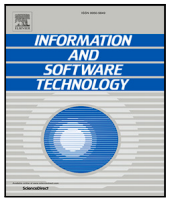
Information and Software Technology 158 (2023) 107170

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof



An investigation of causes and effects of trust in Boundary Artefacts

Raquel Ouriques [a,*], Fabian Fagerholm [a,b], Daniel Mendez [a,c], Baldvin Gislason Bern [d]

[a] Blekinge Institute of Technology, Karlskrona, 37179, Sweden
[b] Aalto University, Espoo, FI-00076, Finland
[c] fortiss, Munich, 80805, Germany
[d] Axis Communications, Lund, 22369, Sweden

ARTICLE INFO

Keywords:
Software development
Boundary Artefact
Trust
Trusting beliefs

ABSTRACT

Context: Boundary Artefacts (BAs) support software development activities in many aspects because it carries lots of information in the same object that can be used and interpreted by several social groups within an organisation. When the BAs are inconsistent regarding their content, such as many meanings or lack of contextual information, their efficiency is reduced because stakeholders will not trust them.
Objective: This study aimed to understand the implications of differences in the perception of trust on software projects and their influence on stakeholders' behaviour.
Methods: We conducted an exploratory case study to observe the creation and utilisation of one specific BA and the implications of differences in trust and their influence on stakeholders' behaviour.
Results : Our investigation has shown that practitioners adding and adjusting existing content do not entirely understand the stakeholders' needs. Together with the partial management of the content, trust is impacted. When the content of BAs does not meet the trust factors, specifically reliability and predictability, the stakeholders cannot execute their tasks appropriately, and several implications affect the software development project. Additionally, they create workarounds to supply their needs.
Conclusion: The differences in trust in BAs affect software projects in different areas of the organisation and interfere with the task execution of various stakeholders. The decrease in trust results from inconsistencies in the content associated with the lack of management of the BA. A structured strategy for representing and managing a BA's content seems appropriate to increase trust levels and efficiency.

1. Introduction

Boundary Artefacts (BAs) are central objects that tie together teams in multiple social worlds along the software development lifecycle while holding different meanings [1,2]. They are used to steer collaborative work between various stakeholders. For instance, a use case describing how users intend to interact with the system is defined from a requirements perspective. It serves as a basis for various other activities such as project organisation and management (e.g. effort estimation), design, and (acceptance) testing.

The BAs are essential for effective collaboration because they contain relevant information that supplies different groups with different needs. They materialise mainly as electronic or printed documents and, for the most part, are produced and used by humans. Being central resources and extensively used, disregarding the methodology adopted to guide software development processes, they provide value by condensing practitioners' knowledge in different formats, such as architecture descriptions, requirements specifications, test cases, etc. [3,4].

Software development teams, especially when geographically dispersed, rely on BAs for efficient coordination and knowledge sharing. The successful handling of such artefacts depends on how people manipulate and interpret information, and how they use technology to spread knowledge. Such complexity increases when a BA requires contributions from different social groups with distinct perspectives and knowledge background [3,5].

While providing benefits such as shared understanding and knowledge resources to distinct groups, BAs can also fail in their purpose. Misunderstandings can happen when the artefact is not plastic enough to accommodate the different meanings. In other cases, it requires more contextual information so stakeholders can process the available information, or constant updates are necessary to satisfy needs [6–8].

As such inconsistencies occur, stakeholders' usage can be undermined, and the efficiency of these artefacts is reduced. For example, when a BA is mismanaged, learning across diverse groups becomes limited and integrating additional knowledge is difficult [9]. Moreover, when a BA do not comply with sufficient level of detail and

* Corresponding author.
E-mail address: raquel.ouriques@bth.se (R. Ouriques).

https://doi.org/10.1016/j.infsof.2023.107170
Received 17 October 2022; Received in revised form 2 February 2023; Accepted 6 February 2023
Available online 9 February 2023

practitioners neglect the management of such artefacts, stakeholders will not be able to find relevant and updated content. In worst cases, the workload can increase due to redesigning solutions that could be avoided if artefacts were properly managed [4].

In this circumstance, when practitioners do not feel confident in using the BA, trust decreases, and unknown consequences to the software projects can come into sight. This situation can also create an unpleasant environment with a weak sense of community that can be developed and encourage demand for accountability [5,10,11].

As trustworthy BAs are crucial for executing software development activities successfully, it is important to understand how practitioners create and use them, and what the implications are when there is a lack of trust in the BAs. *Trust* is an essential factor that determines the adoption of artefacts and the extent to which a practitioner feels confident using it [10,12].

Software engineering researchers have examined BAs in several past studies. Most studies explore how companies benefit from using them for information sharing and collaboration among teams [13–16]. On the other hand, BAs must be trustworthy so that practitioners can rely on them to execute their tasks. However, to the best of our knowledge, this aspect has not been examined so far.

In this investigation, we address this gap by reporting on a study examining the creation and utilisation of one specific BA and analysing how trust affects stakeholder behaviour towards it.

In particular, we aim at the following contributions:

- An empirical investigation of how a BA is created, utilised, and managed in a software development environment.
- An analysis of the implications of a decreased level of trust and how they influence the stakeholders' behaviour towards the BA.
- A discussion of the possible implications for future research and potential solutions for managing the analysed BA.

This manuscript is organised as follows: In Section 2, we present a brief background to our study and discuss related work. Section 3 describes the research method. In Section 4, we present the findings of our study. In Section 5, we discuss the implications that our findings have and future research. In Section 6, we then discuss the threats to the validity of our study before concluding our manuscript in Section 7.

