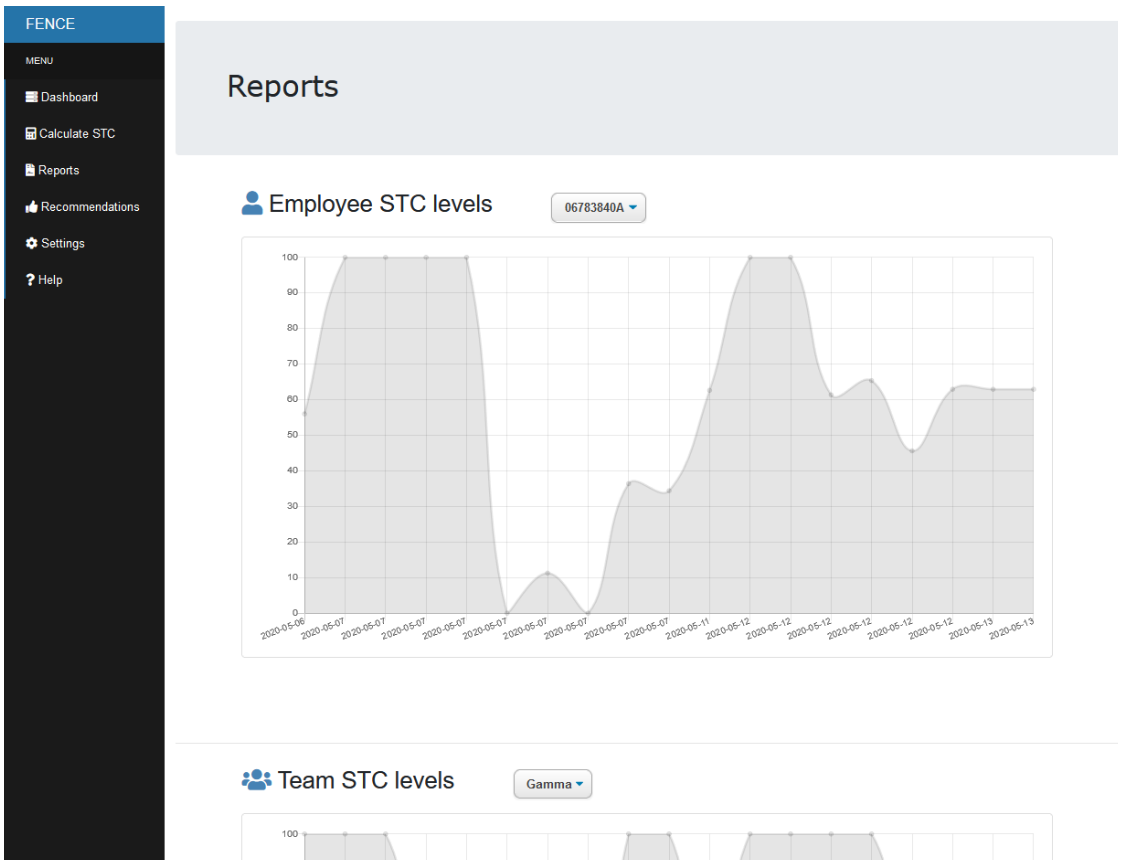
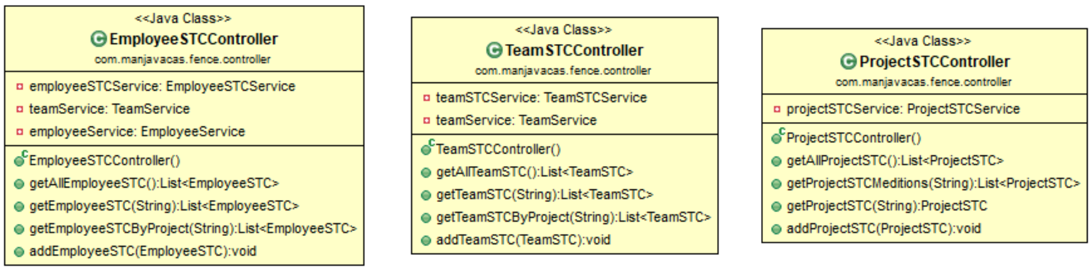


Figure 4.27: Reports interface



<<Java Class>>
EmployeeSTCController
com.manjavacas.fence.controller

- employeeSTCService: EmployeeSTCService
- teamService: TeamService
- employeeService: EmployeeService

- EmployeeSTCController()
- getAllEmployeeSTC():List<EmployeeSTC>
- getEmployeeSTC(String):List<EmployeeSTC>
- getEmployeeSTCByProject(String):List<EmployeeSTC>
- addEmployeeSTC(EmployeeSTC):void

<<Java Class>>
TeamSTCController
com.manjavacas.fence.controller

- teamSTCService: TeamSTCService
- teamService: TeamService

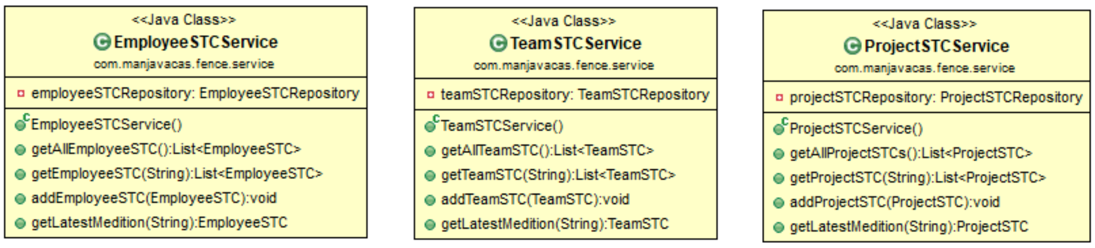
- TeamSTCController()
- getAllTeamSTC():List<TeamSTC>
- getTeamSTC(String):List<TeamSTC>
- getTeamSTCByProject(String):List<TeamSTC>
- addTeamSTC(TeamSTC):void

<<Java Class>>
ProjectSTCController
com.manjavacas.fence.controller

- projectSTCService: ProjectSTCService

- ProjectSTCController()
- getAllProjectSTC():List<ProjectSTC>
- getProjectSTCMeditions(String):List<ProjectSTC>
- getProjectSTC(String):ProjectSTC
- addProjectSTC(ProjectSTC):void

Figure 4.28: STC measurements controllers



<<Java Class>>
EmployeeSTCService
com.manjavacas.fence.service

- employeeSTCRepository: EmployeeSTCRepository

- EmployeeSTCService()
- getAllEmployeeSTC():List<EmployeeSTC>
- getEmployeeSTC(String):List<EmployeeSTC>
- addEmployeeSTC(EmployeeSTC):void
- getLatestMedition(String):EmployeeSTC

<<Java Class>>
TeamSTCService
com.manjavacas.fence.service

- teamSTCRepository: TeamSTCRepository

- TeamSTCService()
- getAllTeamSTC():List<TeamSTC>
- getTeamSTC(String):List<TeamSTC>
- addTeamSTC(TeamSTC):void
- getLatestMedition(String):TeamSTC

<<Java Class>>
ProjectSTCService
com.manjavacas.fence.service

- projectSTCRepository: ProjectSTCRepository

- ProjectSTCService()
- getAllProjectSTCs():List<ProjectSTC>
- getProjectSTC(String):List<ProjectSTC>
- addProjectSTC(ProjectSTC):void
- getLatestMedition(String):ProjectSTC

Figure 4.29: STC measurements services

strategic leaders, to consider what issues are affecting the performance of the global projects.

Once this phase based on data visualisation has been overcome, the next module would allow to undertake concrete actions to improve the STC levels within the organisation.

4.3.5. Module 5. Recommendation system

This module addressed the development of one of the main and most distinctive functionalities of FENCE: the development of a recommendation system focused on providing information to the user about how to improve the STC levels of the organization.

Before starting the development of this system, meetings were held with the different stakeholders:

TECHNICAL PLANNING

Initially, it was discussed through different meetings with José Ángel Olivas how to approach the development of the recommendation system by using the information provided by FENCE. It was agreed that the coordination gap matrix (C_G) was ideal for finding out which users were those on whom recommendations should be made. Moreover, the variables and rules to be used by the fuzzy inference system were also studied and initially outlined.

After these sessions, multiple alternatives were considered when working with fuzzy logic in Java, being finally jFuzzyLogic [8, 9] the library chosen for this task. This library is based on the definition of fuzzy inference systems by means of the standard FCL [19], so it was necessary to learn its syntax in the handling of variables, functions, fuzzy sets, and rules.

The main features of this library that led to its use were: its flexibility and ease of integration with the system developed so far; the use of FCL (a relatively simple fuzzy definition language) as well as being quite updated in relation to other fuzzy logic libraries found.

With this technical base and once familiar with jFuzzyLogic after multiple tests, the Knowledge Engineering work was carried out together with the Aurora Vizcaíno, who played the role of expert.

KNOWLEDGE ENGINEERING

Knowledge Engineering involves the extraction of expert knowledge and its translation into a formalized language in a precise, unambiguous, and exploitable way. On this basis, the Knowledge Engineering tasks performed would allow us to define the features that should be covered by the recommending system.

If we consider its scope and limitations, the following ones were considered:

- **Scope**: given the characteristics of the users that integrate the multiple teams on a given project, as well as the organization's STC levels, the system will offer as output recommendations to improve those STC levels in a clear and traceable way.
- **Limitations**: the number of features covered will be strictly limited to the knowledge of the expert. The system shall also not contemplate specific features of the project, teams and users going beyond the information provided as input.

Subsequently, once the domain of the system had been established, the choice of the expert willing to transmit her knowledge was made. In this case, as previously stated, it was Aurora Vizcaíno, university professor and researcher in the Computer Science School of Ciudad Real. It was considered that her contributions would be an important source of knowledge given her experience in the field of GSD, as well as her work in multiple research projects related to communication and knowledge management in this domain.

