



# Investigating Conditioning Factors for Transparency in Software Ecosystems

Rodrigo Oliveira Zacarias  [ Universidade Federal do Estado do Rio de Janeiro | [rodrigo.zacarias@edu.unirio.br](mailto:rodrigo.zacarias@edu.unirio.br) ]

Rodrigo Feitosa Gonçalves  [ Universidade Federal do Estado do Rio de Janeiro | [rfeitosa@edu.unirio.br](mailto:rfeitosa@edu.unirio.br) ]

Rodrigo Pereira dos Santos  [ Universidade Federal do Estado do Rio de Janeiro | [rps@uniriotec.br](mailto:rps@uniriotec.br) ]

Patricia Lago  [ Vrije Universiteit Amsterdam | [p.lago@vu.nl](mailto:p.lago@vu.nl) ]

## Abstract

Software Ecosystems (SECO) are a set of actors interacting with a distributed market centered on a common technological platform to develop products and services. In this context, transparency allows third-party developers to learn processes and elements that integrate the SECO platform. This non-functional requirement impacts the coordination of developers and the management of requirements that emerge in SECO. Although it is an essential requirement, there is still a lack of a roadmap on what constitutes transparency in SECO. Thus, this article aims to characterize conditioning factors for transparency in SECO. To do so, we conducted a systematic mapping study (SMS) and a field study to identify and analyze such factors. After investigating the literature, we selected 23 studies to analyze the state-of-the-art about transparency in SECO. Next, we conducted interviews with 16 software developers to characterize the importance of conditioning factors for transparency identified in their interaction with GitHub, a platform to support project-based ecosystems. As results, we obtained a comprehensive view of solutions, conditioning factors, processes, and concerns related to transparency in SECO, whose discussion is centered on three main topics: access to information, communication channels, and requirements engineering. We also present a conceptual framework that structures all the knowledge about transparency in SECO obtained in both studies. Regarding implications for academia and industry, researchers can find a conceptual framework to be used as a foundation for systematic approaches to understanding transparency in SECO. Practitioners can find solutions and conditioning factors that help them to adopt initiatives to contribute to the open flow of information in a SECO and, thus, attract and engage new actors to a common technological platform.

**Keywords:** *Software Ecosystems, Transparency, Systematic Mapping Study, Field Study, GitHub*

## 1 Introduction

With advances in software development strategies and approaches to meet new market demands, it has been a great challenge for corporations to maintain a system/software architecture fully internalized into its organization. Hence, some companies have invested in opening their architectures to allow third-party developers to collaborate in producing their components around a common technological platform. This practice defines the concept of Software Ecosystem (SECO) (Barbosa et al., 2013; Jansen, 2020).

Lungu and Lanza (2010) also characterize a SECO as a collection of software projects that are developed and evolved together in the same environment. This environment can be represented by companies or developers communities that collaborate on software projects. Thereby SECO is characterized by gathering several projects, systems, and actors over a common technological platform, also called an SECO platform.

The SECO paradigm has allowed the development of globalized and large-scale platforms (Manikas and Hansen, 2013). These platforms give rise to more complex systems that integrate a network of artifacts and actors (internal and external developers, users, etc.) that generate complex collaborative relationships. As a consequence, SECO brings a large flow of information that arises through open communication channels, mainly the flow of requirements information between the actors on the common technological platform (Knauss et al., 2018).

Due to the complexity inherent to the dynamics of relationships between actors involved and the collaborative environment risks, maintaining a balanced and sustainable SECO in the software market over time has been challenging for a keystone, organization that owns the platform. To achieve this effectively, actors, especially third-party developers, have to be aware of processes and elements (e.g., documentation files, source code, forums, etc.) that form a SECO platform (Cataldo and Herbsleb, 2010). Thus, SECO transparency is essential (Santos et al., 2016) as it allows actors to access and learn how to use the platform information.

In this context, transparency is considered at the software level, called software transparency, i.e. a condition in which all software functions are disclosed to users and developers, a precondition for adequate risk management (Leite and Cappelli, 2010). As a non-functional requirement, transparency must be considered in all stages of software design. So that, there is clarity on how they provide transparency to organizational processes and information. Thus, transparency is not only in the final object but also in the entire processing and information treatment and must allow stakeholders (software developers, project managers, clients, and end-users) to answer their questions about the software during its life cycle (Cysneiros, 2013; Hosseini et al., 2016).

As an example of the importance of software transparency in SECO, we can mention third-party developers' interaction in mobile SECO (e.g., Android<sup>1</sup> or iOS<sup>2</sup>). These third-party

<sup>1</sup><https://developer.android.com/>

<sup>2</sup><https://developer.apple.com/>

developers access SECO portals to obtain information about how to develop applications in this ecosystem, i.e., check which tools they should download, check what configurations are necessary, access source code repository, access programming language documentation, and check store requirements of applications, among others. If this information cannot be accessed or understood easily, these third-party developers may have difficulty developing their applications autonomously. This situation can generate an unpleasant Developer Experience (DX) and cause them to give up interacting with the common technological platform (Barbosa et al., 2013; Knauss et al., 2018; Meireles et al., 2019).

In addition, software transparency contributes to value creation for different actors, which generates benefits for SECO. Transparency in SECO helps actors to be aware of the evolution of development activities (Cataldo and Herbsleb, 2010). It causes a positive impact on the trust and credibility of key-stone strategic actions. Furthermore, easy access to communication channels helps to receive feedback from the community, which can lead to the emergence of new requirements that contribute to the SECO platform's evolution (Knauss et al., 2018; Hou and Jansen, 2023).

According to Knauss et al. (2018), openness and transparency in SECO encourage contributions from actors, mainly third-party developers, that emerge through open communication channels. This early engagement can be crucial to the health and success of requirements in SECO. So, Hanssen (2012) sees openness and transparency as key features of social networks in a SECO that allow actors to collaborate more efficiently.

However, transparency in SECO has received little attention from the scientific community, which is puzzling since developers often start development by choosing a platform that satisfies most of the functionality needed for software development (Jansen et al., 2013; Santos et al., 2016; Meireles et al., 2019). As we previously mentioned, the lack of transparency can hinder communication between the actors, hamper access to information, and make it difficult to understand the information provided in SECO (Knauss et al., 2018; Meireles et al., 2019). These characteristics affect DX and may result in a lack of interest and engagement among third-party developers. This situation could harm SECO, as it depends on these developers' contribution to remain sustainable in the software market (Fontão et al., 2021).

Therefore, this article aims to characterize conditioning factors for transparency in SECO. To do so, we conducted two studies to identify and analyze such factors. Firstly, we conducted a systematic mapping study (SMS) on scientific databases and digital libraries to map and analyze the state-of-the-art of transparency in SECO to answer the following research question (RQ): "*How is transparency in the SECO context characterized?*" After investigating the literature, we selected 23 studies for our analysis. As a result, we could have a comprehensive view of solutions, conditioning factors, processes, and concerns related to transparency in SECO, whose discussion is centered on three main topics: access to information, communication channels, and requirements engineering.

Secondly, we conducted a field study with software developers to characterize the importance level of conditioning

factors for transparency<sup>3</sup> identified in the previous study in their interaction with a SECO. In this study, we interviewed 16 software developers who use GitHub<sup>4</sup>, a platform to support project-based ecosystems, mainly open source software ecosystems (OSSECO) (Lungu and Lanza, 2010; Liao et al., 2019), to answer this RQ: "*What is the importance level of conditioning factors for transparency according to developers in the context of an OSSECO?*". As a result, we delved deeper into the conditioning factors for transparency identified in the literature and observed that most software developers agree with them. Thus, we could organize all the knowledge obtained in a conceptual framework to understand transparency in SECO better.

This article is an extended version of a conference paper (Zacarias et al., 2023), awarded as a distinguished paper at the 37th Brazilian on Software Engineering (SBES 2023). This paper presented the first study's results and characterized transparency in SECO based on the scientific literature. We complement our previous work by confirming the conditioning factors for transparency with 16 software developers who work in a SECO context, identifying the transparency benefits and the consequences of lack of transparency, and presenting a conceptual framework that comprises all the knowledge about transparency in SECO obtained in both studies.

Regarding implications for academia and industry, academics can find in this work a conceptual framework to better understand transparency in SECO. We also have listed future perspectives that can contribute to the advancement of state-of-the-art. With this conceptual framework, practitioners can understand transparency as a key element in dealing with requirements that emerge from different communication channels in platform openness. We have presented solutions and conditioning factors that can help them to adopt initiatives to contribute to the open flow of information in a SECO and, thus, attract and engage new actors in a common technological platform.

The remainder of this article is organized as follows: Section 2 presents background and related work; Section 3 depicts the research method of both studies; Section 4 presents the results obtained in both studies as well as the conceptual framework for understanding transparency in SECO; discussion and implications of the results of both studies are presented in Section 5; Section 6 describes the threats to validity and credibility; and, finally, Section 7 concludes the article with final remarks and future work.

## 2 Background

This section describes the concepts related to SECO and transparency. In addition, we also present related work to this

<sup>3</sup>In the context of this research, conditioning factors are defined as elements, characteristics, or actions that are necessary but not sufficient for the transparency of SECO. For example, providing real-time information about changes occurring in the common technological platform can help developers better understand its evolution. This action creates a condition of transparency of SECO information that tends to positively impact the developer's interaction with the platform, although it is not guaranteed that everyone will be impacted in the same way.

<sup>4</sup><https://github.com/>

research.

## 2.1 Software Ecosystems

A SECO can be defined as a set of actors that function as a unit and their relationships and interactions with a distributed market between software and services. These relationships are largely centered on a technology platform or a common market, which allows the exchange of information, resources, and artifacts (Jansen et al., 2009). These elements together form a SECO and require the integration of support mechanisms and tools to carry out those exchanges and guarantee communication and interaction between developers and users (Santos, 2016; Jansen, 2020).

In this multiple actors' context, we can identify three main roles: keystone, end-users, and third-party developers. A keystone is an organization or group that drives the development of the SECO platform. End-users are customers who need the platform to do their business. Finally, third-party developers use the platform as a basis to develop new products and solutions (Hanssen and Dybå, 2012).

SECO can be classified into three types: proprietary, open source, and hybrid. Proprietary SECO (PSECO) have their value creation based on proprietary contributions (e.g., SAP - System Applications and Products - and Amazon). Open source SECO (OSSECO) allow contributions from different actors and communities (e.g., Eclipse Foundation, GitHub, GitLab, and Apache Foundation). Finally, hybrid SECO support both proprietary and open source contributions (e.g., Android and iOS) (Manikas, 2016).

## 2.2 Transparency

The term transparency can have many meanings depending on the field. For example, in science, transparency refers to the degree to which a medium allows radiation to pass through. It can also refer to the quality of an object such as glass that can be seen through (Chen et al., 2022). In political and social contexts, opening up the flow of information and government processes allows for a democratic society's development. Through transparency, citizens can become increasingly engaged in the struggle to preserve their rights and demand government action. Transparency in the context of organizations, or organizational transparency, is a factor that can allow and/or improve the vision and management of processes and access, use, and storage of information for stakeholders (Camelo Rincón, 2020).

Processes execution and management as well as access, use, and storage of organizational information are carried out with software support. So, transparency also becomes a concern when projecting software (Leite and Cappelli, 2010). As a non-functional requirement, transparency must be considered in all stages of software design, so that there is clarity on how they provide transparency to organizational processes and information. Thus, transparency is not only in the final object but also in the entire information processing and treatment and must allow stakeholders (software developers, project managers, clients, and end-users) to answer their questions about the software during its life cycle (Cysneiros, 2013; Hosseini et al., 2016).

Considering related needs in the software context, Leite and Cappelli (2010) define transparency as the union of characteristics that contribute to its formation and the open information flow: (i) Accessibility: information about the software is available to the external environment; (ii) Usability: available information can be easily obtained and used; (iii) Informativeness: information is made available with expected quality; (iv) Understandability: external users can understand the available information; and (v) Auditability: external users can certify that the available information is trustworthy. To represent them, the authors developed a structure called SIG (Softgoal Interdependency Graph), which allows the identification of dependency relationships between the quality characteristics and their contributions to the operationalization of transparency.

## 2.3 Related Work

In our searches, we have not found many studies that focused on transparency in SECO. So, we have identified some studies in the literature that explored SECO and/or software development and mention transparency at any moment in the discussion of their results. Vegendla et al. (2018) investigated requirements engineering and quality attributes in SECO. The authors analyzed 44 studies and could infer that most of them have approached the requirements management, prioritization, verification, and traceability activities. They also highlighted that transparency is one of the quality attributes in SECO and depends on how open code is provided for platform extension. Transparency is an important non-functional requirement to maintain the ecosystem openness.

Setzke et al. (2019) performed a systematic literature review (SLR) to synthesize and integrate extant interdisciplinary research on the concept of platform openness. The authors analyzed 73 studies and identified five themes: measurement frameworks, implementation mechanisms, motivators for opening and closing platforms, trade-offs when designing openness, and the impact of changing openness on ecosystems. The authors noticed that few studies consider the transparency dimension of openness, such as technical documentation, communication with end users, or transparency of market mechanisms.

Dabbish et al. (2012) interviewed several core and peripheral users of GitHub, examining the value of transparency for large-scale distributed collaborations and communities of practice. They found that four keys features of visible feedback drove a rich set of inferences around commitment, work quality, community significance, and personal relevance. These inferences supported collaboration, learning, and reputation management in the community. For the authors, there is a potential for transparency to radically improve collaboration and learning in complex knowledge-based activities.

Obie et al. (2023) presented a preliminary study to investigate developers' perceptions and experiences related to human values, with a focus on the human value of transparency. The authors interviewed five experienced developers and conducted a thematic analysis to explore how developers perceive transparency and violations of transparency. Their findings reveal the significance of transparency as a fundamental

value in software development, with developers recognizing its importance for building trust. These findings contribute to the understanding of transparency in software development and provide insights for promoting ethical practices.

Despite not having transparency as their main focus, these studies point out the need of addressing transparency in SECO, given its impact on the openness and extension of the platform, which are important factors for attracting and retaining new actors in SECO. Therefore, this study seeks to fill that gap with an overview of how transparency has been treated and identify what are its conditioning factors within SECO.