2. Background and related work

In this section, we first introduce the basic notion of boundary artefacts (BAs) as background of our study before discussing related work.

2.1. Boundary artefacts

The concept of boundary artefacts is not new and originates in the discipline of ecology [8,17,18]. Two main terms are commonly used in the literature: Boundary objects and Boundary artefacts. As we focus on the software engineering discipline, we use the term *boundary artefact* (BA) due to its proximity to the field. While we refer to Mendez et al. [19] for a more elaborate discussion and definition of the term, an artefact is, in simple terms, a work product that is produced, modified, or used by a sequence of tasks that have value to a role, such as code, test cases, or a requirements documentation.

The original introduction of the term boundary artefact roots back to 1989 and was coined in a study developed for the Museum of Vertebrate Zoology [17]. The authors define it as "objects which are both plastic enough to adapt to local needs and the constraints of the several parties employing them, yet robust enough to maintain a common identity across sites".

The boundary does not refer to a limit, edge, or periphery but a shared space structure. These objects carry an interpretative flexibility characteristic, which means that different groups gather information and interpret it differently from the same object [20].

Star and Griesemer [17] detail four types of BAs:

- Repositories, which are traditional and standardised piles of objects such as a library.
- Ideal type, which denotes objects with a high level of abstraction that are adaptable to all domains. However, due to its vague characteristic, it suits better for communicating.
- Coincident boundaries, i.e. objects which, while possessing different contents, have the same boundaries. They fit organisations with geographically distributed stakeholders that need the information to work autonomously while cooperating with a commonly shared referent.
- Standardised forms, which are objects with a rigid structure for adding information.

BAs are, thus, useful objects that can support software development in many aspects, including being the source of information to many stakeholders within an organisation. However, the fast-paced changes to the software products can affect the flow of information in these objects [21].

As boundary artefacts grow, they can lose flexibility because too many boundaries are crossed. When there is an overload of meanings, a BA can lose value to its stakeholders, resulting in workarounds to avoid the BA at all [7,17,22].

Practitioners need to manage the different borders and information needs of other stakeholders to keep the BA useful with shared understanding [3,11]. As we recognise the importance of BA for software engineering, we want to understand why people avoid using them. The circumstances around the creation and management of the BA can affect how people trust them, and as a consequence, the BA can be more or less valuable or attractive, which may explain why they are used or not.

2.2. Work related to trust in (boundary) artefacts

As the notion of trust has been explored in several disciplines, there is still no consensus about its exact definition. The term acquires different meanings depending on the context investigated [11,12].

A large number of definitions of trust connect to human behaviour towards another. In this context, trust refers to how individuals depend voluntarily on other individuals' behaviour. Due to its vague trait, it is complex to evaluate. To make it possible to describe, several authors proposed *trusting beliefs*, which are favourable factors that help to describe what causes a person to consider another person trustworthy [10,23].

Several authors discuss and provide different attributes for trusting beliefs. They depend on the context. The most cited ones are from McKnight et al. [23]: benevolence, competence, honesty/integrity, and predictability. However, these beliefs focus on aspects of human social interaction and denote assessments that people make about each other.

Based on previous research, trust in inanimate software artefacts has been investigated under the premise that people perceive these artefacts as social actors, as if they possess human attributes [24,25].

In an extensive study in information technology artefacts, Lansing and Sunyaev [12] scrutinised the trust literature to identify trusting beliefs applied to inanimate software artefacts. From this perspective, an information technology artefact should be:

- Reliable: Perception that the artefact provides accurate content.
- Predictable: Confidence that the resource is always provisioned as requested.
- Provide the required functionality: perform as needed for the task environment.

In our study at hands, we reuse those three trusting beliefs.

A few studies in the software engineering field focus further on trust in artefacts. Reviewing empirical studies, Lansing and Sunyaev developed a conceptual model to describe trust in the cloud service software

artefact. They comprehensively analyse the several trust constructs to reach a model contributing to theory-building on trust.

Koelmann [26] also provides a theoretical discussion around the concept of trust. However, he focuses on an information technology artefact's role in trust-related interactions. He further details the implications that can arise in situations involving trust.

Regarding empirical studies, we also found an original work focusing on trust in inanimate software artefacts. Thatcher et al. [27] examined how trust is affected by information technology and its support in users post-adoption. We consider this work as relevant to our study as it initiated the development of new concepts regarding trust constructs concerning inanimate software artefacts with evidence from an empirical investigation.

Paravastu et al. [25] explored how trust and trustworthiness apply to users' satisfaction. Their empirical investigation has shown differences between the two explored constructs, and user satisfaction is related to trustworthiness rather than trust in software artefacts. Trust in inanimate software artefacts is little examined in the software engineering field, especially regarding artefacts produced during the development process, which are utilised as resources from one phase to another.

As software development becomes geographically dispersed, people tend to rely more on artefacts to execute their tasks. To do so, they need to trust the information provided by the artefacts. In the interest of advancing research into trust in inanimate software artefacts, our study adds to our understanding of how trust can affect stakeholders' behaviour in software projects.

3. Research method

Our study aims to examine the process of creating and utilising a BA. We are especially interested in the implications of trust beliefs and how they influence the stakeholders' behaviour in this process.