Table 4.8: Plausability study

Plausibility					
Category	**Identifier**	**Weight**	**Value**	**Feature definition**	**Type**
EX	P1	10	8	Experts exist	E
EX	P2	10	9	The assigned expert is genuine	E
EX	P3	8	10	The expert is cooperative	D
EX	P4	7	9	The expert is able to articulate his/her methods, but does not categorize	D
TA	P5	10	8	There are enough test cases	E
TA	P6	10	9	The task is well structured and understood	D
TA	P7	10	8	It only requires cognitive ability	D
TA	P8	9	8	No optimal results are required but only satisfactory ones, without compromising the project	D
TA	P9	9	7	The task does not require common sense	D
DU	P10	7	10	The managers are truly committed to the project	D

On the other hand, one of the main tasks of the knowledge engineer is the study of the viability of the project to be undertaken. One of the most recognised ways of studying this viability is the Slagel Test [32]. This test is focused on the study of the plausibility, justification, success, adequacy, and success of expert systems, allowing to determine if the project is viable based on the reasoning over its characteristics:

- **Plausibility**: determines whether the necessary means to address the problem from the perspective of Knowledge Engineering are available. From this point of view, the characteristics of the expert and the task to be undertaken are analysed.
- **Justification**: analysis of the need for experience and the investment to be made (type of environment, need due to volatile experience, return of investment, alternative solutions, etc.).
- **Adequacy**: it is studied if the problem is adequate to be solved with Knowledge Engineering techniques, analysing its nature, complexity, and type.
- **Success**: the probabilities of the system's success are determined in advance. It involves studying non-technical but important issues in order to decide to apply Knowledge Engineering to the resolution of the problem, such as the involvement and responsibility of the people implicated, as well as the acceptance of the system in its environment.

In this way, the process conducted in order to justify the viability of the recommending system to be developed is shown below:

- First, for each of the dimensions mentioned, three different categories are established:
  - Managers and/or users (MU).
  - Experts (EX).
  - Tasks (TA).

  For each category, the most important features that define that dimension are specified. These features may be essential (E) or desirable (D), considering that essential features must not have a value below a threshold (7 in this case).
  On this basis, each feature is assigned a weight between 0 and 10, depending on its relative importance. This entire process is summarized in tables Table 4.8, Table 4.9, Table 4.10, and Table 4.11.
- Once the values have been assigned, we proceed as follows:
  - If the value of an essential feature does not reach the required threshold, it is counted as 0 and the application is rejected.
  - Otherwise, we multiply the values of each feature by its corresponding weight to obtain their pondered values.
  - We then multiply these pondered values for each dimension.
  - Finally, we compute the geometric pseudo-mean of the weighted values for each dimen-

Table 4.9: Justification study

Justification					
Category	**Identifier**	**Weight**	**Value**	**Feature definition**	**Type**
EX	J1	10	9	The expert is not available	E
EX	J2	10	9	There's a shortage of human experience	D
TA	J3	8	9	There is a need for simultaneous experience in many places	D
TA	J4	10	8	Need for experience in hostile, distressing and/or unrewarding environments	E
TA	J5	8	9	There are no admissible alternative solutions	E
MU	J6	7	9	A high rate of return on investment is expected	D
MU	J7	8	9	Solves a useful and necessary task	E

Table 4.10: Adequacy study

Adequacy					
Category	**Identifier**	**Weight**	**Value**	**Feature definition**	**Type**
EX	A1	5	7	The expert's experience is poorly organized	D
TA	A2	6	7	It has little practical value	D
TA	A3	7	7	It is a more tactical than strategic task	D
TA	A4	7	9	The task provides solutions that serve long-term needs	E
TA	A5	5	9	The task is not too easy, but it is knowledge-intensive, both in the domain and in the manipulation of information	D
TA	A6	6	9	It is manageable in size, and/or a gradual approach is possible and/or, a breakdown into independent sub-tasks	D
EX	A7	7	10	The transfer of experience between humans is feasible	E
TA	A8	6	9	It was identified as a problem in the area and the effects of introducing an expert system can be planned	D
TA	A9	9	10	No "immediate" real-time answers required	E
TA	A10	9	8	The task does not require basic research	E
TA	A11	5	7	The expert basically uses symbolic reasoning involving subjective factors	D
TA	A12	5	9	It is essentially heuristic	D

Table 4.11: Success study

Success					
Category	**Identifier**	**Weight**	**Value**	**Feature definition**	**Type**
EX	E1	8	10	Users do not feel threatened by the project, they are able to feel intellectually attached to the project	D
EX	E2	6	9	Users have a remarkable track record in performing this task	D
EX	E3	5	9	There are agreements on what constitutes a good solution to the task	D
EX	E4	5	9	The only justification for taking a step in the solution is the quality of the final solution	D
EX	E5	6	8	There is no strict deadline, nor does any other project depend on this task	D
TA	E6	7	10	It is not influenced by politics	E
TA	E7	8	7	Expert systems already exist to solve this or similar tasks	D
TA	E8	8	8	There are minimal changes to standard procedures	D
TA	E9	5	9	Solutions are explainable or interactive	D
TA	E10	7	8	The task is one of practical R&D, but not both simultaneously	E
MU	E11	6	9	Users are mentally sound and have realistic expectations of both the scope and limitations	D
MU	E12	7	9	They do not reject this technology out of hand	E
MU	E13	6	9	The system interacts intelligently and friendly with the user	D
MU	E14	9	8	The system is able to explain to the user its reasoning	D
MU	E15	8	9	The insertion of the system takes place without injuries; it hardly interferes with the daily routine of the company	D
MU	E16	6	9	Users are committed for the entire duration of the project, even after its implementation	D
MU	E17	8	9	Adequate technology transfer takes place	E

Table 4.12: Viability of the system

Category	Rate	Mean viability rate	Normalized viability rate
VC1	76,01	$VC = 65.6$	$VC_{norm} = 86.18$ %
VC2	76,43		
VC3	52,37		
VC4	57,57		

sion, as shown in Equation 4.5, Equation 4.6, Equation 4.7, and Equation 4.8.

$$VC_1 = \prod_{i=1,2,5} \left| \frac{V_{pi}}{V_{ui}} \right| (\prod_{i=1}^{10} P_{pi} * V_{pi})^{\frac{1}{10}} = 76.01 \tag{4.5}$$

$$VC_2 = \prod_{i=1,4,5,7} \left| \frac{V_{ji}}{V_{ui}} \right| (\prod_{i=1}^{10} P_{ji} * V_{ji})^{\frac{1}{7}} = 76.43 \tag{4.6}$$

$$VC_3 = \prod_{i=4,7,9,10} \left| \frac{V_{ai}}{V_{ui}} \right| (\prod_{i=1}^{10} P_{ai} * V_{ai})^{\frac{1}{12}} = 52.37 \tag{4.7}$$

$$VC_4 = \prod_{i=6,10,12,17} \left| \frac{V_{pi}}{V_{ui}} \right| (\prod_{i=1}^{17} P_{ei} * V_{ei})^{\frac{1}{17}} = 57.57 \tag{4.8}$$

- Finally, we compute the normalized mean of the weights of all the dimensions, as shown in Equation 4.9 and Equation 4.10, resulting in a percentage that reflects the viability of the system (Table 4.12):

$$VC = \frac{VC_1 + VC_2 + VC_3 + VC_4}{4} = \frac{76.01 + 76.43 + 52.37 + 57.57}{4} = 65.6 \tag{4.9}$$

$$VC_{norm} = \frac{VC * 100}{Max.VC} = \frac{65.6 * 100}{76.1237} = 86.18 \tag{4.10}$$

With a viability rate of 86.18%, the proposal was considered viable and, therefore, its development was undertaken.

The next step was the acquiring of expert knowledge, whose extraction was carried out by means of structured interviews agreed upon with the expert. In these meetings, the solutions that the system should provide to different types of situations presented by the interviewer were discussed, as well as the different variables and values that would be translated into fuzzy system inputs.

A total of three meetings made it possible to refine the knowledge obtained, and thus establish the basis for the recommendation system to be developed.

EXPERT SYSTEM DEVELOPMENT

With the expert knowledge formalized and translated into rules, the implementation of the system was addressed. First, we will see the variables considered and their corresponding fuzzy sets:

a) Input variables

- **Cultural distance**. Average of the distances between socio-cultural factors identified by [16] according to the nationality of the users. Calculated as already shown in the C_R matrix explanation of section 4.3.3, with the same fuzzy sets.

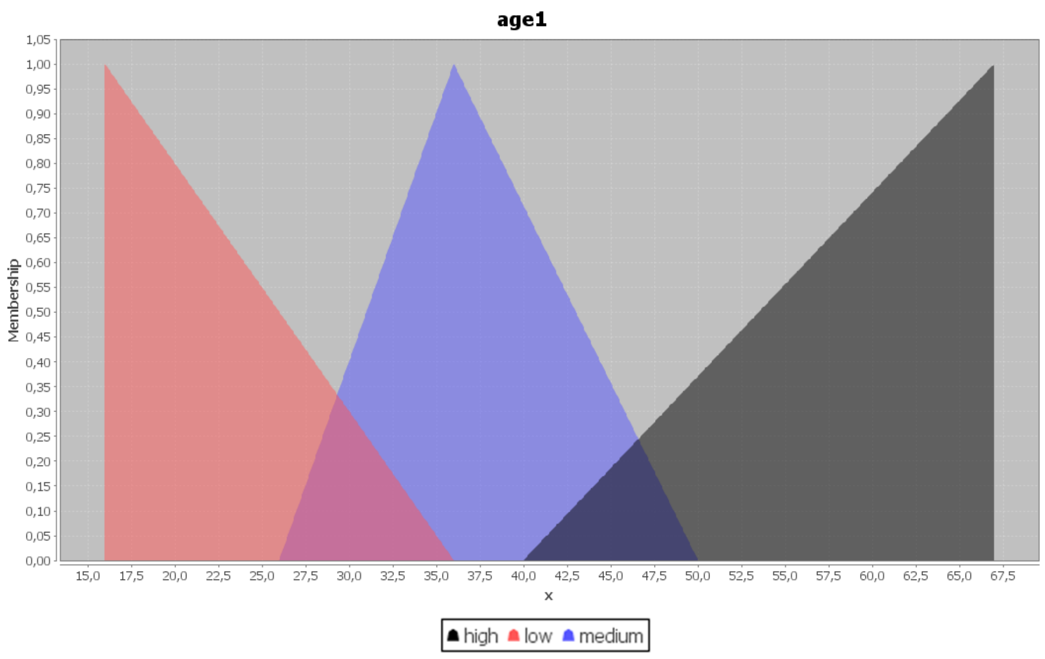


Figure 4.30: Age fuzzy sets

- **Time overlap**. Coincidence between employees' time zones (UTC), calculated as the inverse of their time difference. A greater time overlap will facilitate the use of synchronous means of communication, while the impossibility of temporal coincidence in communication will result in greater use of asynchronous means. As in the previous case, this input parameter was already defined in section 4.3.3.
- **English level**. Level of English shared by users. The method for obtaining this value and its fuzzy sets was also defined in section 4.3.3.
- **Age1**, **Age2**. Age of the evaluated employees, represented by the fuzzy sets in Figure 4.30.
- **Experience1**, **Experience2**. Individual user work experience, whose levels are depicted in Figure 4.31.