### 3 Research Method

Following the guidelines of ACM SIGSOFT Empirical Standards (Ralph, 2021), this research method is characterized as exploratory and follows quantitative and qualitative approaches to data collection and analysis. This study aims to characterize the importance level of conditioning factors for transparency in SECO and extend the findings of our previous study (Zacarias et al., 2023). To do so, we defined a research method consisting of five steps, as shown in Figure 1: (i) Previous study findings; (ii) Planning, (iii) Execution; (iv) Data Analysis Procedures; and (v) Results. These steps are described next:

**(i) Previous Study Findings:** we conducted an SMS on scientific databases and digital libraries to review the state-of-the-art and characterize transparency in SECO. SMS are designed to give an overview of a research area through classification and counting contributions about the categories of that classification by searching the literature (Kitchenham and Charters, 2007; Petersen et al., 2015). After investigating the literature, we selected 23 studies for our analysis;

**(ii) Planning:** we conducted a field study with software developers to characterize the importance of conditioning factors for transparency in their interaction with a SECO. Field studies aim to investigate and understand how individuals who carry out a certain activity deal with practice and problem-solving within their respective contexts. A set of data collection techniques can be adopted in a field study, including questionnaires and interviews, for example (Singer et al., 2008). We conducted a pilot study with a PhD student in Computer Science to evaluate our interview guide;

**(iii) Execution:** we refined the protocol and conducted the interviews with 16 software developers who use GitHub, a platform to support project-based ecosystems, mainly OS-SECO (Lungu and Lanza, 2010; Liao et al., 2019). We aimed to deepen and detail the conditioning factors for transparency identified in the SMS from the developers' perspective. The interviews lasted an average of 25-27 minutes;

**(iv) Data Analysis Procedures:** we performed a coding process on the transcribed interviews to identify the software developers' perspective on the importance of conditioning factors of transparency in their interaction with a SECO;

**(v) Results:** the interviews also allowed us to identify more information about the benefits of transparency and the consequences of lack of transparency. So, as a result, we organized all the knowledge obtained into a conceptual frame-

work to be used as a foundation for systematic approaches to understanding transparency in SECO. All the details of the protocol for both studies are presented in the following sections.

#### 3.1 Systematic Mapping Study

To review the state-of-the-art of transparency in SECO, we conducted a SMS following the guidelines proposed by Petersen et al. (2015). This protocol is structured in five steps: (i) definition of research questions; (ii) search; (iii) study selection; (iv) data extraction; and (v) results.

##### 3.1.1 Definition of Research Questions

This mapping study aims to characterize transparency in SECO. To achieve this goal, we proposed one main RQ and derived four sub-questions (SQ):

- **RQ:** How is transparency in the SECO context characterized?
  - **SQ1:** What types of solutions are used to provide or assess transparency in SECO?
  - **SQ2:** What are the conditioning factors for transparency in SECO?
  - **SQ3:** What types of SECO processes must be transparent?
  - **SQ4:** What are the concerns for transparency in SECO?

These research questions aim to provide an overview of how transparency is approached within a SECO, considering the open flow of information for the SECO opening. So, SQ1 aims to summarize the types of transparency solutions used in the context of SECO, such as theory, model, method, practice, tool, or framework. SQ2 addresses factors that specifically contribute to transparency in SECO. Furthermore, SQ3 identifies the processes that must be transparent within a SECO. Finally, SQ4 points out the transparency concerns and how these aspects can influence SECO. Concerns refer to issues and challenges frequently highlighted in the literature or by software industry professionals regarding a particular environment or domain. Through concerns, researchers and professionals can identify and plan research opportunities (Motta et al., 2018). The anchors to our study are references Cataldo and Herbsleb (2010), Souza et al. (2020), and Herbsleb et al. (2016). Their results and proposals for future work were the basis for elaborating the questions and classification patterns.

##### 3.1.2 Search

The framework PICO (Population, Intervention, Comparison, and Outcomes), suggested by Kitchenham and Charters (2007), was used to identify keywords and formulate search strings from the questions. **Population:** In our context, the population comprises studies in the SECO field. **Intervention:** The intervention is transparency for the open information flow. **Comparison:** There is no clear comparison in the context of this study. **Outcomes:** The outcome is the studies

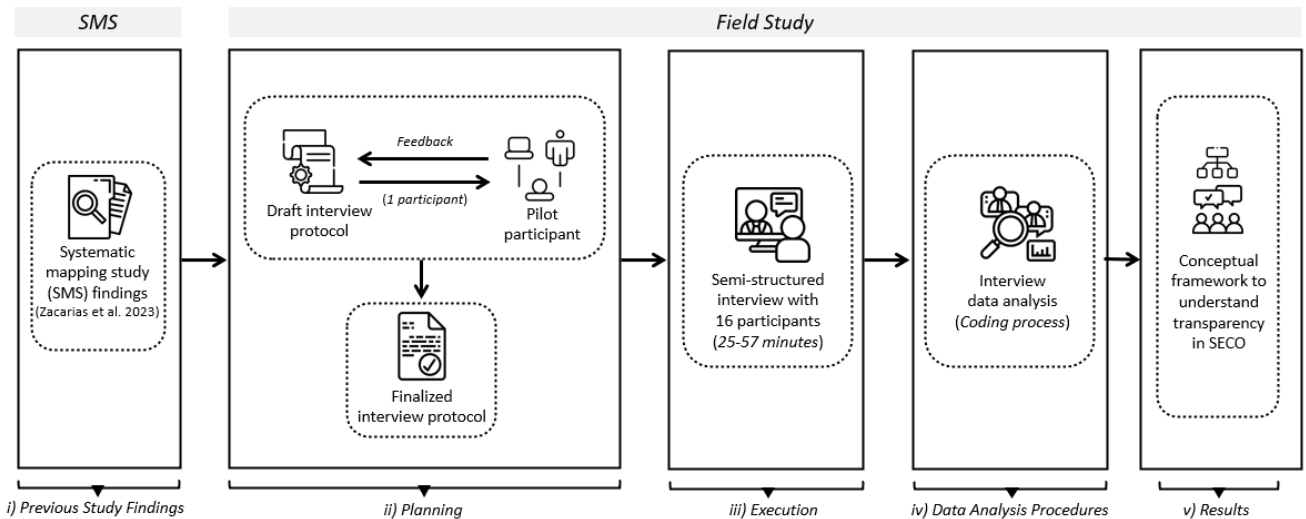


Figure 1. Overview of the research method of this extended study

that discuss solutions (theory, model, method, practice, tool, or framework) for transparency in SECO.

To create our search string, we joined the keywords representing the population and the intervention in the framework PICO. As our study concerns a mapping study, there was no specific comparison nor the need to limit the search space regarding outcomes, following the adopted search strategy by Villamizar et al. (2021). The keywords were then grouped into sets with their synonyms and considered to formulate the search string.

- **Set 1:** Scoping the search for SECO: “software ecosystem”, “SECO”, “information ecosystem”, and “ERP ecosystem” (synonyms based on the work of Manikas and Hansen (2013));
- **Set 2:** Search terms related to transparency and open information: “transparency” and “open” (to broaden the scope of search into the openness of SECO and information).

Those two sets were used together to form the base search string, which was run on the following databases: Scopus, Science Direct, IEEE Xplore, Engineering Village (Compendex), and ACM Digital Library. These databases have been selected based on the recommendations of Dyba et al. (2007). The search strings used for each database can be found in Table 1.

### 3.1.3 Study Selection

Below, we list the inclusion (IC) and exclusion (EC) criteria for the studies retrieved by the search string. During the filtering stages, we sought to identify studies that presented solutions for transparency in SECO.

- **IC1:** The study presents a discussion on transparency in a SECO.
- **IC2:** The study presents a discussion about the openness of SECO and information.
- **IC3:** The study presents solutions (theory, model, method, practice, tool, or framework) for transparency in SECO.

Table 1. Searches in databases

Database	Search
Scopus	TITLE-ABS-KEY (“software ecosystem*” OR “SECO” OR “information ecosystem*” OR “ERP ecosystem*”) AND (“transparen*” OR “open*”)
Science Direct	Title, abstract, keywords: (“software ecosystem” OR “SECO” OR “information ecosystem” OR “ERP ecosystem”) AND (“transparency” OR “open”)
IEEE Xplore	(“All Metadata”：“software ecosystem*” OR “All Metadata”：“SECO” OR “All Metadata”：“information ecosystem*” OR “All Metadata”：“ERP ecosystem*”) AND (“All Metadata”：“transparen*” OR “All Metadata”：“open*”)
Engineering Village	( (“software ecosystem*” OR “SECO” OR “information ecosystem*” OR “ERP ecosystem*”) AND (“transparen*” OR “open*”) ) WN ALL
ACM Digital Library	[[All: “software ecosystem*”] OR [All: “SECO”] OR [All: “information ecosystem*”] OR [All: “erp ecosystem*”]] AND [[All: “transparen*”] OR [All: “open*”]]

- **EC1:** The study content is not available in its entirety.
- **EC2:** The study is not a research article or a conference paper.
- **EC3:** The study is not primary.
- **EC4:** The study is duplicated.
- **EC5:** The study does not meet any of the inclusion criteria.
- **EC6:** The study has less than four pages.

The selection process consisted of seven stages: (1) Search execution; (2) Removal of duplicate studies; (3) 1st Filter: reading of title, abstract, and keywords; (4) 2nd Filter: reading of introduction and conclusion; (5) 3rd Filter: complete reading of the study; (6) Application of backward snowballing; and (7) Data extraction. To ensure the reliability of the results, two researchers analyzed each study in stages 3 to 6, and they discussed the differences with a third researcher

until a consensus was reached. This study was conducted in February 2023 with the support of Parsifal<sup>5</sup>, an online tool designed to help researchers perform literature reviews in Software Engineering. We used this tool for executing stage 2 when duplicate studies were removed in an automated way.

Next, we performed the filtering of studies in three stages, applying the necessary inclusion and exclusion criteria. In the first filter (stage 3), we read title, abstract, and keywords. In the second filter (stage 4), we read the introduction and conclusion and, in the third filter (stage 5), we read the full text of the remained studies. After performing the filtering stages, we applied backward snowballing (stage 6) to verify the selected studies' references and identify more studies to include in this research. Figure 2 shows the number of remaining studies in each stage. At the end of the process, we selected 23 (twenty-three) studies for data extraction.

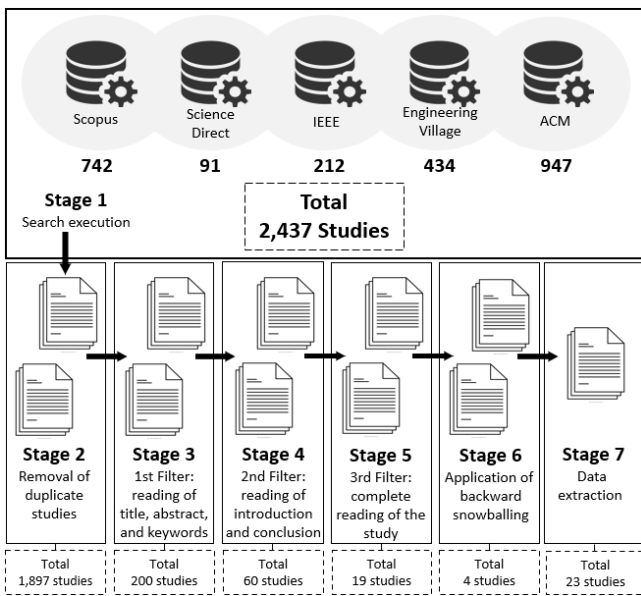


Figure 2. Results of the study selection process

### 3.1.4 Data Extraction and Analysis

The data was extracted in stage 7, when the full text was read and organized in Microsoft Excel®. We used the data to present an overview of the selected primary studies and to answer the research questions. The form was structured with the following fields: (i) **Study ID**: identifier; (ii) **Study title**: name of the study; (iii) **Author(s)**: names of the authors; (iv) **Year**: year of publication; (v) **Venue**: name of publication venue; (vi) **Country**: names of authors' countries; (vii) **Solution**: type of solution for transparency - SQ1; (viii) **Conditioning factors**: list of conditioning factors for transparency in SECO - SQ2; (ix) **Processes**: list of processes that must be transparent in SECO - SQ3; and (x) **Concerns**: list of concerns for research on transparency in SECO - SQ4.

We used open and axial coding methods for qualitative analysis when extracting categories from the selected studies, as presented in Corbin and Strauss (2014). In the open coding, we analyzed and extracted parts of text from the studies that brought some kind of relevant data to answer each

SQ and created codes to represent them. In the axial coding, we grouped the identified codes into categories (SQ1 - solutions, SQ2 - conditioning factors, SQ3 - processes, and SQ4 - concerns). Two researchers analyzed each study separately and then came together to compare the texts extracted and codes identified for the categories. If these codes were unable to help answer the research questions, the researchers discarded them. When there were divergences, they discussed them with a third researcher (expert). We provided an algorithm to demonstrate how we made the decisions throughout the analysis in the supplementary material located at <https://doi.org/10.5281/zenodo.10602816>.

## 3.2 Field Study

Following the recommendations of Singer et al. (2008) for executing the field study, we developed a process consisting of three steps: (i) planning; (ii) execution, and (iii) data analysis procedures.

### 3.2.1 Planning

In this planning section, we present details about the interview protocol, population characterization, and the pilot of this study.

**Research Question Definition:** The results obtained in the SMS provided us with a comprehensive view of the solutions, conditioning factors, processes, and concerns related to transparency in the SECO. To enrich this body of knowledge, we decided to go into the field to understand how software developers perceive SECO transparency in practice.

The conditioning factors for transparency represent elements, characteristics, or actions that can contribute to transparency in SECO and are related to other SMS findings. Therefore, we decided to consider them as the basis for our field study design. So, our objective was to verify the importance of the conditioning factors for transparency from the point of view of software developers in an OSSECO. To do so, we defined this RQ: *“What is the importance level of conditioning factors for transparency according to developers in the context of an OSSECO?”*

**Interview Protocol:** In order to answer our RQ, we organized the interview protocol into three stages. In the first step, the researchers provided the Informed Consent Form (ICF) to each participant. The participant should read and agree or disagree with the ICF, before starting the study. Whether the participant agreed, in the second step the researchers gave a presentation about the context of this research, explaining SECO concepts and presenting the conditioning factors for transparency identified in the SMS (see Section 4.1.3). Finally, in the third step, the researchers conducted a semi-structured interview with a set of questions about the characterization of the participants (see Table 2) and questions about the conditioning factors for transparency (see Table 3).

**Population Characterization:** Following the guidelines of Kitchenham et al. (2015), we defined the population profile for this study as software developers who use GitHub

<sup>5</sup><https://parsif.al/about>

**Table 2.** Questions regarding the characterization of field study participants

Questions	Answer Options
What is your professional career segment?	Academic, Industrial or Academic and Industrial
What is your most recent academic qualification?	High school or technical degree, Bachelor degree, Specialization degree, Master degree or Doctorate degree
How long have you been a professional developer?	(Open Question)
How long have you been using GitHub?	(Open Question)
For what purpose do you use GitHub?	Personal project, Academic project or Professional project (industry)

to manage their projects. GitHub is a platform to support project-based ecosystems, mainly OSSECO (Lungu and Lanza, 2010; Liao et al., 2019). We adopted the non-probabilistic convenience sampling to define the sample due to the impossibility of accurately defining the total number of participants eligible for this research.

Looking for professionals who fit the defined profile, we sent invitations to participate to email lists of undergraduate and graduate courses in the area of Computer Science from different regions of Brazil and to software developers through Whatsapp<sup>6</sup> groups. Furthermore, we used the snowball sampling technique, in which the first participants nominated other professionals to participate in the interview.

**Pilot:** After the elaboration of the interview protocol, we conducted a pilot with a PhD student in Computer Science, who had experience in SECO and requirements engineering. We sent the invitation to participate in the pilot by e-mail. The invitation email contained information about the researchers of this study, the objective and duration of the interview session, a link to the ICF, and request availability for scheduling (day and time).