We developed our Research Questions (RQ) through discussions and brainstorming sessions with our company partner (introduced in Section 4.1). We addressed four RQs:

RQ1 How do practitioners decide how to represent knowledge and information in boundary artefacts and their expectations concerning their use?
RQ2 How do practitioners manage boundary artefacts?
RQ3 What are the implications of the differences in trust in boundary artefacts to software development?
RQ4 How does a difference in the level of trust in boundary artefacts influence the stakeholders' usage behaviour?

To answer these questions, we conducted a qualitative study analysing data on phenomena related to human behaviour [28]. We followed the principles of an exploratory case study to gather insights into creating and utilising the selected BA [29].

We collected the data via a series of sources, including interviews, archive documents, chat via Microsoft Teams™ and a workshop. We examined the interviews by two-level coding (see Section 3.3) to identify relevant codes that enabled us to answer our RQs and explain the trust cycle (see Fig. 6) [30].

3.1. Research approach

Fig. 1 depicts the approach that we developed to investigate the actions and interactions surrounding the BA. Our approach focused on three perspectives:

a. Contributors We investigated how the practitioners added content to the BA and their perception of who are the stakeholders of the object and how they utilise it.

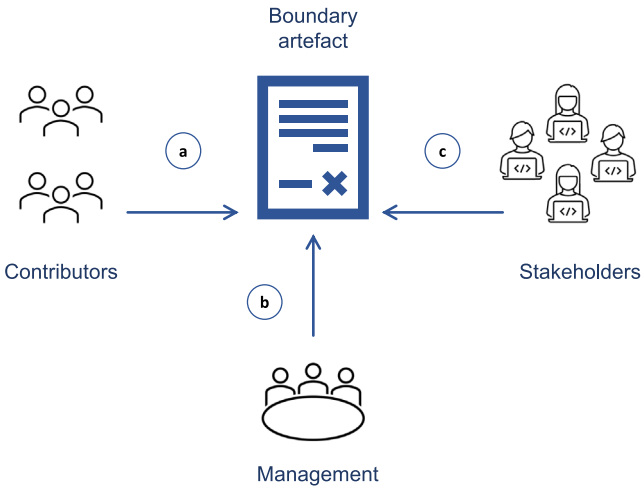


Fig. 1. Research approach.

Table 1
Description of the practitioners included in this study (Interviews).

Participants	Role	Type
P1	Senior engineer	Contributor
P2	Senior engineer	Stakeholder
P2	Senior engineer	Stakeholder
P3	Specialist engineer	Stakeholder
P4	IT product owner	Management
P5	Experienced engineer	Stakeholder
P6	Senior engineer	Stakeholder
P7	Senior engineer	Contributor
P8	Senior engineer	Stakeholder

b. Content management In this perspective, we identified content management aspects associated with the BA. We paid attention to existing guidelines for adding content, responsibilities, and ownership.

c. Stakeholders We explored how the stakeholders utilised the BA. We examined their perception of trusting beliefs in relation to the BA and how they influenced the stakeholders' behaviour.

This approach allowed us to identify misalignment between the content creation and management from the contributors' side and evaluate the trust of the stakeholders concerning the BA.

3.2. Data collection

The data for this study was collected by means of interviews, archive documents, workshop, and informal conversations with practitioners. The first and fourth authors had full access to the company network and Information and Communication Technologies (ICTs) through dedicated employee accounts and devices set up and provided as part of the collaboration.

We utilised semi-structured interviews and one unstructured. The interview questions (see Fig. 2) were selected according to the role of each practitioner who was interviewed, be it contributors or stakeholders (see Table 1) within the Quality Assurance unit, except for the content management approach. Thus, the inter-stakeholders differences are not of relevance to our analysis. In this approach, we used an unstructured interview due to the exploratory characteristic of the event.

We chose the trusting beliefs conceptualisation proposed by Lansing and Sunyaev [12] as our guide for elaborating the interview guide and analysing the results. It represents a combination of several theories of trust applied to inanimate software artefacts.

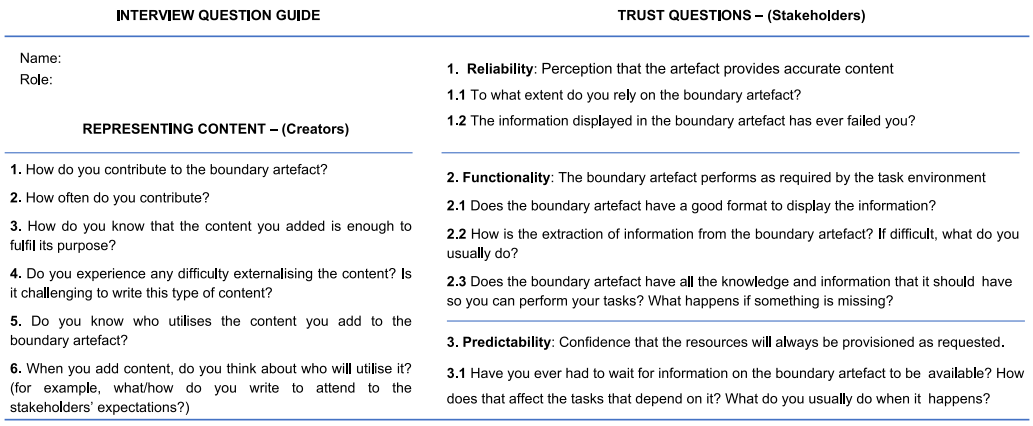


Fig. 2. Interview guide.

Table 2
Description of the practitioners included in this study — Workshop.