b) Output variables

- **Mediator**. Indicates whether a mediator is needed if there is a high cultural distance between users. It is a singleton set with singleton centre of gravity (COGS) as defuzzification method, as shown in Figure 4.32.
- **Training**. Represents the need for cultural training when the cultural distance between individuals is medium or moderate. This is another singleton set (see Figure 4.33) with COGS as a defuzzification method.
- **Supervisor1**, **Supervisor2**. They represent the need for supervisors for inexperienced users. This need is represented by singleton sets and COGS as defuzzification method (see Figure 4.34).
- **Solution1**, **Solution2**, ..., **Solution9**. These variables cover the 9 solutions proposed to reduce communication problems. Each solution shows a series of recommendations and communication means related to user input values. All of them are represented by singleton sets with COGS as defuzzification method (see Figure 4.35). Also note how the proposed communication solution offered as output will be always unique.

With the input and output variables defined, using jFuzzyLogic for the implementation was not difficult after learning how to use it together with FCL. In general, the system is divided into four distinct parts:

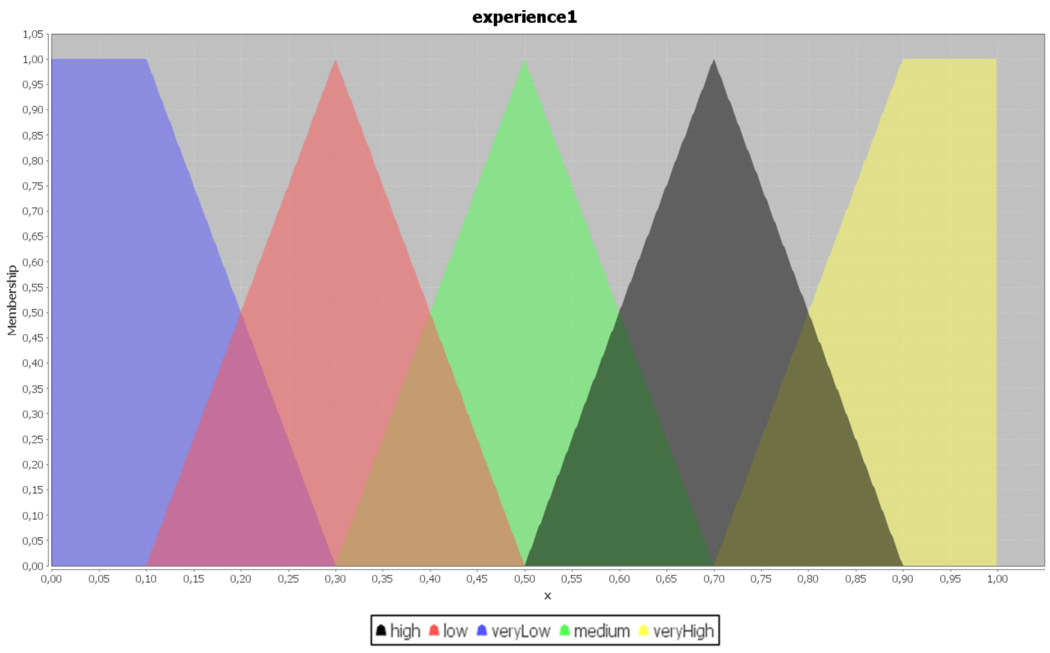


Figure 4.31: Experience fuzzy sets

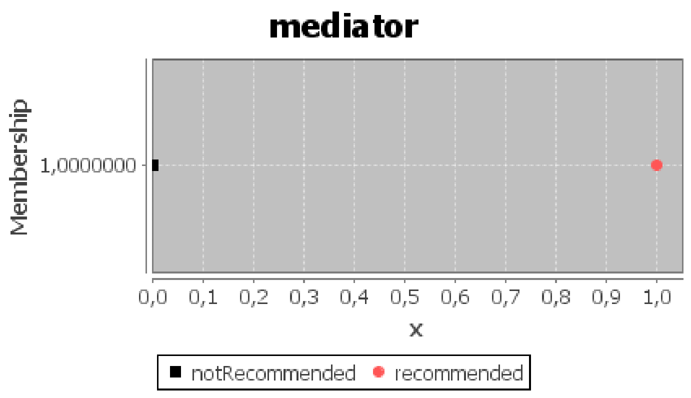


Figure 4.32: Mediator singleton fuzzy sets

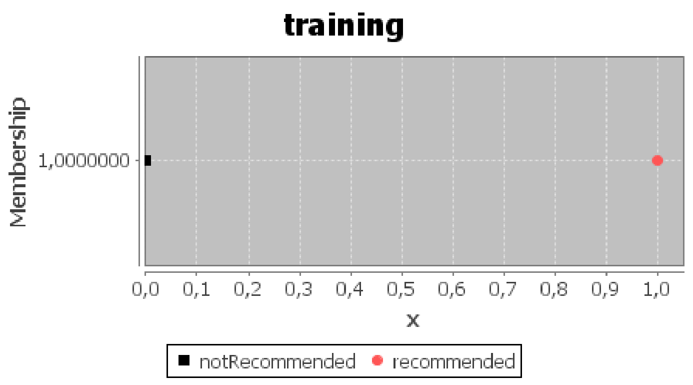


Figure 4.33: Training singleton fuzzy sets

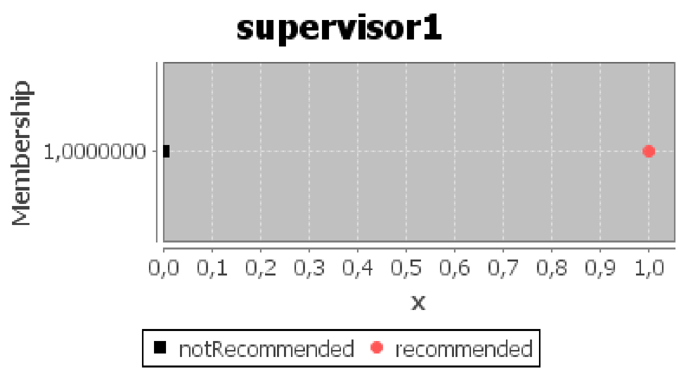


Figure 4.34: Supervisor singleton fuzzy sets

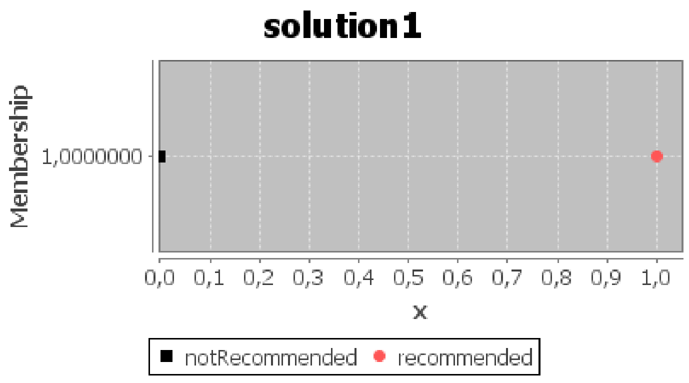


Figure 4.35: Solutions singleton fuzzy sets

- First, we have the variable declaration block, as shown in Listing 4.1, where we indicate the input and output variables of the system.

Listing 4.1: Variables declaration

```
VAR_INPUT
  culturalDist : REAL;
  overlap : REAL;
  age1 : REAL;
  age2 : REAL;
  englishLevel : REAL;
  experience1 : REAL;
  experience2 : REAL;
END_VAR

VAR_OUTPUT
  mediator : REAL;
  training : REAL;
  supervisor1 : REAL;
  supervisor2 : REAL;
  solution1 : REAL;
  solution2 : REAL;
  solution3 : REAL;
  solution4 : REAL;
  solution5 : REAL;
  solution6 : REAL;
  solution7 : REAL;
  solution8 : REAL;
  solution9 : REAL;
END_VAR
```

- Next is the definition block for fuzzy sets, or fuzzification block, where the input values are translated into fuzzy labels (Listing 4.2).