The purpose of the pilot was to evaluate the protocol and the conduct of the interviews. The session was held on Google Meet<sup>7</sup> that allows to record the session. We asked the pilot participant to fill out the ICF. Next, we gave a brief presentation about the study objectives and concepts related to SECO and transparency. Then, we asked the questions about the conditioning factors for transparency.

During the pilot, the participant reported a lack of understanding of some questions. In Q2, when we asked about having access to information, the participant replied that he did not understand what kind of information we were referring to. In Q8, the participant could not understand what kind of reliability we were referring to. Based on these situations, we decided to put a description in these two questions about what kind of information (interface and platform documentation) and what kind of reliability (about the reliability of references, content authors, updates etc.). The final version of the artifacts used in the field study can be viewed in full at <https://doi.org/10.5281/zenodo.10602816>.

### 3.2.2 Execution

We sent out invitations to email lists to Computer Science courses in Brazil and software developers on the social network WhatsApp, as mentioned in Section 3.2.1. Through the invitation email, we provide the link to access the ICF. ICF included acceptance of participation in the research, agreement with the conditions established in the term, and authorization to record the interview. We emphasized to participants that they had the right to withdraw from the interview at any time and that their responses would be deleted. Finally, we also asked potential participants by email to inform their availability (day and time) to schedule the interview session.

The interviews lasted an average of 25-27 minutes, as estimated in the pilot and following the structure presented in Section 3.2.1. At the end of the interview, we asked participants if they agreed with the list of conditioning factors for transparency presented or if they would like to suggest any changes to the material presented, adding a new factor or removing one that they did not agree with. In total, we interviewed 16 software developers, by convenience and following the concept of “saturation”. According to Guest et al. (2006), saturation is reached when by running a new set of interviews, no emerging data emergent. Thus, initially, we conducted 10 interviews, and after analyzing them, we noticed they were still insufficient, not having reached such saturation. Next, we conducted 2 more interviews, and after analyzing them, we noticed that the eleventh was the last that added a new point to this research, which was not addressed by the other participants. Finally, we conducted the last 4 interviews, in which we noticed that they did not present new points in the research, thus reaching saturation.

### 3.2.3 Data Analysis Procedures

The data obtained in the interviews were analyzed qualitatively. To analyze the data, we performed an open and axial coding approach inspired by the initial procedures of the Grounded Theory. Thus, we adopted an open coding approach, in which interviews were coded inductively (bottom-up) (Charmaz, 2006). To do so, we thoroughly read participants’ responses, dividing the transcripts into **coherent units** (sentences or paragraphs), and subsequently, we defined a set of **focused codes** that captured the most frequent and relevant factors in the participants’ perceptions.

After open coding, we used axial coding, as described by Charmaz (2006), to group the codes into **categories**. To do so, we organized the excerpts from the interviews and the codes into a document, enabling, in several iterative cycles, researchers to follow up on the main findings through discussions. During the coding process, we wrote memos for the codes and categories, and notes were made about the relationships between the codes. Table 4 shows an example of the coding process for one transcript with resulting codes and categories.

It is worth mentioning that data was analyzed by two researchers who work in the areas of SECO, software engineering, and transparency. The results of this process were evaluated iteratively by two other PhD researchers with experience in software engineering and qualitative studies for

<sup>6</sup><https://www.whatsapp.com>

<sup>7</sup><https://meet.google.com/>



**Table 3.** Questions about the conditioning factor for transparency on GitHub

Conditioning Factors (CF) for Transparency	
<b>ID</b>	<b>CF1: The existence of communication channels between actors and keystone</b>
Q1	Have you ever used any platform channel to communicate with co-workers, other developers, or even platform support? How important do you think communication channels are for collaborative development?
<b>ID</b>	<b>CF2: Information about platform made available in an accessible way</b>
Q2	Can you access this information from different devices? Have you had the experience of using GitHub both on mobile and the web? Did you notice any differences in the information made available in these two versions? Do you find it easy to access any information on GitHub? Have you ever had experience trying to access information that was not available? How do you evaluate the importance of this factor for your performance at GitHub?
<b>ID</b>	<b>CF3: The actors' understanding of SECO information</b>
Q3	Do you think the information in the interface or documentation is very detailed and concise? Have you ever found any information that you did not understand? How much do you think finding information you do not understand could hinder your project? How much do you think finding information you do not understand could hinder your project?
<b>ID</b>	<b>CF4: The quality of platform information provided by a keystone</b>
Q4	Is the information available on the platform interface or in its documentation clear? Do you notice if they are updated? Have you ever noticed any incorrect or incomplete information on the platform or even in the documentation? In general, how do you evaluate the impact of this information made available on your work as a developer and your interaction with the platform?
<b>ID</b>	<b>CF5: The usability of interfaces with platform documentation</b>
Q5	How do you evaluate GitHub's interfaces? Do you think they are intuitive or easy to use? Have you noticed any usability problems on the platform? Do you think usability is important to you when you search for information on GitHub?
<b>ID</b>	<b>CF6: The auditability of platform processes and information</b>
Q6	How do you check the traceability of information on the platform? Do you find it complex? Do you think how the platform provides information helps you control your projects? Do you think it is very clear and well-presented? Regarding this issue of going through the information available on the platform, how important is this for developers?
<b>ID</b>	<b>CF7: Visualization of the evolution of projects in SECO</b>
Q7	Do you think the platform has mechanisms that allow you to monitor the progress of projects, whether in terms of changes and updates that are made? Do you think it allows that? Do you think these mechanisms are informative? What benefits does it provide for your project?
<b>ID</b>	<b>CF8: Reliability of information provided by a keystone</b>
Q8	Do you notice that the platform provides means that allow you to verify the reliability of the information that is made available, whether in its interface or its documentation? Regarding the reliability of information, do you think this factor is important for your projects?

**Table 4.** Example of the coding process

<b>Coherent unit:</b> "Due to the platform's usability issues, some members needed help to keep up with the project's progress, leading to suboptimal productivity. Occasionally, someone fails to deliver their work due to a usability flaw." D1	
<b>Focused Code</b>	<b>Category</b>
Low developer productivity	Consequences of lack of transparency

at least 15 years. The interview transcripts and other artifacts from the coding process are available in full at <https://doi.org/10.5281/zenodo.10602816>.

## 4 Results

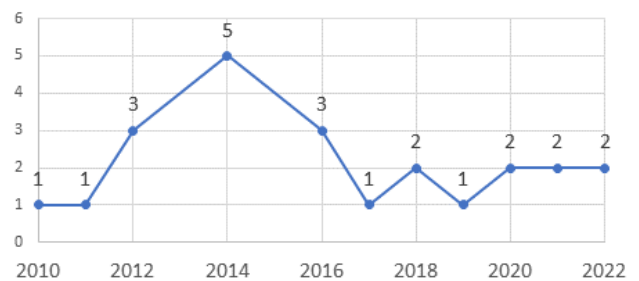
In this section, we present the results of the SMS and then the results of the field study. We also detail the conceptual framework for understanding transparency in SECO, developed from the results obtained in both studies.

### 4.1 Results of Systematic Mapping Study

#### 4.1.1 Demographic Data

The majority of the selected studies were published in journals (nine studies, 39% of the total), and conferences (eight studies, 35% of the total). We identified three studies (13% of the total) published in workshops and three studies (13% of the total) published in symposiums. Figure 3 shows the distribution of selected studies over the years. We noticed that at least one annual publication has been published from 2016 until 2022, reaching the peak in 2014 with four studies. Furthermore, it is important to highlight that there were no studies before 2010. This fact can be explained by the novelty

of the term, which has been more widely explored in recent years.



**Figure 3.** Number of studies per year

As shown in Figure 4, relevant primary studies were predominantly conducted in the USA, with six studies, followed by Brazil, with five studies. Sweden appeared in four studies, while Germany and the Netherlands in three each. Canada and Finland have been represented in two studies. Finally, China, the Czech Republic, India, and Switzerland were present in one study each. A study may have researchers of different nationalities and therefore the sum does not correspond to 23.

About the SECO types, we could identify 15 studies addressing open source (e.g., Python, GNOME), six studies addressing proprietary (e.g., SAP, JobTech), and three studies addressing hybrid (e.g., Android, iOS) ecosystems. It is noteworthy that only one study addresses both open and proprietary SECO.

The complete list of 23 studies can be seen in Table 5, in descending order by year of publication. Identification (ID) codes will be used for referencing studies throughout the



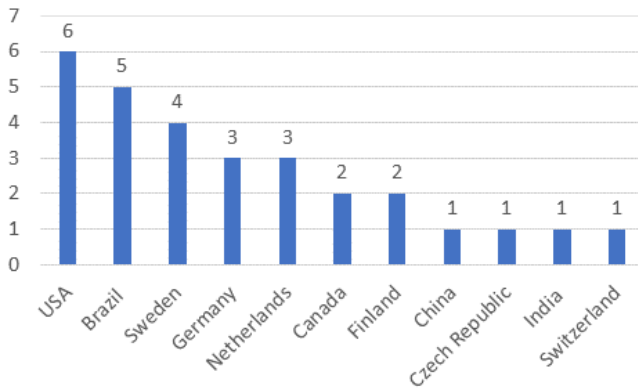


Figure 4. Number of studies per country

next sections. Each study received the identifier (S), followed by a numerical identification (S01-S23). Besides information about the author(s), country, and year of publication, we added a column to describe which solution was used for transparency in SECO. After screening selected primary studies in this SMS, collected data were analyzed to answer each SQ. The raw data and all the steps necessary to reproduce the research are detailed in the supplementary material located at <https://doi.org/10.5281/zenodo.10602816>.

#### 4.1.2 SQ1: What types of solutions are used to provide or assess transparency in SECO?

We were inspired by the taxonomy defined by Shaw (2003) to categorize the solutions identified in the selected studies: (i) theory: a system of ideas intended to explain something; (ii) model: structure or taxonomy for a problem area; (iii) method: a particular form of procedure for accomplishing something; (iv) practice: new or better way to do some task; (v) tool: device that embodies a technique; and (iv) framework: a support structure to do something.

The results showed that Tools/Frameworks (S06, S07, S09, S10, S12, S16, S18, and S19), Models (S03, S04, S13, S14, S20, and S22), Practices (S05, S08, S17, S21, and S23), Methods (S01) and Theory (S11) are ways to promote transparency in SECO. Only two studies did not present some solution (S02 and S15). Figure 5 illustrates how we categorize the solution presented in each study. Most solutions are related to using Tools/Frameworks. It is important to notice that the solutions presented in this SMS may have been detailed or just cited in the selected studies. We present more details about each type of solution in the following subsections.

##### A) Tools/Frameworks

S06 proposes the **user eXperience evaluation tool (T2-UXT)** that automatically monitors the user experience in the Web environment and generates visual artifacts and indicators to document it. S07 and S10 present an **instrument/questionnaire for assessing transparency in SECO portals**, following accessibility, usability, informativeness, understandability, and auditability. S09, in turn, presents an **anomaly detection mechanism in commits**, which enables developers to access information about relevant changes in other projects.

S12 proposes the **intelligent software assurance and**

**monitoring (ISAM) provider**. ISAM issues early warnings as customer applications and open source components evolve. S16 presents the **application transparency framework (AT)**. This framework protects users from malicious applications and targeted attacks and adds transparency to the application's signing process.

S18 brings a **framework for measuring the SECO health**. With the framework, it is possible to quantify an ecosystem's health, and one can gain helpful knowledge for strategic decision-makers. S19 presents the **OSCOMM framework for the SECO openness**. The main contribution of the OSCOMM framework is to make explicit and document the elements required to build and sustain open source ecosystems.

Intending to support researchers and professionals in developing new solutions or reusing and understanding those identified in this research, we summarize open source tools and frameworks. It is important to note that not all identified tools and frameworks are software-intensive or automated.

T2-UXT (S06) is an open source tool and the project is expanding so that other developers and researchers can use it and contribute to its evolution. The anomaly detection mechanism in commits (S09) is also code and both the back end and front end are available on GitHub. Application Transparency (AT) framework (S16) is also presented as open source, but the authors do not mention in their study how other researchers can contribute to the project.

We can also analyze these tools and frameworks by grouping them according to their scope of use. S06, S07, and S10 present solutions aimed at improving the design of SECO portals from the transparency perspective. S09, S12, and S16 provide mechanisms to improve the visualization and monitoring of contribution indicators in OSSECO. S18 provides metrics for analyzing and monitoring SECO health and S19 provides a conceptual framework for promoting the opening of a SECO.

##### B) Models

S03 has defined a **conceptual model of an open data ecosystem (ODE)**. This model can be used to analyze the openness of data of a SECO. S04 presents a **conceptual model for SECO governance** that establishes how to make partnerships and how accessible the ecosystem must be to balance the use, development, and commercialization of products and services.

In S13, the **requirements analysis and management for benefiting openness (RAMBO) model** is introduced for requirements analysis and management to support open innovation in SECO. S14 defines three **collaboration models for SECO**: product-line engineering (PLE) for internal software ecosystems (ISECO), platform reuse for ISECO, and decoupled PLE for ISECO. These models provide a framework for practitioners to discuss architectural measures in advance.

S20 presents the **open software enterprise (OSE) model** that determines the degree of openness of a software-producing organization (SPO). S22 introduces the **clopeness assessment model (CAM)**, which provides insight into the openness of an organization, in turn enabling more informed business decisions and helping an organization to maintain SECO openness or closure.