Participants	Role	Type
P1	Senior engineer	Contributor
P2	Senior engineer	Stakeholder
P3	Specialist engineer	Stakeholder
P4	IT product owner	Management
P5	Experienced engineer	Stakeholder
P6	Senior engineer	Stakeholder
P7	Senior engineer	Contributor
P8	Senior engineer	Stakeholder
P9	Engineer	Engineer invited for discussion
P10	Expert engineer	Experienced engineer invited for discussion

The interviews were conducted through Microsoft Teams™ and lasted around 30 min. We also utilised Microsoft Teams™ for gathering data during the entire collaboration in informal chats with several practitioners. We initiated the data collection in September 2021 with the interviews and finished in April 2022 with the workshop and informal discussions with practitioners.

We gathered data from archival files, including Confluence™ pages, internal presentations, and descriptions of tools. We also collected data from a workshop planned for confirmation of the results and additional data.

The workshop lasted 1,5 h and happened in the last days of the data collection. We had eight participants interviewed in the previous step and two practitioners who had a close interest in the subject (see Table 2). The purpose was to validate the draft of the results and collect complementary data. We focused on the three aspects of our proposed approach (see Fig. 1).

We stored three data types (audio, video and text) in the MAXQDA[1] software for qualitative and mixed methods research. The software supported us in extracting the codes in the two-step coding cycle directly from the data source.

3.3. Data analysis

The data collection process and the analysis of the data were interrelated. The first author transcribed the interviews and established the initial codes. Simultaneously, the first author contacted people already interviewed and other practitioners in the company to clarify internal processes and misunderstandings.

We performed a two-cycle coding [30]. In the first cycle, we assigned process codes to the data chunks. This coding method indicates observable and conceptual action in the data and summarises data segments. The research questions partially steered the codes. For example, when the chunk of data referred to RQ1 and RQ3, we often coded it as "contributing to the boundary object" and "trust implication" (see Fig. 3).

In the second cycle, we broke down the codes generated in the first cycle into smaller concepts that represented patterns in the data (see Fig. 3). After concluding the second cycle, the first and the second authors gathered to analyse the generated codes. In this session, we discussed all codes and how and why they were added, aiming for consistency. We also examined the codes against the transcripts. We corrected misunderstandings, consolidated the initial findings through agreement, and identified gaps that required further clarification.

To increase reliability, we conducted a workshop with practitioners participating in the study. We asked for confirmation of our findings in relation to the three aspects of our approach, including content creation and management of the BA, the implication of trust beliefs, and practitioners' behaviour changes. After the workshop, we adjusted our interpretations regarding the content management findings, which changed to *partial* content management. Although we validated most of the results, we still had open questions that required further investigation, including the stakeholders' behaviour, which was later clarified in discussions with practitioners through Microsoft Teams™.

3.4. Ethical concerns

We followed the Swedish Research Council [31] guidelines for conducting ethical research. Besides the guidelines, we also paid attention to how the interview guide was elaborated. We were careful in designing a guide that would not raise intense emotions or cause psychological harm [32].

The participants knew the study's goal and that they could choose to remove their interview from the research before the research was published. We asked for permission to record the interviews and explained that the data would be stored on the company's computer only. Moreover, every mention of specific pieces of data, such as quotes, would be anonymised.

When sampling the practitioners to participate, we decided based on the following criteria: their role (creating content, managing or utilising the BA) and availability during the period we performed the interviews.

We had an intermediate step in our research where we conducted a workshop to present a draft of the findings, aiming for validation and additional data collection. During the workshop, we were cautious not to reveal participants' names or the teams they belonged to, avoiding any discomfort.

Regarding the data analysis, we avoided potential stigmatisation or harm to specific populations by not collecting or analysing data related to the religion, race, or ethnicity of any participant in this study.

[1] Available at https://www.maxqda.com.

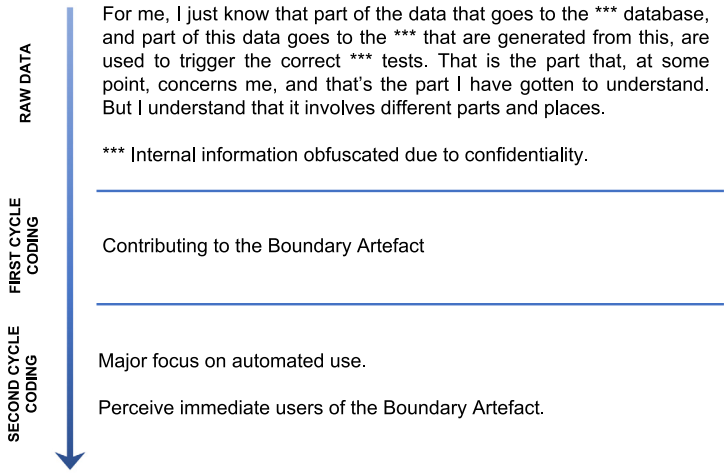


Fig. 3. Example of the two-cycle coding process.

4. Results

In this section, we report on the results of our study, which investigates the creation and utilisation of one specific BA and the implications of trust and their influence on stakeholders' behaviour. We divide the subsections in tune with our RQs (see Section 3).

4.1. Case company

Axis Communications provides network solutions in video surveillance, access control, intercom, and audio systems. They collaborate with thousands of technology and system integration partners around the world to deliver solutions and support customers with focus on openness and quality. Axis has around 4000 employees in over 50 countries and has its headquarters in Lund, Sweden, which is where the case study was centred.