Listing 4.2: Fuzzification block

```
FUZZIFY culturalDist // Cultural distance
  TERM veryLow := trape 0 0 0.1 0.3;
  TERM low := trian 0.1 0.3 0.5;
  TERM medium := trian 0.3 0.5 0.7;
  TERM high := trian 0.5 0.7 0.9;
  TERM veryHigh := trape 0.7 0.9 1 1;
END_FUZZIFY

FUZZIFY overlap // Time overlap
  TERM veryLow := trape 0 0 0.1 0.3;
  TERM low := trian 0.1 0.3 0.5;
  TERM medium := trian 0.3 0.5 0.7;
  TERM high := trian 0.5 0.7 0.9;
  TERM veryHigh := trape 0.7 0.9 1 1;
END_FUZZIFY

FUZZIFY age1 // User1's age
  TERM low := trian 16 16 36;
  TERM medium := trian 26 36 50;
  TERM high := trian 40 67 67;
END_FUZZIFY

FUZZIFY age2 // User2's age
  TERM low := trian 16 16 36;
  TERM medium := trian 26 36 50;
  TERM high := trian 40 67 67;
END_FUZZIFY

FUZZIFY englishLevel // English level
  TERM veryLow := trape 0 0 0.1 0.3;
  TERM low := trian 0.1 0.3 0.5;
  TERM medium := trian 0.3 0.5 0.7;
  TERM high := trian 0.5 0.7 0.9;
  TERM veryHigh := trape 0.7 0.9 1 1;
END_FUZZIFY

FUZZIFY experience1 // Experience of user1
  TERM veryLow := trape 0 0 0.1 0.3;
  TERM low := trian 0.1 0.3 0.5;
  TERM medium := trian 0.3 0.5 0.7;
  TERM high := trian 0.5 0.7 0.9;
  TERM veryHigh := trape 0.7 0.9 1 1;
END_FUZZIFY

FUZZIFY experience2 // Experience of user2
  TERM veryLow := trape 0 0 0.1 0.3;
  TERM low := trian 0.1 0.3 0.5;
  TERM medium := trian 0.3 0.5 0.7;
  TERM high := trian 0.5 0.7 0.9;
  TERM veryHigh := trape 0.7 0.9 1 1;
END_FUZZIFY
```

- In the same way, the output values are defined in the defuzzification block (Listing 4.3), following an inverse process to the preceding one.

Listing 4.3: Defuzzification block

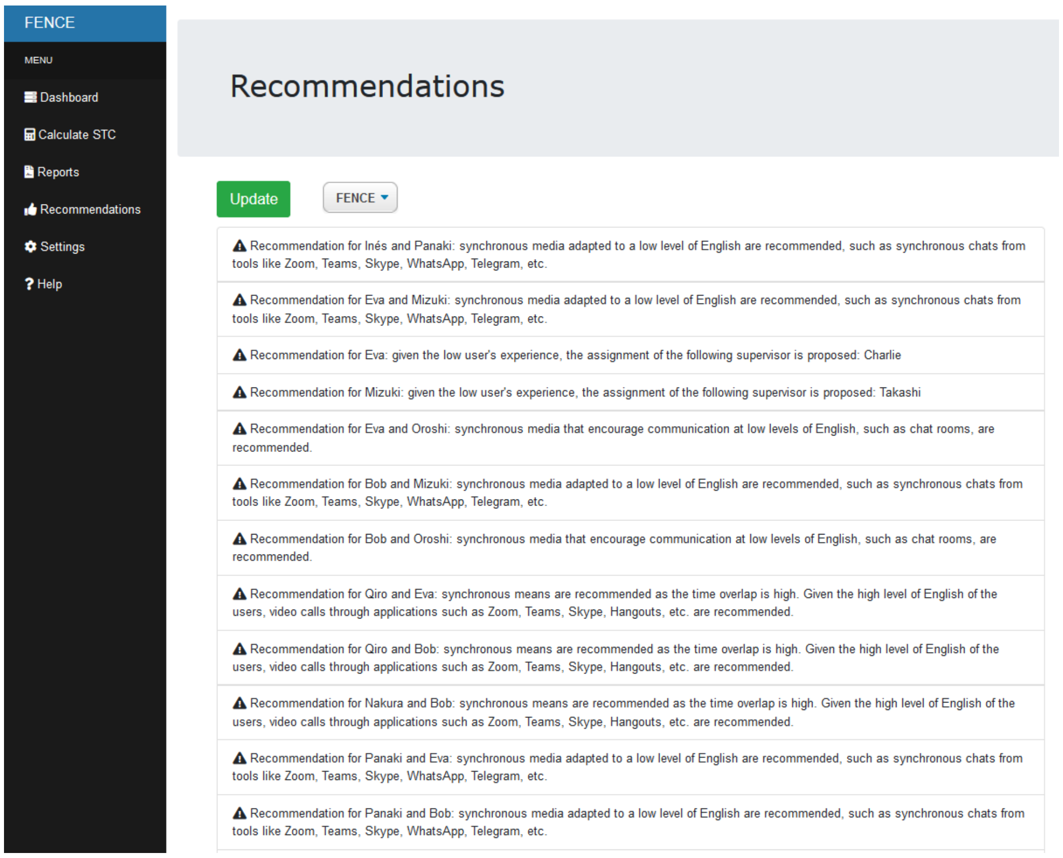
```
DEFUZZIFY mediator // Communication mediator
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY

DEFUZZIFY training // Communication training
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY

DEFUZZIFY supervisor1 // Supervisor for user1
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY

DEFUZZIFY supervisor2 // Supervisor for user2
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY

DEFUZZIFY solution1
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY

DEFUZZIFY solution2
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY

DEFUZZIFY solution3
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY

DEFUZZIFY solution4
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY

DEFUZZIFY solution5
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY
```

```

DEFUZZIFY solution6
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY

DEFUZZIFY solution7
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY

DEFUZZIFY solution8
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY

DEFUZZIFY solution9
  TERM notRecommended := 0;
  TERM recommended := 1;
  METHOD : COGS;
  DEFAULT := 0;
END_DEFUZZIFY
```

- As we have already defined the input and output variables, as well as the fuzzification and defuzzification functions, all that remains is to display the set of rules used by the system, shown in Table 4.13.

Table 4.13: Recommender system rules set

N	FCL RULE
1	IF (age1 IS high OR age2 IS high) AND (overlap IS low OR overlap IS veryLow) AND (englishLevel is low OR englishLevel is veryLow) THEN solution1 IS recommended;
2	IF (age1 IS high OR age2 IS high) AND (overlap IS low OR overlap IS veryLow) AND (englishLevel is medium) THEN solution2 IS recommended;
3	IF (age1 IS high OR age2 IS high) AND (overlap IS low OR overlap IS veryLow) AND (englishLevel is high OR englishLevel is veryHigh) THEN solution3 IS recommended;
4	IF (age1 IS high OR age2 IS high) AND (overlap IS medium OR overlap IS high OR overlap is veryHigh) AND (englishLevel IS veryHigh OR englishLevel IS high OR englishLevel IS medium) THEN solution4 IS recommended;
5	IF (age1 IS high OR age2 IS high) AND (overlap IS medium OR overlap IS high OR overlap is veryHigh) AND (englishLevel IS veryLow OR englishLevel IS low) THEN solution5 IS recommended;
6	IF (age1 IS low OR age1 IS medium) AND (age2 IS low OR age2 IS medium) AND (overlap IS low OR overlap IS veryLow) AND (englishLevel IS medium OR englishLevel IS high OR englishLevel IS veryHigh) THEN solution6 IS recommended;
7	IF (age1 IS low OR age1 IS medium) AND (age2 IS low OR age2 IS medium) AND (overlap IS low OR overlap IS veryLow) AND (englishLevel IS low OR englishLevel IS veryLow) THEN solution7 IS recommended;
8	IF (age1 IS low OR age1 IS medium) AND (age2 IS low OR age2 IS medium) AND (overlap IS medium OR overlap IS high OR overlap IS veryHigh) AND (englishLevel IS veryHigh OR englishLevel IS high OR englishLevel IS medium) THEN solution8 IS recommended;
9	IF (age1 IS low OR age1 IS medium) AND (age2 IS low OR age2 IS medium) AND (overlap IS medium OR overlap IS high OR overlap IS veryHigh) AND (englishLevel IS veryLow OR englishLevel IS low) THEN solution9 IS recommended;
10	IF (culturalDist IS veryHigh) THEN mediator IS recommended;
11	IF (culturalDist IS high) THEN training IS recommended;
12	IF (experience1 IS low OR experience1 IS veryLow) THEN supervisor1 IS recommended;
13	IF (experience2 IS low OR experience2 IS veryLow) THEN supervisor2 IS recommended;



Recommendation for Inés and Panaki: synchronous media adapted to a low level of English are recommended, such as synchronous chats from tools like Zoom, Teams, Skype, WhatsApp, Telegram, etc.

Recommendation for Eva and Mizuki: synchronous media adapted to a low level of English are recommended, such as synchronous chats from tools like Zoom, Teams, Skype, WhatsApp, Telegram, etc.

Recommendation for Eva: given the low user's experience, the assignment of the following supervisor is proposed: Charlie

Recommendation for Mizuki: given the low user's experience, the assignment of the following supervisor is proposed: Takashi

Recommendation for Eva and Oroshi: synchronous media that encourage communication at low levels of English, such as chat rooms, are recommended.

Recommendation for Bob and Mizuki: synchronous media adapted to a low level of English are recommended, such as synchronous chats from tools like Zoom, Teams, Skype, WhatsApp, Telegram, etc.

Recommendation for Bob and Oroshi: synchronous media that encourage communication at low levels of English, such as chat rooms, are recommended.

Recommendation for Qiro and Eva: synchronous means are recommended as the time overlap is high. Given the high level of English of the users, video calls through applications such as Zoom, Teams, Skype, Hangouts, etc. are recommended.

Recommendation for Qiro and Bob: synchronous means are recommended as the time overlap is high. Given the high level of English of the users, video calls through applications such as Zoom, Teams, Skype, Hangouts, etc. are recommended.

Recommendation for Nakura and Bob: synchronous means are recommended as the time overlap is high. Given the high level of English of the users, video calls through applications such as Zoom, Teams, Skype, Hangouts, etc. are recommended.

Recommendation for Panaki and Eva: synchronous media adapted to a low level of English are recommended, such as synchronous chats from tools like Zoom, Teams, Skype, WhatsApp, Telegram, etc.

Recommendation for Panaki and Bob: synchronous media adapted to a low level of English are recommended, such as synchronous chats from tools like Zoom, Teams, Skype, WhatsApp, Telegram, etc.