Table 5. List of selected studies

ID	Title	Author	Country	Solution	Year
S01	Are you of value to me? A partner selection reference method for software ecosystem orchestrators	Beelen et al. (2022)	The Netherlands and Finland	Method	2022
S02	Collaboration in software ecosystems: A study of work groups in open environment	Chen et al. (2022)	China	None	2022
S03	Open Data Ecosystems - An empirical investigation into an emerging industry collaboration concept	Runeson et al. (2021)	Sweden	Model	2021
S04	Software Ecosystems Governance - An Analysis of SAP and GNOME Platforms	Oliveira and Alves (2021)	Brazil	Model	2021
S05	Privacy in Software Ecosystems - An Initial Analysis of Data Protection Roles and Challenges	Valença et al. (2020)	Brazil and Germany	Practice	2020
S06	T2-UXT: A Tool to Support Transparency Evaluation in Software Ecosystems Portals	Souza et al. (2020)	Brazil	Tool/framework	2020
S07	An Instrument for the Evaluation of Transparency Mechanisms in Software Ecosystem Portals	Meireles et al. (2019)	Brazil	Tool/framework	2019
S08	Continuous clarification and emergent requirements flows in open-commercial software ecosystems	Knauss et al. (2018)	Sweden and Canada	Practice	2018
S09	Identifying unusual commits on GitHub	Goyal et al. (2018)	India and USA	Tool/framework	2018
S10	Building a questionnaire to evaluate transparency in software ecosystem portals	Meireles et al. (2017)	Brazil	Tool/framework	2017
S11	Building a Socio-Technical Theory of Coordination: Why and How (Outstanding Research Award)	Herbsleb (2016)	USA	Theory	2016
S12	Intelligently Transparent Software Ecosystems	Herbsleb et al. (2016)	USA	Tool/framework	2016
S13	Requirements Analysis and Management for Benefiting Openness	Linåker and Wnuk (2016)	Sweden	Model	2016
S14	Architecture Challenges for Internal Software Ecosystems: A Large-Scale Industry Case Study	Schultis et al. (2014)	Germany	Model	2014
S15	A Quantitative Analysis of Developer Information Needs in Software Ecosystems	Haenni et al. (2014)	Switzerland	None	2014
S16	Hey, NSA: Stay Away from My Market! Future Proofing App Markets against Powerful Attackers	Fahl et al. (2014)	Germany and Czech Republic	Tool/framework	2014
S17	Openness and requirements: Opportunities and tradeoffs in software ecosystems	Knauss et al. (2014)	Sweden and Canada	Practice	2014
S18	Proposed Metrics on Ecosystem Health	Monteith et al. (2014)	USA	Tool/framework	2014
S19	From proprietary to open source - Growing an open source ecosystem	Kilamo et al. (2012)	Finland	Tool/framework	2012
S20	Shades of gray: Opening up a software producing organization with the open software enterprise model	Jansen et al. (2012)	The Netherlands	Model	2012
S21	Social Coding in GitHub: Transparency and Collaboration in an Open Software Repository	Dabbish et al. (2012)	USA	Practice	2012
S22	Clopenness of systems: The interwoven nature of ecosystems	Molder et al. (2011)	The Netherlands	Model	2011
S23	Architecting in Software Ecosystems: Interface Translucence as an Enabler for Scalable Collaboration	Cataldo and Herbsleb (2010)	USA	Practice	2010

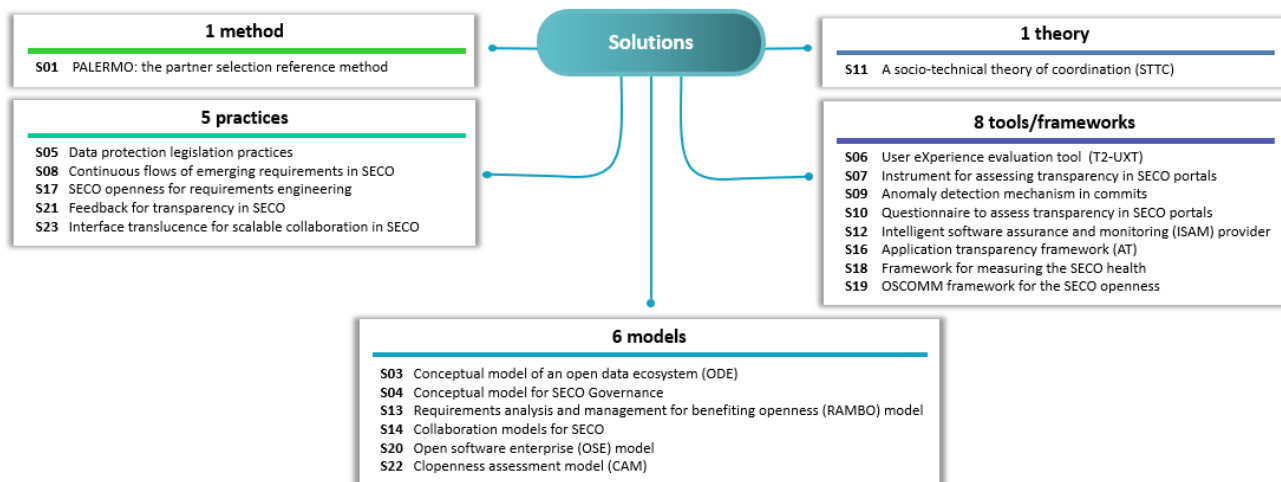


Figure 5. Solutions used to provide transparency in software ecosystems

We can also analyze these models by grouping them according to their scope of use. S03, S13, S20, and S22 present solutions focused on how to promote the opening of SECO in a transparent, strategic, and efficient way. S13 and S14 focus on governance approaches in SECO, mainly for transparency regarding architecture and requirements management.

C) Practices

S05 discusses **data protection legislation practices** that SECO should follow and how privacy requirements affect software development in such ecosystems. S08 presents the use of open communication channels to provide transparency to handle **continuous flows of emerging requirements in SECO**. S17 emphasizes the importance of **SECO openness**

**for requirements engineering**, as it provides transparency to the process.

S21 highlights the importance of the practice of **feedback for transparency in SECO**. The study assesses the value of transparency for large-scale distributed collaborations and communities of practice through commits and profiles on GitHub. S23 presents **interface translucence for scalable collaboration in SECO**. The study highlights that interface translucency increases transparency in SECO by conveying only the relevant organizational information to ecosystem members.

We can also analyze these practices by grouping them according to their scope of use. S08 and S17 focus on transparency to improve requirements flow and management in SECO. S05 and S23 bring practices aimed at the level of

access to different information in SECO. S21 addresses the importance of implementing feedback channels to improve project management in SECO.

#### D) Methods

S01 presents **PALERMO: the partner selection reference method**. The focus is the creation of a structure within a company, describing the partner selection process, training new colleagues, and providing transparency for partners. As such, potential partners can know the conditions they must fulfill to become a partner in a SECO.

#### E) Theories

S11 presents the **socio-technical theory of coordination (STTC)**. The theory assists actors gain insight into transparent environments that can help developers cope with the scale and decentralized organizational structure of SECO by being able to take advantage of a huge set of libraries, frameworks, and other code artifacts available in these environments, such as SECO.

#### 4.1.3 SQ2: What are the conditioning factors for transparency in SECO?

This SQ seeks to summarize which factors specifically contribute to transparency in SECO, describing the main characteristics that are related to this context. Analyzing the selected studies, the topics mentioned were grouped into 8 factors (Table 6), written by the authors, according to their influence on transparency. Studies S01, S05, S09, S11, S12, S16, and S20 did not specify any conditioning factors in their content.

**Table 6.** Conditioning factors for transparency in SECO

ID	Conditioning Factors	Studies
CF1	The existence of <b>communication channels</b> between actors and keystone	S02, S03, S04, S07, S08, S17, and S21
CF2	Information about platform made available in an <b>accessible</b> way	S06, S07, S10, S14, S15, and S22
CF3	The actors' <b>understanding</b> of SECO information	S06, S07, S10, S15, S19, and S22
CF4	The <b>quality of platform information</b> provided by a keystone	S06, S07, S10, and S18
CF5	The <b>usability</b> of interfaces with platform documentation	S06, S07, S10, and S18
CF6	The <b>auditability</b> of platform processes and information	S06, S07, and S10
CF7	<b>Visualization of the evolution of projects</b> in SECO	S13, S21, and S23
CF8	<b>Reliability of information</b> provided by a keystone	S03 and S13

CF1 highlights that communication channels (e.g., forums and contact emails) are an important instrument for capturing and monitoring the implementation of requirements coming from the developer community. CF2 emphasizes that all actors must be able to access the information available about a common technological platform. CF3 describes that actors must be able to understand the information provided about a technological platform. CF4 describes that the quality of platform information provided by a keystone helps actors, espe-

cially developers, to more easily learn how SECO processes work.

CF5 explains that good usability of interfaces contributes to better interaction of actors with the documentation provided. CF6 explains that actors should be able to certify platform processes and information provided are correct and true. CF7 highlights that actors must be aware of changes and updates made in a common technological platform. CF8 emphasizes that actors must trust the information provided by a keystone. Therefore, these conditioning factors contribute to the open flow of information within a SECO. The more factors present, the more easily transparency can be achieved in a SECO.

In our analysis, we can highlight the recurrence of characteristics defined by Leite and Cappelli (2010) for transparency (accessibility, usability, informativeness, understandability, and auditability) in studies S06, S07, and S10. These studies explore SECO portals and try to associate these characteristics of Web portals in the SECO context to assess the degree of transparency.

It is also possible to notice the recurrence of factors related to communication (S02, S03, S04, S07, S08, S17, and S21), visibility, and access to information about projects and codes contained in the SECO platform (S13, S21, and S23). Transparent environments and social coding platforms help developers stay up-to-date on changes in the development and maintenance phases of a project, as highlighted in S9. There is potential for such transparency to radically improve collaboration and learning in complex knowledge-based activities, as mentioned in S21.

Transparency is also related to the degree of openness and security of a SECO (S13). For a keystone to be able to coordinate such an open business model, it is necessary to define to which degree each type of actor will have access to information to avoid overload and guarantee the preservation and security of confidential data. According to S22, in addition to transparency, availability, accessibility, reciprocity, and licensing factors must also be considered in this process.

Finally, S18 presents a transparent vision based on information about SECO health, which influences ecosystem growth. S03 and S13 state that the information provided must be true and address how this affects the credibility and trust of actors in a SECO.

#### 4.1.4 SQ3: What types of SECO processes must be transparent?

This SQ seeks to summarize which SECO processes should be transparent and how they are being explored or addressed by the solutions identified in SQ1. Table 7 presents a list of these processes that were identified in the studies, written by the authors. It should be noted that this is a view based on the selected studies. The levels of transparency for these processes depended on the business context and organizational objectives of each SECO. In any case, we want to highlight that these processes need attention when talking about transparency in SECO.

The process of accessing documentation, source code, and tools is the most cited among the selected studies (P1). This is due to the developers' need to have access to information

**Table 7.** Processes that must be transparent in SECO

ID	Processes	Studies
P1	Access to documentation, source code, and tools	S06, S07, S15, S16, S18, S19, and S22
P2	Access to information about code in repositories	S07, S09, S10, S12, S15, and S21
P3	Communication channels between actors and keystone	S01, S02, S06, S07, S08, and S11
P4	Processes related to SECO governance	S02, S04, S14, S18, and S20
P5	Access to information about requirements flow	S08, S13, S17, and S20
P6	Processes related to data collection, processing, and sharing	S01, S03, S05, and S20
P7	Access to information about SECO architecture	S20 and S23

required to know and learn how to use the SECO platform, in addition to allowing for interactivity in forums.

Regarding accessing information about code in repositories (P2), it helps actors, especially developers, to be aware of changes and updates of software technologies on the platform, avoiding compatibility problems with products. Another highlighted process is related to the communication channels between actors and keystone (P3). Transparency in communication between actors and keystone is a key element in coordinating SECO activities, mainly in maintaining the engagement that keeps SECO active.

Processes related to SECO governance (P4) need to be transparent as it is a key success factor for SECO as fun activities occur simultaneously. Regarding access to information about the flow of requirements (P5), the transparency of this process supports the management of demands in SECO, allowing a keystone to demonstrate credibility in meeting the needs of actors.

The transparency of the processes related to the collection, processing, and sharing of data (P6) contributes to the information being perceived as reliable and true, providing credibility in SECO. Finally, access to information about SECO architecture (P7) can improve the vision of the technical aspects that allow the expansion of a SECO platform. For third-party developers to be able to make some code contribution to the common technological platform, they need to know and understand the technical and architectural aspects of that platform.

#### 4.1.5 SQ4: What are the concerns for transparency in SECO?

SQ4 aims to identify concerns for research on SECO transparency. Following the main concerns (open source ecosystems, governance, analysis, openness, quality, and software architecture) related to SECO presented by Barbosa et al. (2013), we could identify five concerns for transparency in SECO from the studies (S01, S02, S03, S06, S08, S17, and S19). We regarded the characteristics related to transparency defined by Leite and Cappelli (2010) to state them. Although many concerns are related to software development, others are specific to requirements engineering in SECO due to the complexity and diversity of interests among actors in this context.

**Attracting and engaging potential partners (actors) to OSSECO:** the number of actors interacting with OSSECO directly influences the ecosystem's health indicators, i.e. pro-

ductivity, robustness, and niche creation. Therefore, it is essential to define processes to attract and engage more actors, mainly developers, to OSSECO (S01 and S02). One of the strategies is related to the *informativeness* characteristic related to transparency, which considers the quality of the information available about the common technological platform. This information allows actors to get to know and learn OSSECO processes, improving their experience and, potentially, the number of contributions on the platform.

**Handling large volumes of data in OSSECO:** there are large volumes of data transiting through OSSECO, due to its open characteristic that allows contributions of different actors who can enter and leave the ecosystem at any time. Often, keystones face problems related to access and use of this data by them. Although on the one hand, free access to data can be positive from the perspective of transparency, on the other hand, undefined access to a large volume of data can make it difficult to interact with the common technological platform. Therefore, it is necessary to build visual data tools that add transparency resources and investigate how to moderate the permissions and access levels of different actors to data shared in an OSSECO (S03 and S05), contributing to better *usability* and *understanding* of this data.

**Guaranteeing access and understanding of keystone's actions and strategies in SECO:** software product management strategies must be established by a keystone and must be communicated and understood by other SECO actors. To achieve this, there is a need to create an information architecture, that is, an organized information structure that guarantees *accessibility* and *understanding* by these different actors. This architecture contributes to improving the interaction of actors with information about the common technological platform since all its actions and strategies for the use of human and technical resources can be understood and followed by them. Consequently, this concern can improve the trust and credibility of a keystone and facilitate governance actions (S4).

**Defining the level of transparency of software architecture in SECO:** when we talk about defining the level of data and information transparency of a SECO, we consider that, in some ecosystems such as PSECO and hybrid SECO, some data and information must be protected or have access restricted following the keystone's business rules. This must be taken into account at the same time that open data needs to have its access facilitated and guaranteed by actors. Therefore, to support decision-making on the level of transparency agreed in a SECO, it is necessary to propose a software architecture model for the common technological platform that makes it possible to operationalize the different levels of transparency of data and information. One of the solutions to this concern is the concept of interface translucency in SECO architecture (S23), which allows managing *accessibility* to data through an architectural organization of the technological platform. However, this concept still needs to be further investigated.

**Maintaining effective and transparent communication channels:** communication channels (e.g., forums, issues, emails, etc.) are one of the main sources of demands from the developer community in a SECO. These demands may become future requirements for improving common techno-

logical platforms. Therefore, there is a growing need for research focused on establishing transparent communication channels and better solutions to collect feedback from different actors. In addition to facilitating the sending of community demands, communication channels must provide resources that enable keystone to manage all the requirements that emerge from them, while at the same time being accountable (*auditability*) for their decisions to other actors (S08 and S17).

## 4.2 Results of Field Study

After running the field study, we organized and analyzed data from all 16 participants to answer this study's RQ: "What is the importance level of conditioning factors for transparency according to developers in the context of an OSSECO?"

### 4.2.1 Demographic Data

Table 8 presents a summary of information characterizing the profile of the participants. We assigned an identifier (ID) to each software developer, following the order of the interviews carried out (D1 to D16), to identify them throughout this article.

Regarding professional careers, four participants work in an academic career, ten in an industrial career, and two in both academic and industrial careers. Concerning academic qualifications, two participants have a high school or technical degree, seven have a bachelor's degree, six have a specialization degree, and one has a master's degree. The average experience as a professional developer among participants is approximately six years and the average experience with GitHub is approximately five years. It is worth mentioning that developers only considered experience as professional developers when answering the questionnaire, but some of them had previously used GitHub for studies. Finally, regarding the purpose of using GitHub, eight participants answered that use it for academic projects, ten for professional projects, and 14 for personal projects.