The AXIS OS operating system for edge devices is used in more than 200 of Axis' products. Over 1000 research and development engineers work directly or indirectly with developing AXIS OS as a software product line. There are also other Axis employees and Axis partners that are dependent on information about individual products in the AXIS OS product portfolio, such as technical writers, support personnel and development partners.

4.2. Description of the boundary artefact

The BA analysed in this study represents the core list of features of all of the products in a software product line of the company (covering over 200 products), and it is physically represented as an XML file. It stores all the developed features for the firmware used in the company's products, be it new, modified or deprecated features. Teams with specific functions (approximately 48 teams), e.g., audio and streaming, add the features and associated content to the BA.

The artefact is handled as code and managed in the same source code management tool used for the product code for the software product line. Automated regression tests for the software product line use the BA so that discrepancies are detected in the build pipeline. Those that need changes to the BA will create pull requests, represented by the dotted lines in Fig. 4.

Fig. 4 also displays a summary of the several internal stakeholders in different parts of the organisation. Customer and external partners (for example, application development partners) are also stakeholders. Our investigation focused on the Quality Assurance department.

The content of this BA is utilised in automated and manual formats, but mostly automated. The information is automatically retrieved through a framework that collects product information using an API (Application Programming Interface).

Through the automatic retrieving, information from the BA crosses several boundaries within the organisation, including marketing, global services, quality assurance, platform management, solution management, customer information, etc. At each boundary, the information provided by the BA serve different purpose.

Besides the mentioned department boundaries within the organisation, the BA represents a shared structure for teams with similar activities but different goals and focuses. Those characteristics resemble the *Coincident boundaries* artefact type described by Star and Griesemer [17] (see Section 2). It has the advantage of providing input to perform tasks with different goals. Teams work autonomously but collaborate in this shared artefact.

4.3. Representing and managing the content of the BA

In this subsection, we detail the process of adding content to the BA and also the approach adopted to manage it, addressing both *RQ1* and *RQ2*.

So-called function teams add new features and make adjustments to existing ones. This process is meant to be flexible. Creators in such teams begin with rudimentary versions of the content and improve them until they have a satisfactory version, such as descriptions and parameters.

Regarding the new features, the content addition is guided by a multidisciplinary team — having the goal of building a generic vocabulary that the whole company can use. They book sessions so the creators can explain the new feature. The team then choose names for the features and make sure that the names are technically correct and easy to translate. The multidisciplinary team partially manages the artefact with the activities detailed in Table 3.

The management team also clear up the features that do not belong to the artefact and deprecate features that are no longer in use but keep there as historical data.

In our investigation, we found that those function teams perceive the immediate users of the artefact. Their focus is on automated use, while the manual extraction of the content is not widely known, exemplified by P1: "*For me, I just know that part of this data goes to the database, and part of this data goes to the feature lists that are generated from this, are used to trigger the correct tests*".

As this happens, the stakeholders of the manual extraction do not have their needs observed. We detail these findings in the following subsection and the discussion section.

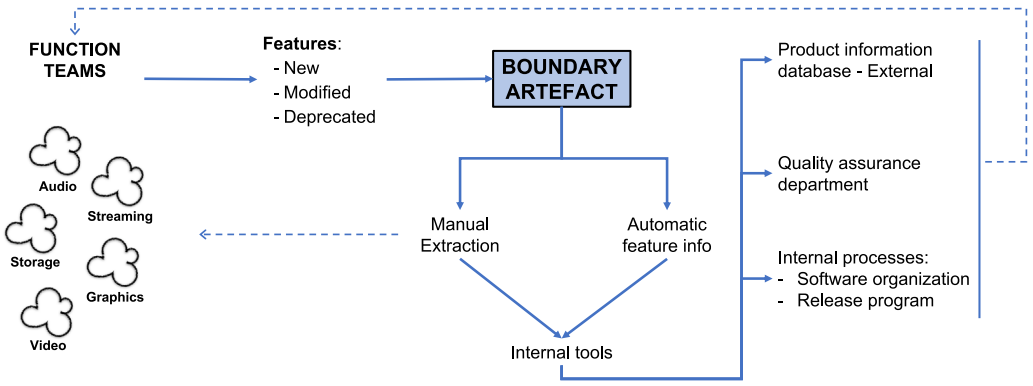


Fig. 4. Depiction of the utilisation of the boundary artefact.

Table 3
Activities for managing the BA.

Activity	Description
Obtaining an overview of the utilisation	Awareness of how the content is used but predominantly in the automated side and generation of information to customers.
Assigning responsibility for parts of the content	Finding owners for each feature.
Achieving flexibility in adding content	Possibility for incremental edits to the existing content by the teams.
Capturing knowledge about each feature	The multidisciplinary team establish meetings with creators for details about new features.
Establishing a common terminology	Defining terms and names to be known by the whole company and customers.
Planning for future increments	Predicting that a feature has the potential to grow and list the future options to reduce the updates.
Controlling over modifications	Every modification generates an approval request which at least two people review.
Creating generic content	Making features for different setups and not connected to a specific device.

4.4. Implication of trusting beliefs

In this subsection, we detail our investigation of the differences in trust and the implications generated when the BA is not reliable or predictable — answering the *RQ3*.

As the stakeholders rely on BA to execute their tasks, they expect the information stored in the artefact to fulfil their needs. Several implications can affect the development process when practitioners understand that information does not meet favourable factors of trusting beliefs.