Figure 4.36: Recommendations window

Once the system was implemented and ready to be used, it was integrated into the FENCE interface.

EXPERT SYSTEM INTERFACE

The integration of the expert system with the FENCE interface is quite simple and easy to use. When the user accesses the "Recommendations" window (shown in Figure 4.36), the first thing that will be loaded will be the last recommendations generated for the project selected in the dropdown menu. However, if the user wants to recalculate these recommendations to get an up-to-date list, then simply pressing the "Update" button will display the new list of advices.

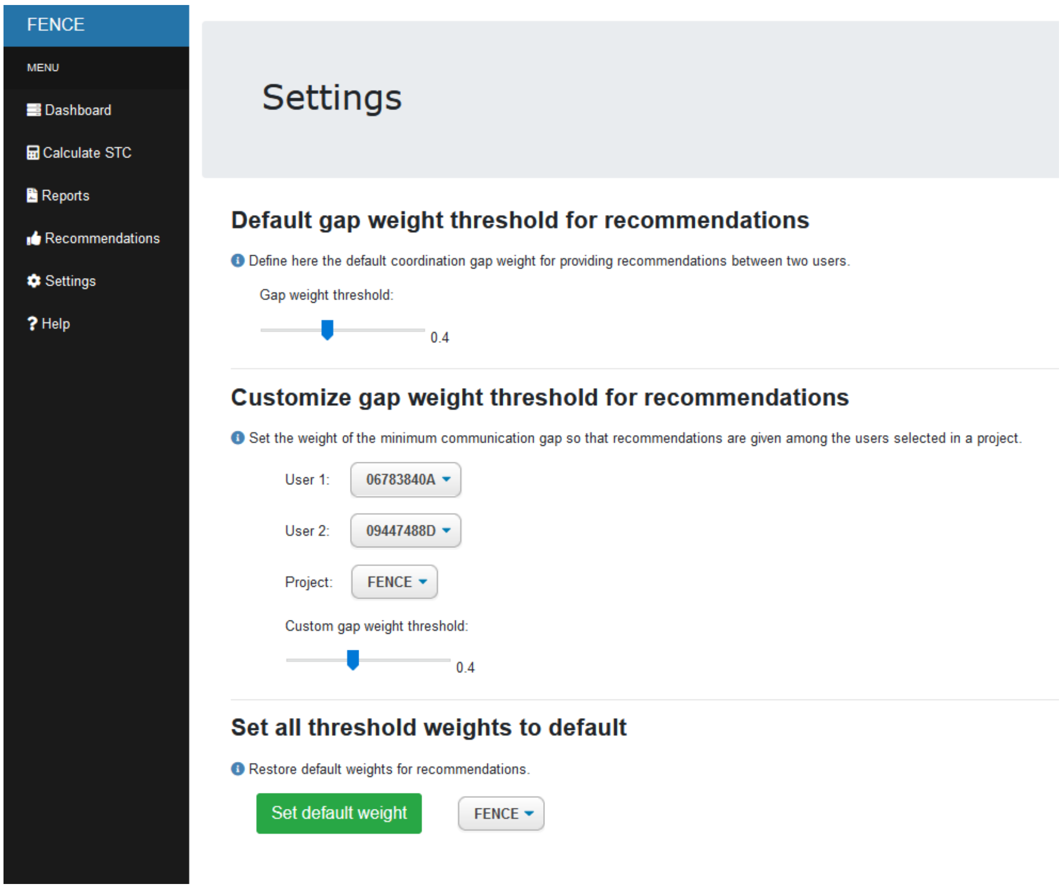
As indicated at the beginning of this section, whether or not to show recommendations for certain users should depend on the weight of the coordination gap present among them in the C_G matrix. On this basis, one of the improvements proposed at the end of this phase was the possibility of customizing from which weight in C_G the system would show recommendations for pairs of employees. This feature as well as others related to customization will be covered in the following section.

4.3.6. Module 6. Settings and preferences

The last functional module of FENCE is the one related to customization and settings. The objectives of this module include enabling the user to modify certain parameters used by the tool so that it can be adjusted to particular projects, teams or user's needs.

After the development of the STC measurement and recommendation modules, it was agreed that the following customisable features may be implemented:

- On the one hand, the possibility to generally adjust the weight in the C_G matrix considered to provide recommendations between pairs of users. As already explained, when offering recommendations by the expert system, the weight of the cells in C_G matrix is taken as a



Settings

Default gap weight threshold for recommendations

Define here the default coordination gap weight for providing recommendations between two users.

Gap weight threshold: 0.4

Customize gap weight threshold for recommendations

Set the weight of the minimum communication gap so that recommendations are given among the users selected in a project.

User 1: 06783840A

User 2: 09447488D

Project: FENCE

Custom gap weight threshold: 0.4

Set all threshold weights to default

Restore default weights for recommendations.

Set default weight FENCE

Figure 4.37: Settings window

reference to evaluate if there are communication problems between users and, therefore, to know if it is necessary to recommend communication solutions.

- On the other hand, it was also considered the possibility of establishing custom thresholds for the C_G weights used to provide recommendations among specific employees. To better illustrate the customization of this C_G weight threshold, let us consider the following example:

  Suppose that the system provides recommendations for any pair of users with a weight in the C_G matrix equal to or greater than 0.4.

  Two users, Alice and Bob, have a lack of coordination reflected in the C_G matrix with a value of 0.6, which means that the expert system will offer recommendations to improve their communication.

  However, we know that the communication between Alice and Bob is simple and effective because they are used to working together, so it would be convenient to be able to adjust the C_G threshold value to avoid getting unnecessary recommendations, for example, to 0.9.

  Thus, the user can set the minimum weight required to receive recommendations for the particular case of Alice and Bob to 0.9 through the FENCE settings interface.

With these ideas in mind, the development of these customizations was conducted (see Figure 4.37). In turn, the option to restore all weights to the established default value was also included, as well as the possibility of establishing these weights at project level, thus increasing customization and flexibility. In this way, the main functionalities of FENCE had finally been completed. All that remained to be done was to test and make improvements in the presentation and performance of

FENCE, an iterative procedure based on stakeholders' feedback.

4.4. TRANSITION PHASE

Although the tool was already functional and ready to be used, there were still some non-functional changes and actions to be performed. This transition stage includes the performance, usability and optimization testing of FENCE, as well as the deployment of the tool. This actions will be explained in the following subsections.

4.4.1. Interface improvements

With the backend part completed, efforts were focused on improving the application's interface and usability. This work involved reorganizing and improving the cascade style sheets (CSS), optimizing the JavaScript code and improving the overall appearance by modifying the HTML views.

Further graphic information, such as icons and coloured buttons, was also included, as well as a "Help" tab with information about STC and the use of FENCE for its proper management.

4.4.2. Algorithms optimization and bug resolution

Bugs related to the expert system and the recommendations provided were resolved, as well as attempts were made to further optimize the STC measurement algorithms.

Moreover, security improvements were attempted to be added in order to increase the robustness of the tool, such as input control in the database. However, given the time constraints of the project, this aspect was not addressed as deeply as would have been desired, so it is proposed as a future improvement of the tool.

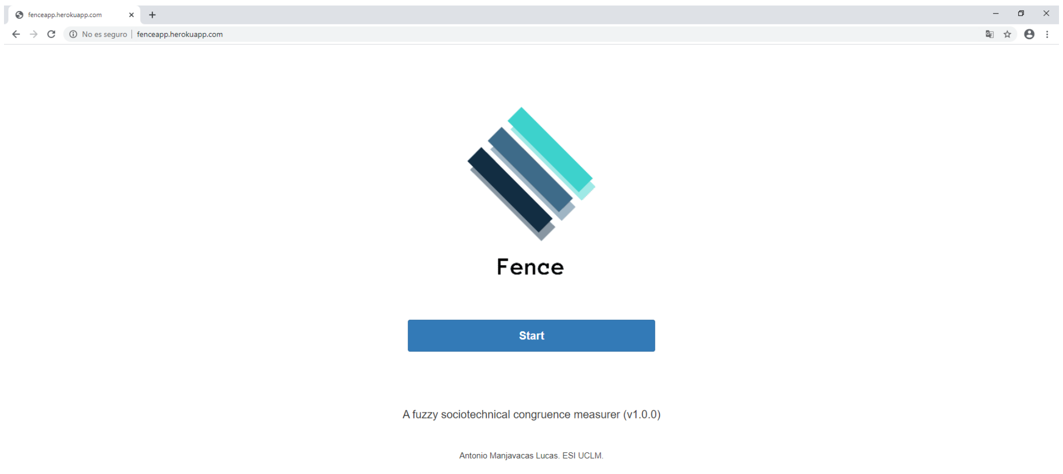
4.4.3. First release and documentation

Once the tool was finally considered to be completed, the building and packaging of the distributable executable of FENCE was performed. At the same time, most of the tool's documentation was produced in this phase: both in the GitHub repository where it is hosted and in this document itself.

4.4.4. Application deployment

Finally, the application was deployed on the Heroku cloud platform. This platform as a service (PaaS) allows free and limited deployment of cloud services, serving as an example of how FENCE would be used in a real web environment (see Figure 4.38, where the service hosted on www.fenceapp.herokuapp.com is shown).

It should also be noted that FENCE was initially considered as a tool for local use, and that it would be necessary to include security and access policies to ensure its concurrent use by multiple users. However, it can be seen that in this first version it is completely operative on the web, and that it would be a matter of future improvements to better adapt the tool to a fully distributed environment.



A fuzzy sociotechnical congruence measurer (v1.0.0)

Antonio Manjavacas Lucas. ESI UCLM.

Figure 4.38: FENCE deployed in Heroku Cloud Platform

CHAPTER 5

CONCLUSIONS

In this last chapter, we shall address the conclusions to which the project has led. We shall first look at how the project objectives have been met and how the project has been justified with the intensification competencies. We shall subsequently discuss future work and personal assessments after the completion of the project.