### 4.2.2 Conceptual Framework for Understanding Transparency in SECO

The main outcome of our research is the proposed conceptual framework for transparency in SECO (Figure 6). The conceptual framework emerged from the SMS and the field study data. Parahoo (2014) states that a conceptual framework is fundamental in organizing research and establishing an argument based on foundations and concepts (Green, 2014). According to Kon et al. (2015), a conceptual framework assists in the understanding of the elements that make up the research objective and the relationships between them. In addition to this unifying vision, another purpose of a conceptual framework is to promote the development and improvement of academic and industrial practices (Issac et al., 2004).

The central concept of the proposed conceptual framework is **transparency in SECO** in which software functions are disclosed to users and developers. On the left side of the framework, we list the two categories: **conditioning factors for transparency** and **processes that must be transparent**.

These categories emerged by consolidating the results of the SMS that were later verified with software developers in the field study. The existence of the elements from these categories can contribute to improving transparency in SECO. The category listed below is related to **solutions to provide transparency in SECO**. It is a consolidation of the results presented in Figure 5, which were identified in the SMS.

On the right side of the framework, we present other two categories that are directly influenced by **transparency in SECO**. The **benefits of transparency** consist of positive advantages of promoting transparency in SECO and the **consequences of lack of transparency** refer to some problems that are generated when transparency is not enough in SECO. The category of benefits of transparency emerged from the SMS and was later discussed with the software developers in the field study. Concerning the consequences of lack of transparency, these results were mentioned only in the field study, during the discussion and evaluation of the previous categories. The following sections detail the four categories addressed in the field study.

### 4.2.3 Conditioning Factors for Transparency

As described in Section 3.2.3, we coded iteratively the responses of the participants (software developers) in order to answer the RQ: "What is the importance level of conditioning factors for transparency according to developers in the context of an OSSECO?". This process allowed us to capture the perceptions of 16 software developers regarding the conditioning factors for transparency identified in the SMS. Figure 7 illustrates the importance attributed to each factor by the interviewed developers. In the following sections, we will delve into a detailed presentation of the significance of each factor for these developers.

#### (CF01) - The existence of communication channels between actors and keystone

The communication channels for 12 (75%) developers are considered very important and important for four (25%) developers. The communication channels provided by the platform are crucial for communicating changes made on the platform. The developers mentioned the example of being informed about changes made on the platform. This instance underscores the importance of effective communication on platforms such as GitHub, particularly between various actors and the keystone within the ecosystem. D4 stated: "There was a change in the security aspect, and they informed us about it, mentioning that something would become outdated. In this specific instance, this communication was important". These channels enable developers to maintain a comprehensive development environment in a single space and stay connected with other developers within that environment. D1 explained that: "It is very important because it is a way to interact with other developers in a single space for the complete development of a project". According to some developers, asynchronous communication contributes to collaborative development, making the software development process easier. D10 shared that: "It is very important to have asynchronous communication in collaborative development for code control and advancing related activities. So, if it were not easy to communicate with the team, develop-



Table 8. Characterization of the developers

ID	Professional Career	Academic Qualification	Experience as Professional Developer	Experience with GitHub	Purpose of using GitHub
D1	Academic	High school or technical degree	3 years	3 years	Academic project
D2	Industrial	High school or technical degree	2 years	8 years	Personal project, professional project (industry) and academic project
D3	Academic and Industrial	Bachelor’s degree	3 years	4 years	Personal project, professional project (industry) and academic project
D4	Academic	Bachelor’s degree	4 years	4 years	Professional Project (industry) and academic Project
D5	Industrial	Specialization degree	5 years	3 years	Personal project and academic project
D6	Industrial	Specialization degree	8 years	6 years	Personal project, professional project (industry) and academic project
D7	Academic	Specialization degree	5 years	4 years	Personal project and professional project
D8	Academic	Bachelor’s degree	5 years	5 years	Personal project and academic project
D9	Industrial	Bachelor’s degree	3 years	6 years	Personal project and professional project (industry)
D10	Industrial	Bachelor’s degree	3 years	5 years	Personal project, professional project (industry) and academic project
D11	Industrial	Specialization degree	4 years	5 years	Personal project, professional project (Industry) and academic project
D12	Industrial	Bachelor’s degree	3 years	5 years	Personal project and academic project
D13	Industrial	Specialization degree	7 years	6 years	Personal project, professional project (industry) and academic project
D14	Academic and Industrial	Master’s degree	4 years	7 years	Personal project, professional project (industry) and academic project
D15	Industrial	Specialization degree	21 years	5 years	Personal project and academic project
D16	Industrial	Bachelor’s degree	12 years	3 years	Personal project and academic project

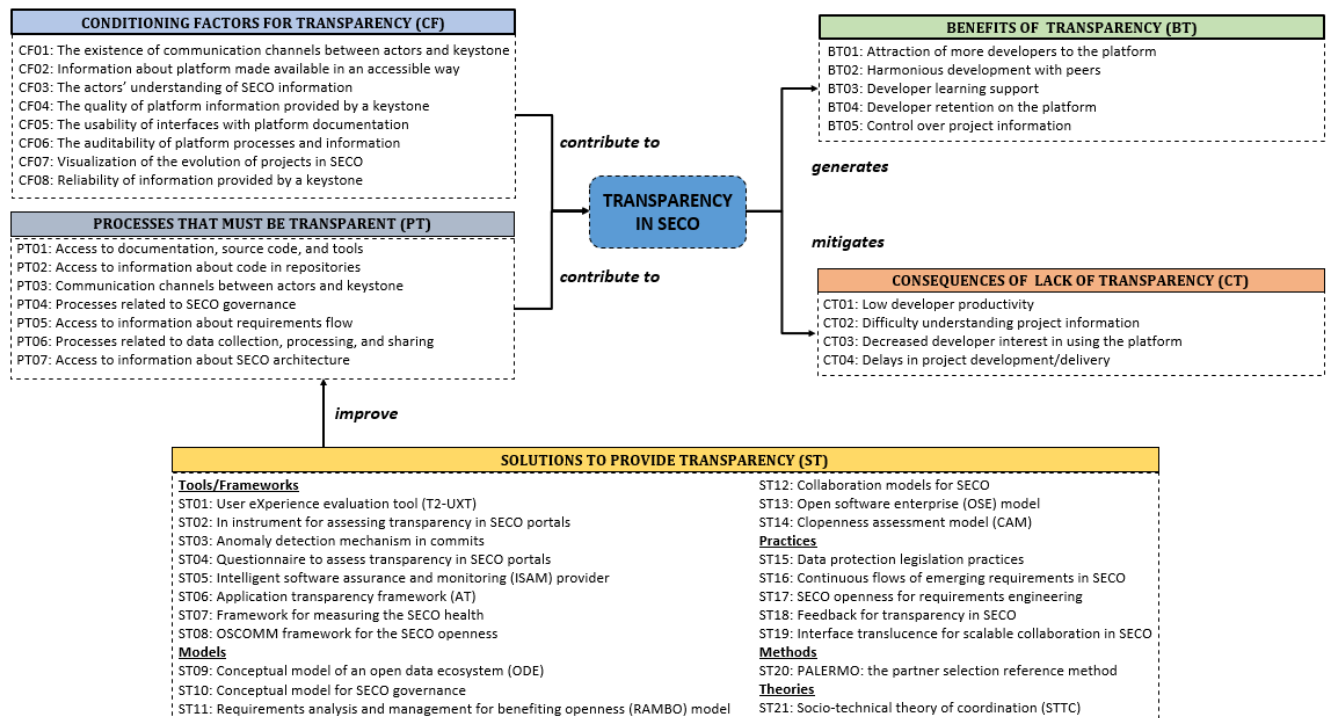


Figure 6. Conceptual framework for understanding transparency in SECO

ing software might be more difficult”. This perspective emphasizes the need for effective and accessible communication for both the platform and collaborative software development projects.

**(CF02) - Information about platform made available in an accessible way**

The availability of information accessible for nine (56.25%) developers is considered very important, and important for seven (43.75%) of them. Developers emphasize

that if information is easily accessible, obtaining and utilizing the platform more effectively becomes possible. D6 stated that: “It is very important because the presentation of information might be a determining factor in choosing between platforms. I notice this when there is ease in using a tool”. D7 added that: “I consider it extremely important to make information more accessible because the simpler the information, the better its access. If it is accessible, the developer can obtain that information and use the platform”.

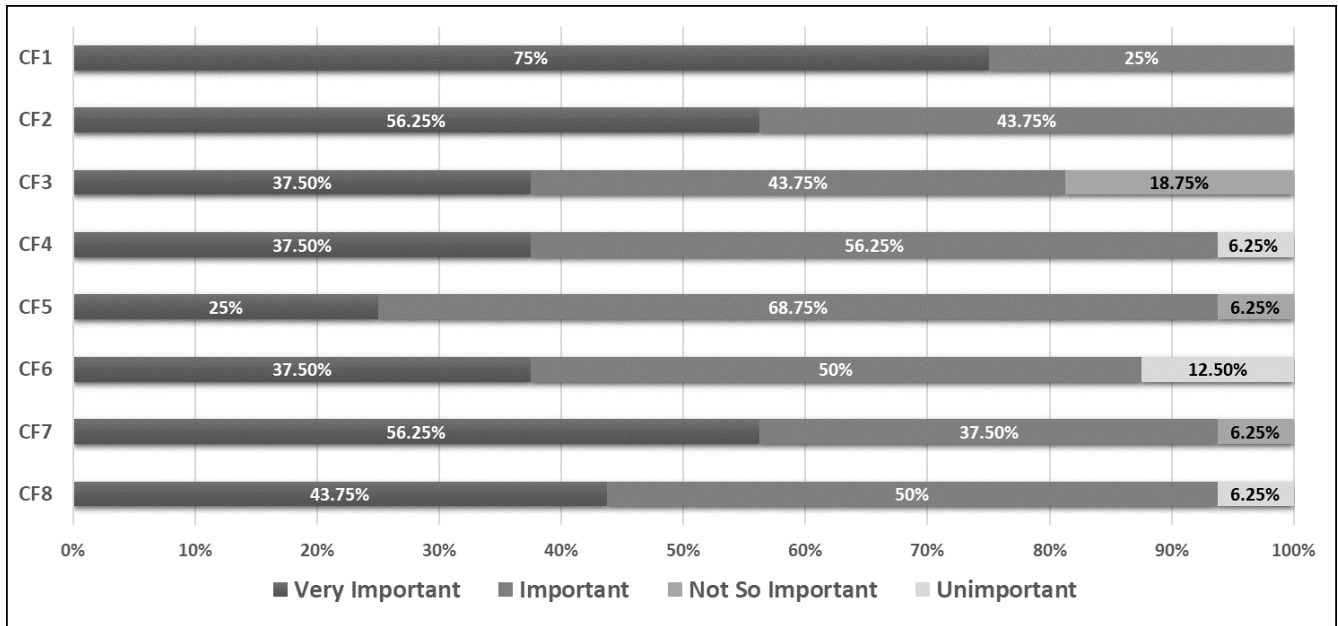


Figure 7. Developers’ evaluation of the conditioning factors for transparency

The developers stated that searching for or acquiring desired information can significantly influence the DX in projectors. If there are difficulties in this process, it can result in a negative experience, impairing developer satisfaction. D4 stated that: *“It is important because if there is difficulty in finding information or getting the desired information, it creates a poor experience for the developer, impairing satisfaction. Hence, the ease of obtaining information is crucial to our projects”*. Furthermore, developers mention that the clear and accessible presentation of information can be a determining factor in choosing different platforms.

**(CF03) - The actors’ understanding of SECO information**

Understanding the information presented on the platform for six (37.50%) developers is considered very important, important for seven (43.75%) of them, and not so important for three (18.75%) developers. The developers highlight a challenge faced by platforms such as GitHub that present a large amount of information within a single interface. Information overload can severely compromise the quality and usability of the platform, hindering users’ understanding of project information. Excessive transparency can thus undermine information comprehension. Balancing transparency and usability is crucial for developers to ensure an effective platform experience. D1 stated that: *“A lot of information is presented in a single interface, which compromises the quality of the information, as it impairs the platform’s usability and understanding of project information. Excessive transparency can, therefore, jeopardize the understanding of information”*.

The lack of understanding also generates problems in the project and the development of systems, harming both the project and the team. D11 shared that: *“I think it could harm my project and the team. When the documentation is unclear about a process or instruction, this situation causes the developer to create bad practices in using the platform, as they cannot fully learn the processes”*. D4 mentioned that: *“This can hinder the project, as lacking detailed or clear information can slow development”*. The developers also empha-

size the importance of clarity and organization in technical documentation, as the absence of these elements can negatively impact the developer’s experience. D5 explained that: *“When dealing, for example, with poorly organized documentation, sometimes we give up using a specific tool and look for other alternatives”*.

**(CF04) - The quality of platform information provided by a keystone**

The quality of information for six (37.50%) developers is considered very important, important for ten (56.25%) of them, and only one (6.25%) developer considers it unimportant. The quality of the information is a determining factor so that developers maintain interest in the platform and feel safe to use it. D4 argued that: *“I think it has a decisive impact on the continued use of the platform. Suppose it is information that does not have quality and is not accurate. In that case, this ends up causing the developer to lose interest because it generates some problems and maybe even insecurity”*. Another developer confirms this factor. D15 mentioned that: *“The quality of the information is very important because the quality meets the users’ satisfaction. If the information is of good quality, the developers will be more interested in using the platform”*.

In this way, when the quality of information is present in its interface and documentation, it generates comfort for the developer to use the platform. D12 mentioned that: *“The information in the documentation and interface is precise. The documentation is a great bonus that makes the quality of information important”*. Additionally, the quality of the information helps the developer obtain more accurate results. D8 shared that: *“Quality helps to obtain accurate information, as is the example of libraries”*.

**(CF05) - The usability of interfaces with platform documentation**

The platform interface usability for four (25%) developers is considered very important; 11 (68.75%) of them consider it important, and only one (6.25%) developer considers it not so important. Developers highlight the interplay between us-



ability and information availability within platforms such as GitHub. Developers' acknowledgment of the learning curve and the platform's focus on information access underscores the importance of ongoing efforts to refine and improve usability. By enhancing usability, GitHub can foster a more intuitive and user-friendly environment, maximizing its effectiveness for developers. D7 stated that: *"It is not easy to use or intuitive at this initial stage. However, it is a matter of learning; it could be better in that aspect. GitHub does not prioritize offering 200% usability. It aims to display all the data on the screen so you can work with it, but it will take some time to get used to. GitHub's focus is on information availability. So, for the platform, it's more important to have access to information than usability, ensuring that information is available at your fingertips"*.

A good interface helps in the quick understanding of information. It helps the developer dedicate more time to their projects instead of trying to solve problems with the platform itself. D10 stated that: *"Yes, it is very important, especially in terms of speed of understanding of information. Often, we do not have much time to look for ways to solve platform-related problems because we need to focus on work"*. According to the developers, the interface's usability is crucial in making the programming world accessible to new developers. D2 stated that: *"Usability is essential, especially for collaborating and onboarding new individuals into an ecosystem. Therefore, usability is important for us to make this world of direction accessible to new developers"*.