The information provided by the BA meets the **functionality favourable factor**. One explanation is that most of the content is in a code format, read by developers and other tools that automatically extract the information. In this aspect, P2 commented: "*Well, the XML is never human-readable. So, no, it doesn't, but programmatically it works, of course. There is no problem in reliability for the code in the BA, but it is not human readable*". To a stakeholder not involved in this level (development), a web version of the artefact displays the description for each feature. For those stakeholders, the web version meets the functionality factor.

As to **reliability**, we found that the content stored in the BA failed the stakeholders. When the stored information is incorrect, the automatic extraction can run tests that will fail, execute wrong tests, or not even run them. P8 exemplifies saying: "*It sometimes affects when we execute new tests. The BA must be updated before the tests run on devices with the correct features*". P7 adds, "*There have been updates to the BA that broke tools that I used because it didn't follow the XML standard or someone added something that simply didn't work. So, there failed from time to time*".

When those issues occur, it is evident that the development process can suffer a setback due to waiting time to fix them. The incorrectness of the information can also generate mistakes in appointing tasks to the wrong practitioners. As we mentioned in the previous section, when the content creators of the BA are not familiar with the stakeholders' needs, the content generated will not support the execution of their tasks. In this regard, P6 stated: "*I think it is important that the features have the right ownership to address tasks concerning that features to the right tester. The information can be fetched through other sources, but it takes longer, so the BA is a quick way to get it*".

Terminology occasionally causes misunderstanding because, depending on the department of the practitioners, they spend some time understanding what the feature is about. Regarding this issue, P2 commented: "*I need to land in the BA first to understand which feature I am looking for because the namings in the BA are not the same as the namings that I'm used to. Because it is not the same name I use in software terminology, sometimes it is confusing to figure out how a feature is activated on the camera*".

Regarding **predictability**, we examined if the practitioners had to, at any time, wait for information to be available. When the information of the BA does not meet this criterion, the automated tests get delayed, which connects to the implications found in the reliability factor. In this regard, P4 explains: "*If it is missing, you sure have to wait. I am not the one who put it in. So, I have to tell someone who has to go through this whole process, so there is a huge delay. I expect it to take a long time if it is not in there*".

At last, the lack of predictability generates the feeling of uncertainty about the availability of the information by stakeholders. As P3 observed, "*It sometimes affects when we can't take in new tests. The BA needs to be updated before, so the tests are run on devices with the correct features*".

The combination of the reported implications influences the stakeholders' behaviour as they need to execute their tasks. We describe this behaviour in the following subsection.

4.5. Change in behaviour

In this subsection, we explore how a difference in the level of trust in the BA influences the stakeholders' behaviour *(RQ4)*.

There are features with multiple implementations in the BA due to differences among the products. In some situations, the stakeholders find it difficult to know if they can use one of those implementations. They cannot use it in other cases because the BA does not offer the necessary implementations.

To execute their tasks, the stakeholders create workarounds such as complementary artefacts. They are made within the function teams and address their specific need. This behaviour is derived from our reliability investigation. When the information fails the stakeholder, they move towards another solution that will not be ideal, which is

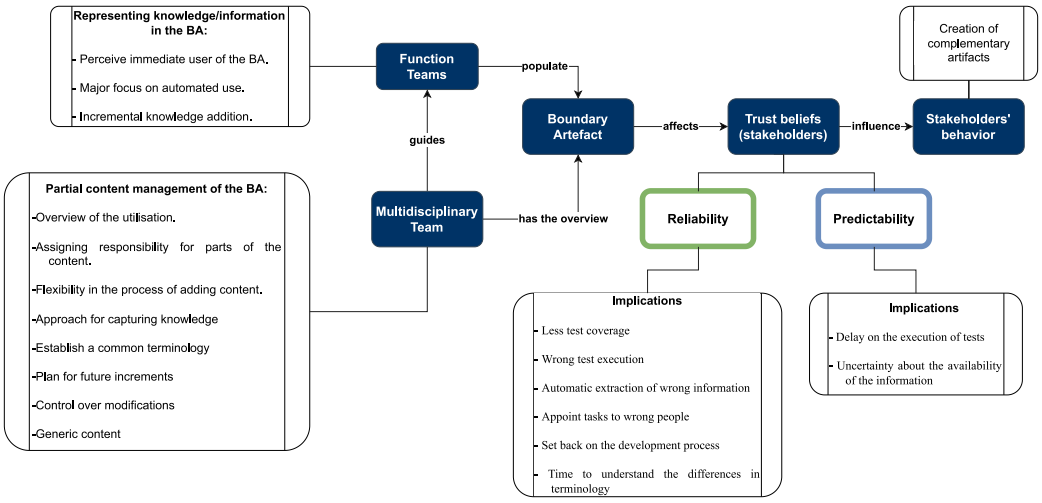


Fig. 5. Summary of the findings.

this case. Note the remark made by P4: *"It is often that we can't use the BA in those cases, so we can't make a test out of it. We have our own test suite with our own made-up BA for many products".*

The content is expected to be merged with the official BA. However, there is no control over when those cases happen, how many teams create them, or if the content has been actually merged. P4 details this process: *"This artefact is official in the sense that we use it for our tests. No other usage than for test purposes. Just my team, as far as I know. It grows with the need to separate tests for products. We add that if we have a test that needs to be run depending on a feature. If the official BA had these implementations, we would not need it".* During the workshop, we also confirmed that this happens to other teams, which can be disastrous, according to one of the experienced engineers. Besides, during our investigation, one feature was missing in the BA.