5.1. ACHIEVEMENT OF THE OBJECTIVES

The general objectives established at the beginning of this project were: the development of a tool with which to measure STC within an organisation, along with a recommendation system that could be used to improve the organisation's STC levels.

At this point, we can confirm the fulfilment of both objectives and we shall now, therefore, provide details on how they have been achieved by meeting the different specific objectives of the project:

- The project began with the development of a dashboard with which to allow the manipulation of the resources required in order to measure STC in an organisation. This was a challenging functional module, as it required training in frontend resources that the student had never previously used. Time constraints prevented the control of inputs and outputs, along with more robust testing, from being carried out to the level desired. Despite this, the functional module fulfils its task and meets the requirements of the tool without major problems.

- The subsequent implementation of the algorithms required in order to measure STC and its adaptation to GSD was undoubtedly the functionality that required the greatest amount of time. Efforts were also made to attempt to improve their performance and refine the measurement by means of a fuzzy control system. This objective also involved the simulation of communications in order to test the algorithms, the interface corresponding to this functional module and consultations with an expert in order to refine the measurements.

- One of the principal contributions of FENCE is the use of a fuzzy control system to improve STC measurement. Working with attributes expressed through linguistic labels provided the possibility of using a fuzzy control system to improve the calculation of STC levels. This functionality can also be considered as an original contribution if we compare it with other existing tools used to measure STC [47], since it allows the system to be more flexible for organizations with different socio-cultural distances that can impact on STC levels.

- In comparison with the previous modules, the visualization of the STC history levels meant less effort than expected, since the main tasks carried out were training in the use of *Chart.js* and the adaptation of the data model to the needs of the timelines. The greatest difficulty was, again, the use of frontend technologies not previously employed, although in this case it was not a major challenge.

- Finally, the implementation of the recommendation system for the improvement of STC levels was another of the most challenging milestones of the project. The complexity of this functional module meant exploring the processes of Knowledge Engineering and fuzzy logic in greater depth. It was also necessary to study the FCL standard and the jFuzzyLogic library in order to translate the expert knowledge into rules that could be used by the system. It is worth noting that one of the features that emerged while developing the expert system was the customization of threshold weights with which to provide recommendations. This feature, which was not originally intended, was added with the intention of improving the recommendations offered and making FENCE a more flexible tool.

As can be seen, the objectives set out at the beginning of the project were achieved. Furthermore, some of them were even expanded in order to improve the tool with features not initially contemplated, and this highlighted the importance of continuous communication with stakeholders throughout the project.

5.2. COMPETENCIES JUSTIFICATION

This section provides the rationale for the Computation competencies addressed in this project. Let us, therefore, study the competencies covered during the development of FENCE:

CM3: *Ability to evaluate the computational complexity of a problem, discover algorithmic strategies that can lead to its resolution and recommend, develop, and implement that which guarantees the best performance according to requirements.*

This project has led to a detailed study and more profound exploration of algorithms already existing in literature, with the objective of measuring STC. The existing algorithms were simultaneously improved and adapted to the specific needs of GSD contexts.

CM4: *Ability to understand the fundamentals, paradigms, and techniques of intelligent systems and to analyse, design and build computer systems, services and applications using these techniques in any field of application.*

The development of FENCE has involved the implementation of an intelligent system for the measurement and improvement of STC. Fuzzy inference systems have been specifically used to achieve this goal.

CM5: *Ability to acquire, obtain, formalize, and represent human knowledge in a computable manner for problem solving by means of a computer system in any field of application, particularly those related to aspects of computing, perception and performance in environments or intelligent environments.*

This competence has been addressed through the development of a knowledge-based system. Its implementation has involved Knowledge Engineering procedures, in addition to the formalization and representation of knowledge through the use of fuzzy rules.

5.3. LESSONS LEARNED

The development of this project has made it possible to relate the importance of STC in organisations, its significant influence in GSD contexts and how useful it is to employ fuzzy logic and knowledge-based systems in order to improve it.

With regard to the STC measurement, various alternatives were initially studied when deciding which algorithms to use for FENCE. The STC measurement method outlined by Cataldo et al. [6, 7] was discarded owing to the low flexibility provided by its bivalued-weight matrices. Other metrics, such as that of Valetto et al. [55] could not be adapted to GSD contexts, as explained in [47]. Only the solutions proposed by Portillo et al. [40] and Kwan et al. [24, 25] were suitable for such environments, being the proposal of Kwan et al. the one finally chosen, since it is a more general procedure and, therefore, more open and adaptable to FENCE requirements.

It was simultaneously necessary to carry out a reformulation of the algorithms presented by Kwan et al. in order to improve their performance as much as possible. This was achieved by replacing the use of matrices with objects stored in the database, thus reducing their complexity and, therefore, execution time.

Furthermore, extracting expert knowledge was not an easy task and it took several meetings to agree on a robust proposal that would serve as the basis for the expert system. The development of this expert system signified the need to make major efforts with the knowledge engineer such as in-depth individual research into unknown topics in order to better extract expert knowledge; the planning of structured interviews that would guarantee the expert's comfort when contributing expertise, or dealing with the ambiguity of human language.

One of the most valuable lessons learned from this project was that going into greater depth throughout its development could lead to the consideration of improvements not initially contemplated but which provided the tool with significant value. As an example, in the early days of FENCE, we did not contemplate the customisation of threshold weights in order to provide recommendations, but after a discussion we agreed that it could be an interesting feature to be added. Any changes agreed with stakeholders that do not involve a major departure from the main objectives of the project should, therefore, be welcomed as long as they add value and are profitable.

Moreover, it is important to emphasize the relevance of carrying out a detailed study of the tools that are being used. It took several days to compare multiple Java fuzzy logic libraries and, once chosen, the development of the fuzzy control system required training time, not only as regards the libraries used in the papers published by their authors, but also as regards the notions of fuzzy logic. This was necessary work that took several weeks, although the time invested was worthwhile and speeded up the process of implementing the system.

In this development, it was also necessary to learn the basic syntax of FCL (structures, variables, functions, etc.). This resulted in a really useful and interesting fuzzy control language, which the author recommends when confronting projects based on fuzzy inference systems.

The development of FENCE using tools such as Spring Boot, HTML, JavaScript, CSS or jQuery together for the first time was also a real challenge, involving a long period of self-teaching and improvement throughout the project.

Finally, planning was essential. Many of the proposed objectives would not have been easily achieved without the proper management of the time invested in each iteration, anticipating possible problems or the reduction of risks, such as advancing in important aspects of the project without the stakeholders' prior approval.

5.4. FUTURE WORK

Concerning the future work that could be derived from this project, many possibilities can be explored.

First, one of the drawbacks of the algorithms needed to measure STC is their low scalability. Their performance is not sufficiently high when input data (users, tasks, teams, assignments, dependencies) grow to more realistic sizes. In fact, improving the performance of these algorithms is a complex task that we intend to confront as further work. As an example, and considering the matrixial nature of these algorithms, an initial proposal that could be addressed is that of increasing parallelization by means of threads.

Focusing on the recommendation of solutions by which to improve STC, the work presented in this project could be the basis of a much more complex system, providing a greater number of solutions that are not only related to communication. In-depth information concerning dependencies, the availability of different employees, or task deadlines are some of the initial proposals that are suggested as improvements to be addressed in the future.

Moreover, if we focus on the use of data to manage STC, one possible future approach could be the implementation of a functional module that would allow the estimation of future STC levels based on historical measurements and the organisation's current status.

In conclusion, the possibilities of STC are still wide and varied. This makes STC a broad field of study, in which all the proposed solutions should always lead to the benefit of the organisations and those who shape them.

5.5. PERSONAL APPRAISAL

With regard to personally appraising this project, I consider FENCE to be merely the first step in providing a new approach as to how STC can be measured and improved. My view is that it is a particularly useful and valuable tool if it is put to good use, as it is first and foremost capable of raising the awareness of such important human aspects as coordination and communication.

This project has meant many things to me at a personal level. On the one hand, being able to put into practice what I have learnt during all these years has been a considerable source of motivation and pride. On the other, being able to build bridges between my passion for artificial intelligence and research has also been a tremendous stimulus for me.

Moreover, the development of this project has meant perseverance, self-demand, discipline and, above all, creativity, values that I hope to continue improving over time. This work has undoubtedly also allowed me to look at things in perspective and see how, thanks to these principles, I have arrived where I am today.

Thus, after four years of intense work, this period comes to an end. With no interest in stating more than necessary, the completion of this project closes one of the most important periods of my life, which has modelled me as a person and future engineer and which, despite the enormous efforts and sacrifices it has required, has undoubtedly been worthwhile.

Antonio Manjavacas Lucas
Campo de Criptana, June 3rd, 2020.

BIBLIOGRAPHY

[1] P. J. Ågerfalk *et al.*, “Benefits of global software development: The known and unknown”, in *Making Globally Distributed Software Development a Success Story*, ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2008, pp. 1–9, ISBN: 978-3-540-79588-9. DOI: 10.1007/978-3-540-79588-9_1.

[2] K. Beck *et al.* (2001). “Manifesto for Agile Software Development”, [Online]. Available: https://agilemanifesto.org/ (visited on 01/31/2020).

[3] R. E. Bellman and L. A. Zadeh, “Decision-Making in a Fuzzy Environment”, *Management Science*, vol. 17, no. 4, B–141, Dec. 1, 1970, ISSN: 0025-1909. DOI: 10.1287/mnsc.17.4.B141.

[4] F. Bolici, J. Howison, and K. Crowston, “Coordination without discussion? Socio-technical congruence and Stigmergy in Free and Open Source Software projects”, presented at the Socio-Technical Congruence Workshop in conj Intl Conf on Software Engineering, Vancouver, Canada, 2009.

[5] M. Cataldo and J. D. Herbsleb, “Coordination Breakdowns and Their Impact on Development Productivity and Software Failures”, *IEEE Transactions on Software Engineering*, vol. 39, no. 3, pp. 343–360, Mar. 2013, ISSN: 1939-3520. DOI: 10.1109/TSE.2012.32.