#### **(CF06) - The auditability of platform processes and information**

The auditability of platform processes and information is considered very important for six (37.50%) developers, important for eight (50%) of them, and only two (12.50%) consider it unimportant. The developer's statement underscores the significance of audibility, control, and community engagement within the GitHub ecosystem. By providing developers with the tools and platform to manage their projects transparently and participate in a collaborative community, GitHub facilitates the exchange of knowledge, fosters innovation, and drives the growth of the developer community. D7 shared that: *"GitHub is excellent because it provides a community that can assist you while you contribute. These two points are essential: managing their projects, the auditability of project processes, and being part of a community where you can contribute. It is a two-way street, receiving and offering help to drive the growth of this community"*. The developers emphasize also that this factor is important for tracking and understanding what is happening with the code. The statement suggests that code tracking is essential to comprehend its origin, purpose, and context within a project. D11 shared that: *"It is important because we can track what is happening with our code. We can try to understand where it came from, why it came, and why it was made. So we need to understand the context of a project"*.

The developers also mentioned that controlling the project may not be as challenging in small teams. However, in larger projects, mechanisms that assist in verifying the traceability of information and project control are crucial. D14 said that: *"I think it is very important. In a small team, having control of the project is not that difficult. In the case of super*

*projects, having mechanisms that help check the traceability of information and control the project is crucial"*. D3 added that: *"Knowing where to find information is very important, especially when we work on large projects"*.

#### **(CF07) - Visualization of the evolution of projects in SECO**

The possibility of following the evolution of projects for nine (56.25%) developers is considered very important, important for six (37.50%) of them, and not so important for just one (6.25%) developer. The visualization of projects helps developers who work on large projects to have control over the contributions arranged in the projects present in the platform environment. D13 argued that: *"It is very important, especially when working on large projects involving several people. A key GitHub factor is visualizing what happens on each project and which people contribute to the code"*. So, this factor is important when the developer is part of a big team. D4 said that: *"Yes, it is important because it makes it easier to follow changes, even more so in a big team. Controlling versions also makes it easier to identify and revert changes when necessary. This helps prevent conflicts between developers"*. This visualization possibility allows the developer to view project architecture management, which contributes to projecting management. D6 shared that: *"This brings a project architecture management benefit and provides a view to projecting management through metrics"*.

Moreover, the developers mentioned that visualization of the evolution of projects in SECO allows for a comprehensive view of all contributions made to the repository, enabling the assessment of each team member's activity duration. D14 said that: *"Having this complete view of the project is very important. We can have an overview of all the contributions to the repository and evaluate each project member's activity time"*.

#### **(CF08) - Reliability of information provided by a key-stone**

The reliability of the information for seven (43.75%) developers is considered very important, important for eight (50%) of them, and only one (6.25%) is considered unimportant. One of the interesting points is that the developers consider that the information's reliability is not the platform's responsibility but that of the developers who are consumers of the information. D14 mentioned that: *"It is hard to see that information on GitHub is reliable, especially from third parties. However, it is not the platform's role to verify the information's reliability; it is the developer's"*. D15 shared that: *"The reliability factor should be the responsibility of the developers who disclose information. The platform cannot have control over everything that is disclosed in the environment. However, reliability is very important"*. D16 shared that: *"The reliability of the information is not GitHub responsibility, but that of the developers who consume the information. However, reliability is important for platform developers"*.

The developers highlighted the importance of having access to information about the organization of a project and the people involved related to reliability. D12 shared that: *"It is very important because seeing all the information about the organization of a project and the people involved influences my decision-making and my work"*.

#### 4.2.4 Processes that Must Be Transparent

During the field study, some developers mentioned processes that must be transparent in SECO. We confirmed five processes that must be transparent based on the participants' responses (Figure 6) and detailed them below.

##### (PT01) - Access to documentation, source code, and tools

Some developers emphasized the importance of the relationship between documentation quality and the interface in people's ability to reproduce procedures efficiently. Clarity in the provided information is crucial in this context. Developers assert that detailed and quality documentation is fundamental to empowering users to replicate procedures in their work environments, contributing significantly to effectiveness and productivity. D5 shared that: *"That is more about knowing whether the tool is reliable and if and offers security. Sometimes, this reliability is tied to the interface or documentation, while other times, it is associated with a specific piece of code to verify its origin and accountability"*. D11 shared that: *"I notice that the impact is on how to use the platform in the best possible way. So when we have documentation that teaches step by step with quality, people can reproduce that on their machines or in the work they are doing"*.

##### (PT02) - Access to information about code in repositories

The developers emphasized the importance of checking the last update date when analyzing a repository, especially when considering adding libraries or code. This underscores the need for clarity in code differences to prevent them from hindering the team from keeping up with changes in the project. Additionally, this highlights the importance of transparency and practical code understanding, especially in dynamic and multifaceted work environments. D2 explained that: *"It is one of the first things I do, without a doubt, when I look at any repository. What is the date of its last update? When adding any library or code, we must see its support status and how updates are. We are suspicious of the project when we see something is too old and has stayed the same"*. D3 shared that: *"In an environment with multiple simultaneous activities across various work fronts such as development, testing, and database, clarity in the code difference becomes essential to prevent it from being a barrier to the team keeping up with the changes in the project"*.

##### (PT03) - Communication channels between actors and keystone

The developers mentioned the need for communication channels on the platform, emphasizing their importance for various users, whether for security or convenience. This underscores the importance of communication channels in development platforms. D4 shared that *"Having these communication channels on the platform is crucial. Some users use the platform's communication means, either for security reasons or simply for convenience. Therefore, these features need to be accessible to all users"*.

##### (PT06) - Processes related to data collection, processing, and sharing

The discussion on the importance of transparency points to data collection, processing, and sharing processes, high-

lighting its relevance, especially in a teamwork context. One developer mentioned the need for notifications to ensure all members know about integration and information. D4 shared that *"This situation becomes more relevant when working in a team, especially since it is possible to carry out integrations and share the information within projects. Thus, we must start receiving notifications about these integrations"*.

##### (PT07) - Access to information about SECO architecture

The developers mentioned how information about the evolution of the project's architecture goes beyond the technical aspect, providing valuable insights for various areas, including the business domain. The emphasis on the importance of information about the platform and architecture highlights how transparency in these areas is crucial for the efficient progress of developers during the project lifecycle. D6 shared that: *"A benefit of managing the evolution of the project's architecture is the possibility of gaining valuable insights. This includes working on the effort required for the project, among other metrics that can be used in the business area. It is possible to obtain a range of information grounded in this data"*. D10 shared that: *"In general, I believe the information about the platform's version and architecture is important so developers can progress easily during project development"*.

The remaining processes that must be transparent in SECO (PT04: Processes related to SECO governance and PT05: Access to information about requirements flow), identified through SMS, were not mentioned by the developers but are described in Section 4.1.4. Furthermore, the category "solutions to provide transparency", presented in Figure 6, was not confirmed during the field study, as it involves specific solutions derived from studies identified in the SMS. The **solutions to provide transparency** are described in Section 4.1.2.

#### 4.2.5 Benefits of Transparency

During the interviews, participants reported 5 benefits of transparency in SECO. Throughout the following, we detailed each benefit of transparency in SECO.

##### (BT01) - Attraction of more developers to the platform

The developers emphasized the importance of easy access to information, highlighting that it optimizes time dedicated to projects and fosters a culture of collaboration by facilitating assistance among developers. This, in turn, attracts more professionals to use the platform, underscoring the notion that transparency and collaboration are essential factors for the success and growth of a developer community. D1 shared that: *"It is very important because when we manage to have access to information in an accessible way, this helps us to dedicate more time to projects, help other developers, and attract more developers to use the platform"*. D5 shared that: *"Even when attempting to push a project to remote repositories like GitHub, we must reach out if we encounter any issue with a third-party code or platform. Thus, this communication is crucial for maintaining and attracting new developers to the platform"*.

##### (BT02) - Harmonious development with peers

The developers highlighted the relevance of transparency

and usability in collaborative development with other programmers. In essence, developers underscore the synergy between transparency, usability, and collaborative development, indicating that collaborative development based on people's feedback is essential for the success and growth of significant projects in software development. D13 shared that: *"Large projects, such as React, Angular, and Flutter, emerged due to this collaborative development based on people's feedback. Transparency enables harmonious development alongside other developers"*.

#### **(BT03) - Developer learning support**

The developers mentioned that transparency boosts individual learning by allowing the learning of diverse tools and promotes collective learning by facilitating the exchange of knowledge among developers. Thus, developers emphasize the importance of transparency as a key factor for learning and active participation in various projects. D1 stated that: *"It is fundamental because, through this, I have learned to use various tools and engage in numerous projects, primarily due to the ease of accessing information and transparency"*. D8 mentioned that: *"The transparency allows us to learn from other developers because we can leverage code from other developers and thus add value to our project"*.

#### **(BT04) - Developer retention on the platform**

Developers mentioned transparency as a crucial factor in overcoming challenges faced by new users with little experience on the platform. As a result, the availability of documentation and tutorials about the platform can promote a better understanding of the environment, thereby encouraging developers' continued involvement and engagement. D7 mentioned that: *"A newcomer with limited experience using the platform may need help navigating and accessing everything immediately. If resources such as videos, documentation, or study materials are available, that person could better master the environment. This helps maintain developers using the platform. So transparency is very important"*. D2 said that: *"It is very important, especially for using the platform. Thus, transparency is important to keep developers using the platform"*.

#### **(BT05) - Control over project information**

Developers mentioned that the most significant benefit of this factor is for project managers, arguing that they can track the progress of all demands. The ability to visualize the number of commits, conduct code reviews, and analyze the working time of each developer are pointed out as specific advantages. This suggests that transparency benefits understanding of the project and provides valuable insights for project management. D10 shared that: *"I believe the biggest benefit of this factor is to the project managers because they can see the evolution of all the demands. They can see the number of significant commits and how to conduct code reviews. They can also analyze the entire working time of each developer"*. D7 shared that: *"It is extremely important to have transparency in the information process, as it enables control over your project. Creating a backup and shaping the project according to preferences becomes feasible when all information is clear"*. Developers mentioned that the most significant benefit of this factor is for project managers, arguing that they can track the progress of all demands. The ability to visualize the number of commits, conduct code reviews,

and analyze the working time of each developer are pointed out as specific advantages. This suggests that transparency benefits understanding of the project and provides valuable insights for project management. D10 shared that: *"I believe the biggest benefit of this factor is to the project managers because they can see the evolution of all the demands. They can see the number of significant commits and how to conduct code reviews. They can also analyze the entire working time of each developer"*. D7 shared that: *"It is extremely important to have transparency in the information process, as it enables control over your project. Creating a backup and shaping the project according to preferences becomes feasible when all information is clear"*.

### **4.2.6 Consequences of Lack of Transparency**

We asked developers about what consequences of the lack of transparency in SECO. We identified 4 consequences of lack of transparency. The right side of Figure 6 presents the identified consequences of the lack of transparency. Below, we detail each one of them.

#### **(CT01) - Low developer productivity**

Developers have mentioned that usability and transparency issues on the platform can impact developers' productivity. The lack of clarity and transparency can lead to task delivery failures, resulting in potential negative impacts on the schedule and project completion. The developers' feedback underscores the importance of an intuitive user interface and transparent information for project success. Such challenges can be overcome with improvements in usability and transparency, contributing to more efficient collaboration and consistent work delivery by project team members. D1 shared that: *"Due to the platform's usability issues, some members needed help to keep up with the project's progress, leading to suboptimal productivity. Occasionally, someone fails to deliver their work due to a usability and transparency flaw"*. D1 shared that: *"Lack of transparency can contribute to lower developer productivity, as people feel demotivated when they do not find a transparent interface that helps them move forward during the project"*.

#### **(CT02) - Difficulty understanding project information**

Developers mentioned that the excessive amount of information presented in a single interface hinders the quality and usability of the information. Thus, developers state that this compromises the quality and impairs the platform's usability, making project comprehension challenging. This underscores the importance of achieving a balance in the transparency of information. D1 shared that: *"Much information is presented in a single interface, which compromises the quality of the information and impairs the platform's usability and understanding of project information. Excessive transparency can, therefore, jeopardize the understanding of information"*. D3 shared that: *"I have encountered situations where the text was mixed between English and Portuguese. Additionally, I encountered confusing information, leading to misunderstandings and making it harder to understand the project"*.

#### **(CT03) - Decreased developer interest in using the platform**

Developers emphasized the importance of accuracy, transparency, and quality of information on the platform. They pointed out situations where the lack of these criteria can create uncertainties for the developer, influencing their decision to continue using the platform. This highlights the significance of information transparency not only for the efficient execution of tasks but also for maintaining the engagement and interest of developers. D4 stated that: *“If the information is not accurate, transparent, or quality, it might lead to a loss of interest for the developer in the platform, causing certain issues. Such situations can create uncertainty for the developer, which is decisive for continuing to use the platform”*. D4 added that: *“If there are frequent bugs or inconsistencies in the information, it affects my confidence and interest in continuing to use the platform”*.

#### (CT04) - Delays in project development/delivery

The developers mentioned the critical importance of transparency and understanding of information within the development team, emphasizing how its absence can have significant consequences, including project delays and impacts on delivery. This discussion underscores the need for effective communication and mutual understanding, ensuring success and efficiency in project execution. D7 shared that: *“Involving a team can bring problems because a lack of understanding of information and transparency can lead to actions that harm and delay system development. In turn, this affects the system and the entire team”*. D7 shared that: *“I think it delays the project. If they have some information that, no matter how clear, the person cannot understand, they will have to research and study it to try to understand it. Therefore, the time spent on this activity can impact the delivery of the project”*.

### 4.2.7 Guidelines for Using the Conceptual Framework

Overall, the main goal of our proposed framework lies in providing an overview of how scientific literature approaches the topic of transparency in SECO with the results obtained, evaluated, and refined based on software developers' perspective. Transparency in SECO has received little attention from the scientific community and the lack of studies focused on this topic makes it more difficult for researchers to perform studies without a consolidated conceptual basis. Therefore, our proposal is the first step for systematizing this knowledge and encouraging the scientific community to further investigate this topic given its importance in the software industry.

The conceptual framework can be used in practice from an Ask-Plan-Act process inspired by the guidelines of Greiler et al. (2023). The framework may be used as a foundation for systematic approaches developed to assist in the understanding of transparency in SECO. To do so, we suggest a three-step process outlined below. It is important to notice that the application flow presented next can be adjusted to fit the specific needs and circumstances of the research context.

For illustrative case purposes, let's consider the following context: a group of researchers wants to understand what causes delays in project development in SECO. They consider that this problem may be related to transparency in SECO. Thus, they can use our proposed conceptual frame-

work to help them design their research.

**Ask:** The first step to understanding transparency in SECO consists of identifying what can affect it from the actors' perspective in this context. For example, in our illustrative case, the group of researchers wants to understand what causes delays in project development in SECO which in our conceptual framework is considered a consequence of lack of transparency in SECO (CT04). The next step is to analyze the framework's left side which describes the conditioning factors for transparency and processes that must be transparent in SECO. Then, researchers need to ask the actors which SECO processes may be the origin of the problem from their perspective. Our conditioning factors can serve as prompts for gathering feedback through structured mechanisms, such as surveys, or unstructured methods, such as retrospectives and one-on-one meetings.