To better visualise our research findings, we summarised them in Fig. 5. It shows the process of creation, management, and the implications of the differences in trust in a software project. Also, how they influence stakeholders' behaviour.

Together these results provide important insights into how differences in trust over BA can affect software projects, more importantly, how they influence stakeholders' behaviour when using the BA. In our case, this behaviour can be detrimental because the complementary artefacts can remain unknown and possibly needed by other stakeholders. We believe that stakeholders might have other behaviours in a different context and use the BA in other ways or not even use them. We discuss the findings and potential solutions in the following section.

5. Discussion

This study aimed to understand the implications of differences in the perception of trust in BAs on software projects and their influence on stakeholders' behaviour. Our findings suggest that most negative implications and changes in stakeholders' behaviour occur because of inefficient management of the BA.

By abstracting the case results, we can establish a conceptual idea of how the trust cycle of boundary artefacts functions. We state that trusting beliefs about the BA influence how the stakeholders use the BA (see Fig. 6). Negative implications might emerge when practitioners do not perceive the BA as favourable in at least one of the investigated trusting beliefs, reliability, functionality, and predictability.

Thus, the trust cycle is generalisable to different software development companies with different sizes. The negative implications resulting from the inconsistencies, though, can be the same or others because they are conditional to the context. For example, we did not find implications regarding the functional factor. Other contexts or different BAs can reveal them — also, different behaviours.

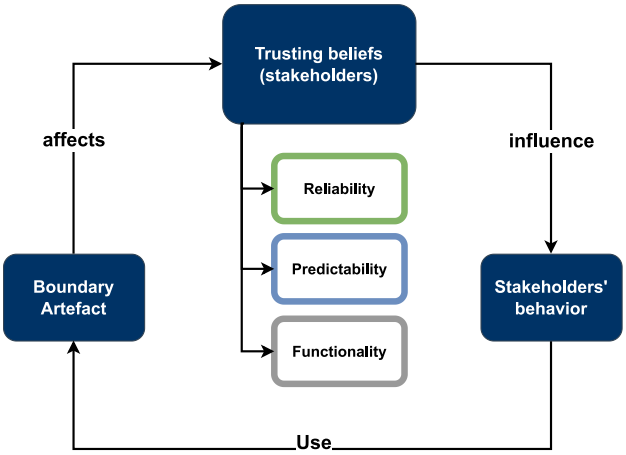


Fig. 6. Trust cycle.

As the BA carries many inconsistencies, the stakeholders can show less confidence, reducing their trust in the BA. They will use the BA differently from how it was supposed to be used or not use it all. For that reason, building up the value of the BA by increasing its trust levels is critical when driving positive behaviour from stakeholders.

In our case, the partial control over the BA has caused several negative implications and influenced the stakeholders' behaviour in creating complementary artefacts. Given that there is a constellation of different teams contributing to and utilising the BA, we believe that adopting a cooperative approach where the management of the BA is shared between practitioners is a good practice for managing the BA and keeping it trustworthy. Dividing the management work can reduce the workload compared to leaving the job to one or a few individuals.

Also, the studied BA already has partial management of the content. Our suggestions lie in the distribution of the work for the whole artefact. One example of this approach would be to divide the handling of the BA between the content and the technical owner when having machine-readable artefacts.

To collaborate with this approach, we provide potential solutions on two main levels. The first refers to the scope of the multidisciplinary team that currently owns the BA. The second level concerns the creation of the BA, which aspects should be observed.

Management level

Having a **formal strategy for managing the entire content** prevents inconsistencies in the BA. This process can be done by assembling interconnected actors and defining roles and responsibilities for each part of the BA's content to provide correct and timely information [3,33].

The incorrect information has caused complications to test execution, be it manual or automated. Every time spent to fix those punctual issues with the BA can generate avoidable loops and increase costs for corrective procedures [34,35].

Such complications can be minimised by **periodic evaluation and feedback processes**. Stakeholders can evaluate the BA regarding the trust aspects through a questionnaire, for example. In addition, the creation of ways to receive and implement feedback can avoid reliability issues and prevent the need for accountability due to outdated or wrong content.

Content creation level

The contributors of the BA have limited knowledge about the stakeholders and the different usage of the content provided. Yet, one crucial aspect of the BA is that the content represents its stakeholders' needs and is a fundamental part of the BA's design. We believe **mapping the stakeholders, and their needs** is a key point for designing BAs. It requires negotiations between contributors and stakeholders to reach a consensus about the multiple domains of stored knowledge [5,36,37].

Occasionally, terminology causes confusion and misinterpretation. When building complex products, the amount of knowledge and its types are often large, creating different terminology levels. The literature reports three main **approaches for dealing with terminology** in BAs, which vary depending on the terms' novelty level [35].

- Syntactic boundary: usually, a common lexicon is sufficient to address the differences — other techniques include taxonomies and store and retrieval technologies.
- Semantic boundary: there is a need for common meanings to create shared meanings to address the differences and dependencies that originated from the terms' novelty. Other techniques: cross-functional interactions/teams and boundary spanners/translators.
- Pragmatic boundary: demand for artefacts that can be jointly transformed. Prototyping is a technique to be used.

Accommodating Experimental and Evolutionary Content is essential for the case we focus on. It seems counter-intuitive to have such flexibility when BA are highly structured from its original characterisation. However, software development is a dynamic activity that usually welcomes change while developing and producing experimental content.