[6] M. Cataldo, J. D. Herbsleb, and K. M. Carley, “Socio-technical congruence: a framework for assessing the impact of technical and work dependencies on software development productivity”, in *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*, ser. ESEM ’08, Kaiserslautern, Germany: Association for Computing Machinery, Oct. 9, 2008, pp. 2–11, ISBN: 978-1-59593-971-5. DOI: 10.1145/1414004.1414008. [Online]. Available: https://doi.org/10.1145/1414004.1414008.

[7] M. Cataldo *et al.*, “Identification of coordination requirements: implications for the Design of collaboration and awareness tools”, in *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, ser. CSCW ’06, Banff, Alberta, Canada: Association for Computing Machinery, Nov. 4, 2006, ISBN: 978-1-59593-249-5. DOI: 10.1145/1180875.1180929. [Online]. Available: https://doi.org/10.1145/1180875.1180929.

[8] P. Cingolani and J. Alcalá-Fdez, “jFuzzyLogic: A robust and flexible fuzzy-logic inference system language implementation”, in *2012 IEEE International Conference on Fuzzy Systems*, Brisbane, Australia: IEEE, Jun. 2012, pp. 1–8. DOI: 10.1109/FUZZ-IEEE.2012.6251215.

[9] ——, “jFuzzyLogic: A java library to design fuzzy logic controllers according to the standard for fuzzy control programming”, *International Journal of Computational Intelligence Systems*, vol. 6, pp. 61–75, sup1 Jun. 2013. DOI: 10.1080/18756891.2013.818190.

[10] E. Ó. Conchúir *et al.*, “Global software development: where are the benefits?”, *Communications of the ACM*, vol. 52, no. 8, pp. 127–131, Aug. 1, 2009, ISSN: 0001-0782. DOI: 10.1145/1536616.1536648. [Online]. Available: https://doi.org/10.1145/1536616.1536648.

[11] M. E. Conway, “How do committees invent”, *Datamation*, vol. 14, no. 4, pp. 28–31, 1968.

[12] K. Ehrlich *et al.* (2008). “An analysis of congruence gaps and their effect on distributed software development”.

[13] J. Gaschnig, “Prospector: an expert system for mineral exploration”, *D. Michie, ed*, vol. 1982, pp. 47–64, 1982.

[14] L. Gupta. (2018). “What is REST – Learn to create timeless REST APIs”, [Online]. Available: https://restfulapi.net/ (visited on 05/14/2020).

[15] J. Herbsleb and R. Grinter, “Architectures, coordination, and distance: Conway’s law and beyond”, *IEEE Software*, vol. 16, no. 5, pp. 63–70, Sep. 1999, ISSN: 1937-4194. DOI: 10.1109/52.795103.

[16] G. Hofstede, *Cultures and Organizations. Software of the Mind.* Jan. 1, 2004, vol. 2, ISBN: 978-0-07-143959-6.

[17] ——, *Compare countries*, Hofstede Insights. [Online]. Available: https://www.hofstede-insights.com/product/compare-countries/.

[18] ——, *The 6 dimensions model of national culture by Geert Hofstede.* [Online]. Available: https://geerthofstede.com/culture-geert-hofstede-gert-jan-hofstede/6d-model-of-national-culture/.

[19] IEC. (1997). “Fuzzy Control Language (FCL). IEC 61131-7 standard”, [Online]. Available: http://ffll.sourceforge.net/fcl.htm.

[20] A. Kandel, *Fuzzy Expert Systems.* CRC Press, Nov. 12, 1991, ISBN: 978-0-8493-4297-4.

[21] S. Krishnankutty. (2020). “eMathTeacher: Mamdani’s fuzzy inference method - Membership functions”, [Online]. Available: http://www.dma.fi.upm.es/recursos/aplicaciones/logica_borrosa/web/fuzzy_inferencia/funpert_en.htm (visited on 05/22/2020).

[22] I. Kwan, M. Cataldo, and D. Damian, “Conway’s Law Revisited: The Evidence for a Task-Based Perspective”, *IEEE Software*, vol. 29, no. 1, pp. 90–93, Jan. 2012, ISSN: 1937-4194. DOI: 10.1109/MS.2012.3.

[23] I. Kwan and D. Damian, “Extending socio-technical congruence with awareness relationships”, in *Proceedings of the 4th international workshop on Social software engineering*, ser. SSE ’11, Szeged, Hungary: Association for Computing Machinery, Sep. 5, 2011, pp. 23–30, ISBN: 978-1-4503-0850-2. DOI: 10.1145/2024645.2024652. [Online]. Available: https://doi.org/10.1145/2024645.2024652.

[24] I. Kwan, A. Schröter, and D. Damian, “A weighted congruence measure”, in *Workshop on SocioTechnical Congruence*, 2009, pp. 1–4.

[25] ——, “Does Socio-Technical Congruence Have an Effect on Software Build Success? A Study of Coordination in a Software Project”, *IEEE Transactions on Software Engineering*, vol. 37, no. 3, pp. 307–324, 2011.

[26] J. Lederberg, “How DENDRAL was conceived and born”, in *Proceedings of ACM conference on History of medical informatics*, ser. HMI ’87, Association for Computing Machinery, Dec. 1, 1987, pp. 5–19, ISBN: 978-0-89791-248-8. DOI: 10.1145/41526.41528. [Online]. Available: https://doi.org/10.1145/41526.41528.

[27] J. Li *et al.*, “Analysis of requirement and constrained model of inter-satellite-link TT&c scheduling problem on navigation constellation”, in *Artificial Intelligence and Computational Intelligence*, ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2012, pp. 160–169, ISBN: 978-3-642-33478-8. DOI: 10.1007/978-3-642-33478-8_21.

[28] B. MacIsaac, O. van der Straate, and R. Balduino. (2018). “Eclipse Process Framework Project (EPF) | The Eclipse Foundation”, [Online]. Available: https://www.eclipse.org/epf/ (visited on 01/28/2020).

[29] B. Malik *et al.*, “Geographical distance and communication challenges in global software development: A review”, *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 5, pp. 406–414, 2018. DOI: 10.14569/IJACSA.2018.090553.

[30] A. Manjavacas *et al.*, “Global software development governance: Challenges and solutions”, *Journal of Software: Evolution and Process*, 2020. DOI: 10.1002/smr.2266.

[31] S. Marczak, I. Kwan, and D. Damian, “Investigating Collaboration Driven by Requirements in Cross-Functional Software Teams”, in *2009 Collaboration and Intercultural Issues on Requirements: Communication, Understanding and Softskills*, Aug. 2009, pp. 15–22. DOI: 10.1109/CIRCUS.2009.2.

[32] J. L. Maté and J. Pazos, *Ingeniería del conocimiento. Diseño y construcción de sistemas expertos.* Córdoba (Argentina): SEPA, 1998, ISBN: 950-9592-06-4.

[33] L. R. Medsker, "Fuzzy logic and expert systems", in *Hybrid Intelligent Systems*, Boston, MA: Springer US, 1995, pp. 95–105, ISBN: 978-1-4615-2353-6. DOI: 10.1007/978-1-4615-2353-6_6. [Online]. Available: https://doi.org/10.1007/978-1-4615-2353-6_6.

[34] R. C. Mora, *Conversaciones con CEOs y CIOs sobre Transformación Digital y Metodologías Ágiles*, 1st ed. Madrid: Agibilibooks, May 17, 2017, 182 pp., ISBN: 978-84-946851-0-1.

[35] C. G. Morcillo, «Técnicas de Softcomputing», pág. 29,

[36] C. Negoita, "Expert systems and fuzzy systems", Jan. 1, 1985. [Online]. Available: https://www.osti.gov/biblio/5127536.

[37] M. Niazi *et al.*, "Challenges of project management in Global Software Development: Initial results", in *2013 Science and Information Conference*, Oct. 2013, pp. 202–206.

[38] X. Peng, M. Ali Babar, and C. Ebert, "Collaborative Software Development Platforms for Crowdsourcing", *IEEE Software*, vol. 31, no. 2, pp. 30–36, Mar. 2014, ISSN: 1937-4194. DOI: 10.1109/MS.2014.31.

[39] D. Popovic, *Methods and Tools for Applied Artificial Intelligence.* CRC Press, May 2, 1994, 548 pp., ISBN: 978-0-8247-9195-7.

[40] J. Portillo-Rodríguez *et al.*, "Using agents to manage Socio-Technical Congruence in a Global Software Engineering project", *Information Sciences*, vol. 264, pp. 230–259, 2014, ISSN: 0020-0255. DOI: https://doi.org/10.1016/j.ins.2014.01.009.

[41] M. Richards. (2015). "Layered Architecture - Software Architecture Patterns", [Online]. Available: https://www.oreilly.com/library/view/software-architecture-patterns/9781491971437/ch01.html (visited on 05/15/2020).

[42] N. Saleem, S. Mathrani, and N. Taskin, "Understanding the Different Levels of Challenges in Global Software Development", in *2019 ACM/IEEE 14th International Conference on Global Software Engineering (ICGSE)*, May 2019, pp. 76–77. DOI: 10.1109/ICGSE.2019.00027.

[43] J. Salido. (2020). "Plantilla guía de TFG de la ESI-UCLM". GitHub, Ed., Universidad de Castilla-La Mancha, [Online]. Available: https://github.com/JesusSalido/TFG_ESI_UCLM.

[44] A. Sarma, J. Herbsleb, and A. v. d. Hoek, "Challenges in Measuring, Understanding, and Achieving Social-Technical Congruence", 2008.

[45] M. Shameem *et al.*, "A systematic literature review to identify human related challenges in globally distributed agile software development: towards a hypothetical model for scaling agile methodologies", in *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, Dec. 2018, pp. 1–7. DOI: 10.1109/CCAA.2018.8777533.

[46] E. H. Shortliffe, "MYCIN: a rule-based computer program for advising physicians regarding antimicrobial therapy selection.", 1974.