**Plan:** After collecting feedback, the responses must be evaluated to determine which conditioning factors and SECO processes require prioritization for improvement from a transparency perspective. Let's consider in our illustrative case that software developers reported that communication problems were impacting the progress of projects. In this case, it is mainly related to CF1 (The existence of communication channels between actors and keystone) and PT03 (Communication channels between actors and keystone) from our conceptual framework. Considering this information, researchers can plan a solution focused on these elements.

**Act:** In this step, our conceptual framework can be used as a reference to monitor and evaluate the results of the solution applied in the SECO context. It is important to highlight that actions to implement transparency might vary depending on each SECO context, i.e., there is no general solution to the underlying problems. Our framework suggests some solutions identified in previous studies so that researchers can replicate them or develop their solutions based on them. The conditioning factors can also be used to derive measures for assessing and monitoring the success of improvement efforts and their benefits. For example, in our illustrative case, researchers can develop a solution based on ST18 (Feedback for transparency in SECO), including practices of continuously collecting feedback from the software developer community about what delays the development of the projects. This is one way to improve communication (CF01 and PT03) in the SECO context. Monitoring the success of improvement solutions is essential to improve platform transparency and mostly DX. Once the planned improvement solutions are implemented, this three-step process can be repeated to continue progressing on this matter.

We highlight that this is an initial version of the conceptual framework for understanding transparency in SECO, developed based on the systematic studies that were performed in this research. This structure can be enriched and expanded by the SE community as new studies are conducted on this topic, with the identification of new elements or new categories for the framework.

## 5 Discussion and Implications

Research on the topic of transparency has several aspects, as its definition varies according to the area of study. This article aims to characterize conditioning factors for transparency in SECO. In this context, we adopt the concept of transparency from the software perspective since SECO main purpose is to develop products and services based on a common technological platform.

During the analysis of the 2,437 studies retrieved in our database searches in the SMS, we observed that the scientific literature still pays little attention to this subject. Despite being a fundamental element for the dynamics of relationships among multiple actors present in a SECO, several studies only mention transparency but do not deepen a discussion about the topic in SECO, bringing few insights to researchers in the field. Therefore, our final set has only 23 studies, which allowed us to gather relevant data to answer our questions.

Considering the impacts of transparency in SECO and the low number of selected studies with a real focus on the subject, it reveals the need to investigate this topic in more depth. The importance of transparency in SECO has gained more strength as the need to attract and keep developers actively contributing to a platform has become essential for SECO sustainability (Meireles et al., 2017, 2019; Jansen, 2020).

Based on this need, we conducted the field study to understand and characterize, in greater depth, the conditioning factors for transparency in SECO. By analyzing them from the perspective of software developers who work in a SECO, we are able to view them more pragmatically, in addition to confirming these findings of the literature.

Based on the responses of the 16 participants, we can highlight that all factors are very important or important for most software developers. The following factors were considered very important by more than half of the developers for their performance at the GitHub platform: (CF1) the existence of communication channels between actors and keystone; (CF2) Information about platform made available in an accessible way; and (CF7) Visualization of the evolution of projects in SECO.

Initially, we identified that for developers to remain on the platform it is necessary that information be made available in an accessible way (CF2) and the platform has interfaces with good usability to facilitate that access (CF5). In addition, the quality of information available (CF4) is a determining factor for them not to lose interest in using the platform, as well as the reliability level of that information (CF8). Otherwise, this can contribute to software developers going away over time, especially beginners.

Many developers pointed out that some of these conditioning factors of transparency on the platform can directly influence a project's success or failure. Problems in access to information (CF2) or in platform usability (CF5) can delay project execution, as developers can spend a lot of time looking for important information on the platform for their progress. The information needs to be presented with the expected quality (CF4) and mainly correct and concise to facilitate its understanding (CF3), as it can lead to environment misuse, in GitHub's case, the versioning environment.

Compared with Dabbish et al. (2012), which investigated

the value of information transparency on GitHub, this study results corroborate that transparency improves the work quality of developers and it is inserted in the propagation of community sense since it directly impacts the arrival and permanence of new developers. Inferences from the information provided also have a proportional impact on learning, engagement, and DX while using the platform, in complex activities environments such as GitHub.

We could also verify in both studies that transparency is addressed according to the keystone's business context and the relationship with its developer niche in each SECO. For example, in a PSECO, some information and processes may have access constraints for third parties to ensure the security and privacy of sensitive data. In an open SECO, in which much information is available, it is important to establish different levels of transparency to avoid information overload when users access it. Some participants in the field study mentioned that they did not know how GitHub managed projects' data regarding access level and privacy.

Corroborating with Manikas and Hansen (2013), this need is allied to SECO openness, aiming at its expansion and consolidation in the market. It has been a real challenge for a keystone. Considering transparency of SECO information and processes is crucial to attract new actors, our conceptual framework (Figure 6) can be a good tool to help keystones in strategic actions in SECO. For example, when facing a contributor loss problem at SECO, keystones can have an overview of critical points for transparency using the framework. They can check whether the problem's origin is a lack of a conditioning factor or an issue with some process and which kind of consequences they should avoid. It is worth highlighting that although we report on solutions to provide transparency described in Section 4.1.2, we cannot claim correlation or causal relationships across the constructs captured by our framework. We hope future work will expand our framework and build theories of improving transparency in SECO, not just describing or predicting it.

Considering the number of mentions, the studies selected in this SMS and the responses in the field study made it possible to identify the three main related topics to the extracted categories in all research questions and directly impacted by transparency: access to information, communication channels, and requirements engineering. Then, we present some perceptions and insights regarding future research on transparency in SECO from this discussion.

### 5.1 Access to Information

As highlighted in CF2 (S06, S07, S10, S14, S15, and S22) and P1 (S06, S07, S15, S16, S18, S19, and S22), access to information about the platform documentation, source code, tools, or requirements is one of the main ways to add transparency in a SECO. By opening their platform architectures, keystones seek to bring more developers to contribute to their ecosystems. The participants in the field study mentioned that platforms, such as GitHub, might not be so easy for beginner developers. Therefore, keystones must organize the information available to facilitate its access (CF2) through easy and intuitive interfaces (CF5) to facilitate the learning of use for these beginner developers. Likewise, the quality

of the information provided (CF4) will facilitate understanding (CF3) and this learning process and, consequently, attract more and more developers. The participants in the field study mentioned that platforms, such as GitHub, might not be so easy for beginner developers. Therefore, keystones must organize the information available to facilitate its access (CF2) through easy and intuitive interfaces (CF5) to facilitate the learning of use for these beginner developers. Likewise, the quality of the information provided (CF4) will facilitate understanding (CF3) and this learning process and, consequently, attract more and more developers. However, the participants highlighted the challenge of GitHub. Information overload poses a dual threat to platform usability and quality. Firstly, it can compromise the platform's usability by overwhelming users with too much information, leading to confusion and frustration. When users are overwhelmed with excessive data, they may struggle to find the specific information they need, resulting in a suboptimal user experience. Secondly, information overload can impede users' understanding of project information. Excessive transparency, while often viewed as a positive trait, can paradoxically lead to diminished comprehension.

Addressing the challenge of information overload is essential for platforms such as GitHub to maintain usability, quality, and effectiveness. By carefully balancing transparency and usability, GitHub can create an interface that empowers users to efficiently navigate project information and collaborate effectively, ultimately enhancing the overall platform experience. Hence, these developers need to have access to all these artifacts so that they can perform development activities autonomously. This first contact is crucial for contributing to a SECO. Moreover, the quality of the available content is fundamental since the difficulty faced by third-party developers in the first contact can push them away from the platform, as also highlighted by Setzke et al. (2019).

## 5.2 Communication Channels

As mentioned by Manikas and Hansen (2013), the SECO level of openness strongly affects social networks and their actors. Therefore, communication channels are critical to aligning keystone's perspectives with the needs of its developer community. That said, the communication channels were highlighted with a condition - CF1 (S02, S03, S04, S07, S08, S17, and S21) and a process that must be transparent - P3 (S01, S02, S06, S07, S08, and S11) in a SECO.

The communication channels available on the platform (CF1), the possibility of viewing project evolution (CF7), and the information reliability (CF8) provided are transparency factors that impact project management and collaborative development. The participants in the field study believe that CF2 is important to concentrate all the storage and exchange of information in a single environment. This improves project management and tracking of implemented changes.

Transparency in communication between actors and keystone is a fundamental element in coordinating SECO activities, mainly in maintaining the engagement that keeps a SECO vibrant. The open flow of information generates more credibility for a keystone (S03 and S13) since it values con-

tact between different actors and encourages exchanges between them, generating a sense of community over the SECO platform.

## 5.3 Requirements Engineering

With a wide variety of actors, it is natural that different sets of needs and demands arise from each group. Such sets become the requirements for platform evolution. As mentioned by Vegendla et al. (2018), transparency in the flow of requirements contributes to maintaining the quality of a SECO.

Studies related to P5 (S08, S13, S17, and S20) point out that the way the requirements are handled by a keystone demonstrates how much effort it takes to deal with trade-off issues and strives to adapt the SECO platform to its community. It should be noticed that S8 also brings some practices to give transparency to handle continuous flows of emerging requirements in a SECO.

In this situation, the reliability of the information provided by a keystone is very important because it is directly related to increasing trust in a SECO. In the field study, some participants mentioned that platforms, such as GitHub, should make information about their evolution more clear and updated to allow them to follow these changes. As mentioned by Obie et al. (2023), this information transparency is important for building trust. Some transparency actions can be used to build trust in a SECO: i) **collaborative decision-making**: involving stakeholders in decision-making processes related to requirements can enhance trust. By seeking input, considering diverse perspectives, and making decisions collaboratively, trust is built through shared ownership of outcomes; ii) **clear communication**: open and transparent communication about requirements, progress, and challenges fosters trust among stakeholders. Regular updates, status reports, and feedback mechanisms can ensure that everyone is informed and involved in the platform updates; and iii) **feedback mechanisms**: implementing feedback mechanisms where stakeholders can provide input on requirements and processes demonstrates the keystone's commitment to listening and adapting to stakeholders' needs. Actively soliciting and responding to feedback can build trust by showing that stakeholder input is valued.

Transparency in the flow of requirements and information within the SECO is crucial for building trust among stakeholders (Knauss et al., 2018; Vegendla et al., 2018; Malcher et al., 2023). By sharing detailed information, encouraging active participation, and maintaining clear documentation, keystone can establish a foundation of trust with stakeholders (Hou and Jansen, 2023). Transparency in decision-making and communication processes can further enhance trust and credibility within a SECO.

## 5.4 Future Perspectives

The analysis of the results of the SMS and the field study enabled us to get some perceptions and insights regarding future research on transparency in SECO. We express the following needs as requirements for the advancement of research on this topic.

**Information architecture models focused on transparency.** To ensure that third-party developers and other types of actors can interact satisfactorily with the documentation on the SECO platform made available by a keystone in their web portals, it is necessary to propose models for information architecture aimed at transparency. The aim is to organize information in such a way that it promotes a condition that allows individual access, ease of use, quality of content, understanding, and auditing.

**The impacts of transparency on DX at SECO.** To keep developers contributing to SECO, it is important to investigate the relationship of the transparency characteristics, presented by Leite and Cappelli (2010), with DX. Proposing transparency solutions aimed at improving DX in SECO benefited the attractiveness and engagement of new third-party developers, important actors for the expansion of the platform.

**New solutions for requirements flow transparency in SECO.** Due to the continuous flow of emerging requirements in SECO, managing demands from different actors is increasingly complex. Thus, proposing new solutions for the transparency of the requirements flow in SECO can facilitate management and open innovation within a SECO, in addition to contributing to the credibility of a keystone.

**New solutions to assessing transparency in SECO.** There is a need to investigate additional ways to assess transparency within a SECO. Although there is an emerging maturity in the adoption of actions that implement this requirement and its benefits for the dynamics of the ecosystem, a better understanding of what would be the appropriate level of transparency for a SECO and what really needs to be considered in this assessment is still required.

## 6 Threats to Validity and Credibility

This work presents two different studies (SMS and field study) and each of them has specific threats and limitations. Thus, we report the identified threats and strategies to mitigate them.

Some threats to the validity of this SMS were identified. During the course of this research, we sought to minimize the influence of these threats and reduce their possible risks. *Descriptive validity:* to reduce this threat, a data collection form has been designed to support the recording of data to answer the questions. *Theoretical validity:* the studies were analyzed and selected under the aegis of software transparency by Leite and Cappelli (2010). The search string was defined inclusively to capture studies related to concepts of transparency in SECO, but we also recognize that we could have applied the forward snowballing and included more sources to try to increase the number of selected studies.

*Generalizability:* generalization is not a huge threat once we have used a structured protocol based on Petersen et al. (2015), which facilitates replication. We also make available the datasets in the supplementary material. *Interpretive validity:* to minimize the researchers' bias, when there was doubt in executing the selection process, this was discussed between two researchers extensively and the differences were analyzed together with a third researcher until there was a

consensus. It is worth highlighting that the protocol for SMS does not consider the quality of the retrieved studies.

In contrast to quantitative studies, qualitative studies are more prone to threats to credibility. *Protocol:* as threats to rigor and reliability, semi-structured interviews can introduce biases, contain ambiguous questions, and be incomplete, even with all the care and attention from researchers. To minimize this threat, we defined the semi-structured interview protocol, being improved after carrying out the pilot. Second, we coded carefully each transcribed interview iteratively. This approach allowed us to link transcripts directly to each participant's video recordings, helping to correct any errors introduced by the automatic transcription process. The coding results were also discussed by at least two researchers and verified by two others with at least 15 years of experience in software engineering and qualitative studies until there was consensus on the categorizations.

*Sample:* we used the non-probabilistic convenience sampling technique, following the guidelines of Kitchenham et al. (2015), due to the impossibility of precisely defining the total number of participants eligible for this research. The strategies for attracting participants were specifically aimed at software developers who use GitHub to manage their projects, in addition to the snowball sampling technique, in which the first participants nominated other professionals to participate in the interview.

The number of interviews carried out was based on the concept of Guest et al. (2006) saturation. Ribeiro et al. (2022) and Steglich et al. (2023) conducted field studies with software developers considering Guest et al. (2006) and reinforce that the main important criterion is saturation, that is when any new interview with relatively homogeneous individuals does not provide any new data or information. For example, Ribeiro et al. (2022) reached saturation with 15 interviews and Steglich et al. (2023) with 20 interviews. In this study, saturation was obtained with 16 interviews, following Guest et al. (2006).

*Context:* it is important to highlight that the participants are Brazilian. The findings of this study are more relevant at a national level and can be extended to similar contexts, but do not necessarily generalize to the global software industry.

## 7 Conclusion

This article aimed to characterize conditioning factors for transparency in SECO. To do so, we conducted two studies to identify and analyze such factors. Firstly, we conducted an SMS on scientific databases and digital libraries to map and analyze the state-of-the-art of transparency and selected 23 studies for our analysis. Secondly, we conducted a field study with 16 software developers to characterize the importance of conditioning factors for transparency identified in the previous study for their performance on GitHub, a platform to support project-based ecosystems (Lungu and Lanza, 2010; Liao et al., 2019).

As the main contribution, we provided in the first study a comprehensive view of solutions, conditioning factors, processes, and concerns related to transparency in SECO, with the discussion centered on three main topics: access



to information, communication channels, and requirements engineering. We extended the contributions in the second study with a conceptual framework to better understand transparency in SECO. This framework is constituted by the conditioning factors for transparency, processes that must be transparent, benefits of transparency, consequences of lack of transparency, and solutions for transparency that emerged from the results of both studies.

In addition, this research also has implications for academia and industry. Academics can find in this work a conceptual framework to better understand transparency in SECO. We also have listed future perspectives that can contribute to the advancement of state-of-the-art. With this conceptual framework, practitioners can understand transparency as a key element in dealing with requirements that emerge from different communication channels in platform openness. We have presented solutions and conditioning factors that can help them to adopt initiatives to contribute to the open flow of information in a SECO and, thus, attract and engage new actors in a common technological platform.

The main takeaway message of this article is software transparency research has been growing, but there is still a lack of studies that focus more on transparency in SECO. Despite that, this non-functional requirement is crucial for the SECO coordination and operation in practice. So, transparency in SECO is a research field that still has a lot to be explored.

For future work, we suggest investigating solutions to assess transparency that could be applied to different SECO types: open source, proprietary, and hybrid. In addition, we have to investigate the relationship between the categories of the conceptual framework – thoroughly examine the relationship, for example, between the conditioning factors for transparency and the solutions for providing transparency identified through the SMS. Additionally, it is important to determine whether the solutions for providing transparency can be applied to a single conditioning factor or multiple factors. Moreover, it is also necessary to conduct studies in order to propose effective actions for increasing transparency of SECO information and processes.

## Acknowledgements

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001 and Grant 88887.928989/2023-00. The authors also thank CNPq (Grant 316510/2023-8), UNIRIO (DPq/PPQ 2022 & 2023) & FAPERJ (Grant 211.583/2019) for partial support.

## References

- Barbosa, O., Santos, R. P. d., Alves, C., Werner, C., and Jansen, S. (2013). *A Systematic Mapping Study on Software Ecosystems through a Three-dimensional Perspective*. In Jansen, S. et al. (eds.) *Software Ecosystems: Analyzing and Managing Business Networks in the Software Industry*, pages 59–84. Edward Elgar Publishing.
- Beelen, L., Jansen, S., and Overbeek, S. (2022). Are you of value to me? a partner selection reference method for software ecosystem orchestrators. *Science of Computer Programming*, 214:102733.
- Camelo Rincón, M. S. (2020). Análisis de la transparencia organizacional y el poder económico a partir la teoría de juegos. *Revista Universidad y Empresa*, 22:257.
- Cataldo, M. and Herbsleb, J. (2010). Architecting in software ecosystems: Interface translucence as an enabler for scalable collaboration. In *ECSAW'10: Proceedings of the IV European Conference on Software Architecture Workshops*, pages 65–72.
- Charmaz, K. (2006). *Constructing grounded theory: a practical guide through qualitative analysis*. Sage Publications, London; Thousand Oaks, Calif.
- Chen, Z., Ma, W., Chen, L., and Song, W. (2022). Collaboration in software ecosystems: A study of work groups in open environment. *Information and Software Technology*, 145:106849.
- Corbin, J. and Strauss, A. (2014). *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*. SAGE Publications.
- Cysneiros, L. M. (2013). Using i\* to elicit and model transparency in the presence of other non-functional requirements: A position paper. In Castro, J., Horkoff, J., Maiden, N. A. M., and Yu, E. S. K., editors, *Proceedings of the 6th International i\* Workshop 2013, Valencia, Spain, June 17-18, 2013*, volume 978 of *CEUR Workshop Proceedings*, pages 19–24. CEUR-WS.org.
- Dabbish, L., Stuart, C., Tsay, J., and Herbsleb, J. (2012). Social coding in github: transparency and collaboration in an open software repository. In *Proceedings of the ACM 2012 conference on computer supported cooperative work*, pages 1277–1286.
- Dyba, T., Dingsoyr, T., and Hanssen, G. K. (2007). Applying systematic reviews to diverse study types: An experience report. In *First international symposium on empirical software engineering and measurement (ESEM 2007)*, pages 225–234. IEEE.
- Fahl, S., Dechand, S., Perl, H., Fischer, F., Smrcek, J., and Smith, M. (2014). Hey, nsa: Stay away from my market! future proofing app markets against powerful attackers. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, CCS '14*, page 1143–1155, New York, NY, USA. Association for Computing Machinery.
- Fontão, A., Cleger-Tamayo, S., Wiese, I., Pereira dos Santos, R., and Claudio Dias-Neto, A. (2021). A developer relations (devrel) model to govern developers in software ecosystems. *Journal of Software: Evolution and Process*, page e2389.
- Goyal, R., Ferreira, G., Kästner, C., and Herbsleb, J. (2018). Identifying unusual commits on github. *Journal of Software: Evolution and Process*, 30(1):e1893.
- Green, H. E. (2014). Use of theoretical and conceptual frameworks in qualitative research. *Nurse researcher*, 21(6).
- Greiler, M., Storey, M.-A., and Noda, A. (2023). An actionable framework for understanding and improving developer experience. *IEEE Transactions on Software Engineering*, 49(4):1411–1425.

- Guest, G., Bunce, A., and Johnson, L. (2006). How many interviews are enough? *Field Methods - FIELD METHOD*, 18:59–82.
- Haenni, N., Lungu, M., Schwarz, N., and Nierstrasz, O. (2014). A quantitative analysis of developer information needs in software ecosystems. In *Proceedings of the 2014 European Conference on Software Architecture Workshops*, pages 1–6.
- Hanssen, G. and Dybå, T. (2012). Theoretical foundations of software ecosystems. In *Proceedings of the 4th International Workshop on Software Ecosystems (IWSECO) - 3rd International Conference on Software Business (ICSOB)*, volume 879, pages 6–17.
- Hanssen, G. K. (2012). A longitudinal case study of an emerging software ecosystem: Implications for practice and theory. *Journal of Systems and Software*, 85(7):1455–1466.
- Herbsleb, J. (2016). Building a socio-technical theory of coordination: why and how (outstanding research award). In *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pages 2–10.
- Herbsleb, J., Kastner, C., and Bogart, C. (2016). Intelligently transparent software ecosystems. *IEEE Software*, 33(01):89–96.
- Hosseini, M., Shahri, A., Phalp, K., and Ali, R. (2016). A modelling language for transparency requirements in business information systems. In Nurcan, S., Soffer, P., Bajec, M., and Eder, J., editors, *Advanced Information Systems Engineering*, pages 239–254, Cham. Springer International Publishing.
- Hou, F. and Jansen, S. (2023). A systematic literature review on trust in the software ecosystem. *Empirical Software Engineering*, (8).
- Issac, G., Rajendran, C., and Anantharaman, R. (2004). A conceptual framework for total quality management in software organizations. *Total Quality Management & Business Excellence*, 15(3):307–344.
- Jansen, S. (2020). A focus area maturity model for software ecosystem governance. *Information and Software Technology*, 118:106219.
- Jansen, S., Brinkkemper, S., Finkelstein, A., and Bosch, J. (2009). Introduction to the proceedings of the first workshop on software ecosystems. In *Proceedings of the First Workshop on Software Ecosystems*, CEUR-WS, page 1–2.
- Jansen, S., Brinkkemper, S., Souer, J., and Luinburg, L. (2012). Shades of gray: Opening up a software producing organization with the open software enterprise model. *Journal of Systems and Software*, 85(7):1495–1510.
- Jansen, S., Cusumano, M. A., and Brinkkemper, S. (2013). *Software ecosystems: analyzing and managing business networks in the software industry*. Edward Elgar Publishing.
- Kilamo, T., Hammouda, I., Mikkonen, T., and Aaltonen, T. (2012). From proprietary to open source—growing an open source ecosystem. *Journal of Systems and Software*, 85(7):1467–1478. Software Ecosystems.
- Kitchenham, B. and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. *EBSE Technical Report EBSE-2007-01*.
- Kitchenham, B. A., Budgen, D., and Brereton, P. (2015). *Evidence-Based Software Engineering and Systematic Reviews*. Chapman & Hall/CRC.
- Knauss, E., Damian, D., Knauss, A., and Borici, A. (2014). Openness and requirements: Opportunities and tradeoffs in software ecosystems. In *2014 IEEE 22nd International Requirements Engineering Conference (RE)*, pages 213–222. IEEE.
- Knauss, E., Yussuf, A., Blincoe, K., Damian, D., and Knauss, A. (2018). Continuous clarification and emergent requirements flows in open-commercial software ecosystems. *Requirements Engineering*, 23:97–117.
- Kon, F., Hazzan, O., Yuklea, H., Cukier, D., and Melo, C. d. O. (2015). A conceptual framework for software startup ecosystems: the case of israel.
- Leite, J. C. S. P. and Cappelli, C. (2010). Software transparency. *Business & Information Systems Engineering*, 2:127–139.
- Liao, Z., Wang, N., Liu, S., Zhang, Y., Liu, H., and Zhang, Q. (2019). Identification-method research for open-source software ecosystems. *Symmetry*, 11(2).
- Linåker, J. and Wnuk, K. (2016). Requirements analysis and management for benefiting openness. In *2016 IEEE 24th International Requirements Engineering Conference Workshops (REW)*, pages 344–349. IEEE.
- Lungu, M. and Lanza, M. (2010). The small project observatory: A tool for reverse engineering software ecosystems. ICSE '10, page 289–292, New York, NY, USA. Association for Computing Machinery.
- Malcher, P., Silva, E., Viana, D., and Santos, R. (2023). What do we know about requirements management in software ecosystems? *Requirements Engineering*, 28(4):567–593.
- Manikas, K. (2016). Revisiting software ecosystems research. *Journal of Systems and Software*, 117:84–103.
- Manikas, K. and Hansen, K. M. (2013). Software ecosystems—a systematic literature review. *Journal of Systems and Software*, 86(5):1294–1306.
- Meireles, A. I., dos Santos, R. P., and Cappelli, C. (2017). Construindo um questionário para avaliar transparência em portais de ecossistemas de software (building a questionnaire to evaluate transparency in software ecosystem portals). In *Anais do VIII Workshop sobre Aspectos da Interação Humano-Computador para a Web Social*, pages 25–35. SBC.
- Meireles, A. I., dos Santos, R. P., and Cappelli, C. (2019). An instrument for the evaluation of transparency mechanisms in software ecosystem portalsalexandre. *iSys-Brazilian Journal of Information Systems*, 12(2):05–38.
- Molder, J. t., Lier, B. v., and Jansen, S. (2011). Clopenness of systems: The interwoven nature of ecosystems. In *IWSECO@ICSOB*, pages 52–64. Citeseer.
- Monteith, J. Y., McGregor, J. D., and Ingram, J. E. (2014). Proposed metrics on ecosystem health. In *Proceedings of the 2014 ACM international workshop on Software-defined ecosystems*, pages 33–36.
- Motta, R. C., De Oliveira, K. M., and Travassos, G. H. (2018). On challenges in engineering iot software systems. In *Proceedings of the XXXII Brazilian symposium on software*

- engineering, pages 42–51, New York, NY, USA. ACM.
- Obie, H. O., Ukwella, J., Madampe, K., Grundy, J., and Shahin, M. (2023). Towards an understanding of developers' perceptions of transparency in software development: A preliminary study. In *2023 38th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW)*, pages 40–45.
- Oliveira, J. and Alves, C. (2021). Software ecosystems governance – an analysis of sap and gnome platforms. In *2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 296–299.
- Parahoo, K. (2014). *Nursing research: principles, process and issues*. Bloomsbury Publishing.
- Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1–18.
- Ralph, P. (2021). Acm sigsoft empirical standards released. *SIGSOFT Softw. Eng. Notes*, 46(1):19.
- Ribeiro, B. B., Costa, C., and Pereira dos Santos, R. (2022). Understanding and analyzing factors that affect merge conflicts from the perspective of software developers. *Journal of Software Engineering Research and Development*, 10:12:1–12:17.
- Runeson, P., Olsson, T., and Linåker, J. (2021). Open data ecosystems — an empirical investigation into an emerging industry collaboration concept. *Journal of Systems and Software*, 182:111088.
- Santos, R., Cappelli, C., Maciel, C., and Leite, J. C. S. d. P. (2016). Transparência em ecossistemas de software. In *WDES'16: Anais do X Workshop em Desenvolvimento Distribuído de Software, Ecossistemas de Software e Sistemas-de-Sistemas*, pages 75–79, Porto Alegre, RS, Brasil. SBC - Sociedade Brasileira de Computação.
- Santos, R. P. d. (2016). *Managing and Monitoring Software Ecosystem to Support Demand and Solution Analysis*. Tese de doutorado, COPPE/UFRJ, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brasil.
- Schultis, K.-B., Elsner, C., and Lohmann, D. (2014). Architecture challenges for internal software ecosystems: A large-scale industry case study. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pages 542–552.
- Setzke, D. S., Böhm, M., and Krömer, H. (2019). Platform openness: A systematic literature review and avenues for future research.
- Shaw, M. (2003). Writing good software engineering research papers. In *25th International Conference on Software Engineering, 2003. Proceedings.*, pages 726–736. IEEE.
- Singer, J., Sim, S. E., and Lethbridge, T. C. (2008). Software engineering data collection for field studies. In *Guide to Advanced Empirical Software Engineering*, pages 9–34. Springer.
- Souza, K. E. S. d., Zacarias, R. O., da Rocha Seruffo, M. C., and Santos, R. P. d. (2020). T2-uxt: A tool to support transparency evaluation in software ecosystems portals. In *Proceedings of the 34th Brazilian Symposium on Software Engineering*, pages 415–420, New York, NY, USA. ACM.
- Steglich, C., Marczak, S., dos Santos, R. P., Guerra, L., Mosmann, L., Moreira, M., and Perin, M. (2023). Factors that affect developers' decision to participate in a mobile software ecosystem. *Journal of Systems and Software*, 205:111808.
- Valença, G., Kneuper, R., and Rebelo, M. E. (2020). Privacy in software ecosystems - an initial analysis of data protection roles and challenges. In *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 120–123.
- Vegendla, A., Duc, A. N., Gao, S., and Sindre, G. (2018). A systematic mapping study on requirements engineering in software ecosystems. *Journal of Information Technology Research (JITR)*, 11(1):49–69.
- Villamizar, H., Escovedo, T., and Kalinowski, M. (2021). Requirements engineering for machine learning: A systematic mapping study. In *2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 29–36.
- Zacarias, R. O., Gonçalves, R. F., and dos Santos, R. P. (2023). Investigating transparency in software ecosystems. In *Proceedings of the XXXVII Brazilian Symposium on Software Engineering*, pages 132–141, New York, NY, USA. ACM.