[47] J. M. Sierra *et al.*, "A systematic mapping study about socio-technical congruence", *Information and Software Technology*, vol. 94, pp. 111–129, Feb. 1, 2018, ISSN: 0950-5849. DOI: 10.1016/j.infsof.2017.10.004.

[48] D. Šmite and Z. Galviņa, "Socio-technical congruence sabotaged by a hidden onshore outsourcing relationship: Lessons learned from an empirical study", in *Product-Focused Software Process Improvement*, ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2012, pp. 190–202, ISBN: 978-3-642-31063-8. DOI: 10.1007/978-3-642-31063-8_15.

[49] Spring. (2020). "Spring Boot", [Online]. Available: https://spring.io/projects/spring-boot.

[50] M. M. M. Syeed *et al.*, "Socio-Technical Congruence in the Ruby Ecosystem", in *Proceedings of The International Symposium on Open Collaboration*, ser. OpenSym '14, Berlin, Germany: Association for Computing Machinery, Aug. 27, 2014, pp. 1–9, ISBN: 978-1-4503-3016-9. DOI: 10.1145/2641580.2641586. [Online]. Available: https://doi.org/10.1145/2641580.2641586.

[51] M. M. Syeed, *On the Socio-Technical Dependencies in Free/Libre/Open Source Software Projects.* Tampere University of Technology, 2015, ISBN: 978-952-15-3525-3.

[52] M. M. Syeed and I. Hammouda, “Socio-Technical Dependencies in Forked OSS Projects: Evidence from the BSD Family.”, *JSW*, vol. 9, no. 11, pp. 2895–2909, 2014.

[53] Tecnoempleo. (2020). “Datos y estadísticas sobre el Empleo en Informática y Tecnología - tecnoempleo.com”, [Online]. Available: https://www.tecnoempleo.com/informe-empleo-informatica.php (visited on 05/26/2020).

[54] A. D. Toro and B. B. Jiménez, “Metodología para la Elicitación de Requisitos de Sistemas Software”, Sevilla, Oct. 2000, p. 78. [Online]. Available: http://www.lsi.us.es/docs/informes/lsi-2000-10.pdf.

[55] G. Valetto *et al.*, “Using Software Repositories to Investigate Socio-technical Congruence in Development Projects”, in *Fourth International Workshop on Mining Software Repositories (MSR’07:ICSE Workshops 2007)*, May 2007, pp. 25–25. DOI: 10.1109/MSR.2007.33.

[56] P. Wagstrom, J. D. Herbsleb, and K. M. Carley, “Communication, team performance, and the individual: bridging technical dependencies.”, *Academy of Management Proceedings*, vol. 2010, no. 1, pp. 1–7, Aug. 1, 2010, ISSN: 0065-0668. DOI: 10.5465/ambpp.2010.54500789. (visited on 05/17/2020).

[57] D. Waterman, “A guide to expert systems”, Jan. 1, 1986. [Online]. Available: https://www.osti.gov/biblio/5480398 (visited on 05/22/2020).

[58] Wikipedia, *Fuzzy logic*, in *Wikipedia*, May 22, 2020. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Fuzzy_logic&oldid=958171295 (visited on 05/22/2020).

[59] L. A. Zadeh, “Fuzzy sets”, *Information and Control*, vol. 8, no. 3, pp. 338–353, Jun. 1, 1965, ISSN: 0019-9958. DOI: 10.1016/S0019-9958(65)90241-X.

[60] ——, “The concept of a linguistic variable and its application to approximate reasoning—i”, *Information Sciences*, vol. 8, no. 3, pp. 199–249, Jan. 1, 1975, ISSN: 0020-0255. DOI: 10.1016/0020-0255(75)90036-5.

[61] A. Al-Zaidi and R. Qureshi, “Global software development geographical distance communication challenges”, *International Arab Journal of Information Technology*, vol. 14, no. 2, pp. 215–222, 2017.

APPENDICES

APPENDIX A

SIMULATED COMMUNICATIONS

This annex includes the Python code used for the generation of random communications (Listing A.1). The simulated communication activities were inserted in the corresponding collection within the MongoDB database. Note how the PyMongo library was used for this purpose.

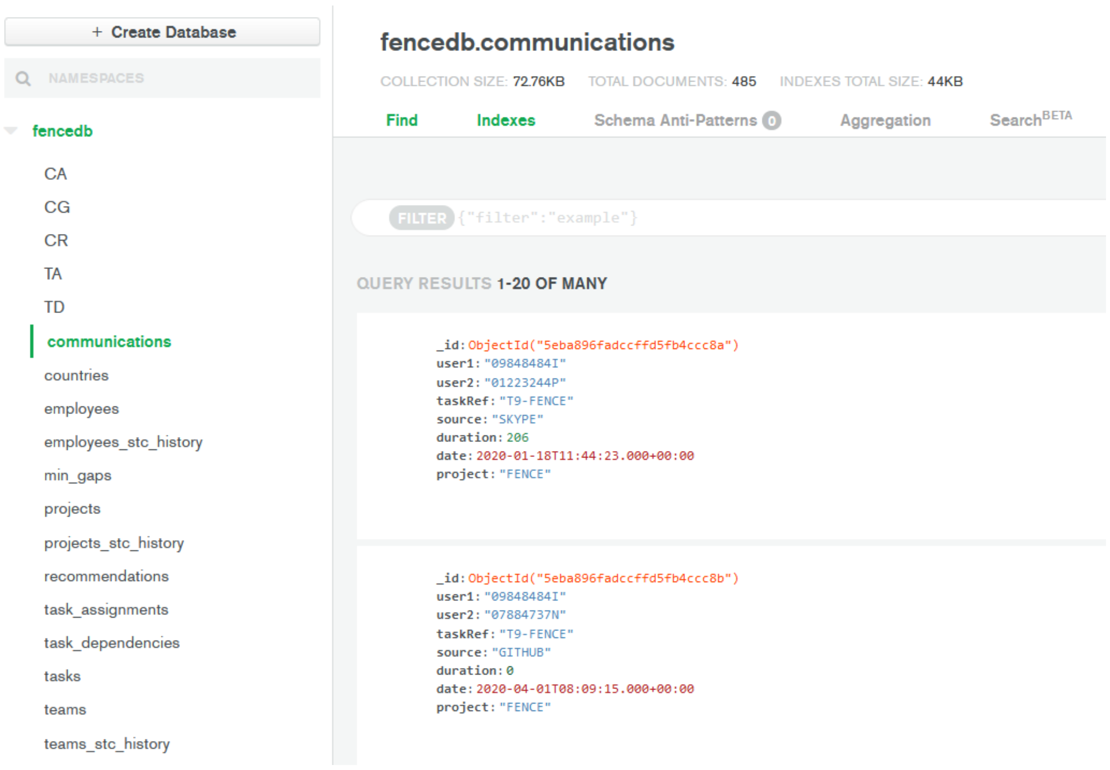
The attributes generated in a simulated way and that constitute a communication activity are the following:

- **user1** and **user2**: the users involved in the bidirectional communication activity.
- **taskRef**: identification of the task addressed in the communication activity.
- **source**: random communication channel.
- **duration**: time spent on synchronous communications.
- **date**: random date and hour between January and May of 2020.
- **project**: project within which the communication is framed.

For security reasons, part of the URI used to make the connection to MongoDB Atlas has been omitted. Therefore, the result of the execution will be a set of simulated communications as the ones shown in Figure A.1.

Listing A.1: Random communications generator

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-

'''
Random communication activities generator
Required packages:
  - dnspython
  - faker
Antonio Manjavacas. FENCE. ESI UCLM. 2020.
'''

import pymongo as pm
import datetime as dt

from faker import Faker
from random import randint


URI = 'mongodb+srv://username:fenceapp@app-httxx.mongodb.net/'

sources = ['JIRA', 'EMAIL', 'PHONE', 'SLACK', 'TEAMS', 'GITHUB', 'SKYPE', ↩
    ↪ 'OTHER']
MAX = 8

client = pm.MongoClient(URI)
db = client['fencedb']

def random_date():
  # year, month, day, hour, minute, second, microsecond
  return dt.datetime(2020, randint(1,5), randint(1,28), randint(8,20), ↩
      ↪ randint(0,59), randint(0,59))


tasks_ids = []
for document in db.tasks.find():
  tasks_ids.append(document['reference'])

employees_ids = []
for document in db.employees.find():
  employees_ids.append(document['dni'])

# delete previous
db.drop_collection('communications')

comm_list = []

requirements = db['CR']
for req in requirements.find():
  n = randint(0,MAX)
  for i in range(0,n):
    communication = {}
    communication['user1'] = req['user1']
    communication['user2'] = req['user2']
    communication['taskRef'] = req['task']

    source = sources[randint(0,len(sources) - 1)]
    communication['source'] = source

    if source in ['PHONE', 'TEAMS', 'SKYPE', 'OTHER']:
      communication['duration'] = float(randint(20,600))
    else:
      communication['duration'] = 0

    communication['date'] = random_date()
    communication['project'] = req['project']

    comm_list.append(communication)

communications = db['communications']
for c in comm_list:
  communications.insert_one(c)
```



fencedb.communications

COLLECTION SIZE: 72.76KB TOTAL DOCUMENTS: 485 INDEXES TOTAL SIZE: 44KB

QUERY RESULTS 1-20 OF MANY

```
_id: ObjectId("5eba896fadccffd5fb4ccc8a")
user1: "09848484I"
user2: "01223244P"
taskRef: "T9-FENCE"
source: "SKYPE"
duration: 206
date: 2020-01-18T11:44:23.000+00:00
project: "FENCE"
```

```
_id: ObjectId("5eba896fadccffd5fb4ccc8b")
user1: "09848484I"
user2: "07884737N"
taskRef: "T9-FENCE"
source: "GITHUB"
duration: 0
date: 2020-04-01T08:09:15.000+00:00
project: "FENCE"
```

Figure A.1: Sample communication activities