In the case we focus on, there is a frequent need to test new features that are not yet implemented; thus, it is not in the formal BA. Therefore, it is important to be flexible to accommodate this type of content that can be changed or removed. Otherwise, the content will remain within the teams and probably not be known by others. In addition, as technology evolves, the need to create content in different formats also changes. For that purpose, **foresee how new content adapts** becomes valuable.

Our study revealed how the differences in trust in BAs affect software projects in different areas of the organisation and interfere with the task execution of various stakeholders. The decrease in trust results from inconsistencies in the content associated with the lack of management of the BA. A structured strategy for representing and managing a BA's content seems appropriate to increase trust levels and efficiency.

It should be noted that we have not investigated the impact of the negative implications on the software project. Although we revealed them, we do not know how severe they are. A future study could examine how to measure the implications' severity level. The results could support practitioners in prioritising activities for making adjustments and fixes. This is particularly important when dealing with large and well-established BAs that would require an excessive workload for restructuring them.

Another aspect we believe is important for better-designing BAs is investigating the notions of value for different stakeholders. A qualitative study exploring the stakeholders' perceptions of value and how to utilise them could shed light on creating BAs. The nuances of value can differ and provide different aspects that can offer additional value to the stakeholders rather than only the correct content, for example. We believe that if the artefact is well-planned from the beginning, there is a chance that fewer issues will occur regarding how the artefact supplies the stakeholders' needs.

Lastly, the third research direction. We suggest replications of this study to contribute to building a body of knowledge on implications to software projects when practitioners do not perceive the BA as favourable concerning trusting beliefs. Other cases could help us classify negative implications by company size, number of stakeholders, BAs not machine-readable, etc.

6. Threats to validity

In this section, we discuss a number of threats to the validity of our study. We followed the guidelines recommended by Runesson and Höst [29], who classify such threats as a construct, internal, external, and reliability threats.

Construct validity refers to how the theoretical constructs investigated are represented by what the researcher has in mind and what was actually investigated during the interviews. In this aspect, we identified one threat: the interpretation of the criteria to evaluate trusting beliefs such as reliability, functionality and predictability. To reduce this threat, during the interviews, we defined each criterion. For each of them, we had questions that helped the practitioner understand them in the context of information in the artefacts (see Fig. 2).

The internal validity concerns whether a third factor might be affecting the factor in the investigation besides the ones already identified. As we chose an exploratory study, this threat is minimised because we do not use pre-defined concepts to be confirmed. To diminish this threat, we utilised a semi-structured interview guide that allowed the practitioners to speak spontaneously. In addition, we validated our findings in a workshop that worked as both a sanity check and additional data collection.

External validity relates to the extent to which it is possible to generalise the study's results to other cases. Also, to what extent do the findings stimulate the interest of other practitioners. As a case study, our results provide analytical generalisation through plentiful contextual information and discussion of the findings [29,38]. The findings of this study extend to cases with common characteristics, such as companies that develop hardware and software and utilise a BA to store and distribute information about all features throughout the development process.

Concerning the reliability threat, we discuss potential threats related to how the analysis depends on a specific researcher. To minimise this threat, we provided an extensive description of our research approach and a detailed explanation of the coding process with examples of the raw data. Also, we offered the semi-structured interview guide as a transparency action.

7. Concluding remarks

Boundary artefacts support task execution for many stakeholders within a software development project. However, if practitioners do not trust the artefact, they will probably not rely on them or have different behaviour towards them.

We conducted an empirical investigation through a case study to reveal how people create, utilise and manage BAs. Moreover, we observed how differences in trust affected software projects and how they change the stakeholders' behaviour.

Our investigation has shown that practitioners who add new and adjust existing content do not entirely understand the stakeholders' needs. In addition, the management of the content is partial. These findings indicate that parts of the content and stakeholders' needs are not supervised, impacting trust levels.

When practitioners do not perceive the BA as favourable in at least one of the investigated trusting beliefs, specifically reliability and predictability, the stakeholders cannot execute their tasks appropriately, and several implications affect the software project development. Additionally, the stakeholders create workarounds to supply their needs. Among those, they create complementary artefacts within their teams.

To function as planned, organisations need to increase the level of trust of their BAs, and the proper management of the content is crucial for increasing this trust level. This management should include creating content based on stakeholders' needs and monitoring changes. Future research could explore and develop other ways for supporting practitioners in creating and managing such artefacts in software development environments.

CRediT authorship contribution statement

Raquel Ouriques: Conceptualization, Methodology, Investigation, Formal analysis, Resources, Data curation, Validation, Writing – original draft, Review & editing, Visualization. **Fabian Fagerholm:** Conceptualization, Validation, Writing – review & editing, Supervision. **Daniel Mendez:** Conceptualization, Validation, Writing – review & editing, Supervision. **Baldvin Gislason Bern:** Writing – reviewing.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.infsof.2023.107170.

Data availability

The data that has been used is confidential.

Acknowledgements

We want to thank all practitioners at Axis who participated and contributed to this study. We would also like to acknowledge that this work was supported by the KKS foundation through the S.E.R.T. Research Profile project at Blekinge Institute of Technology.

References

[1] J. Sapsed, A. Salter, Postcards from the edge: local communities, global programs and boundary objects, Organ. Stud. 25 (9) (2004) 1515–1534.
[2] M. Briers, W.F. Chua, The role of actor-networks and boundary objects in management accounting change: a field study of an implementation of activity-based costing, Account. Organ. Soc. 26 (3) (2001) 237–269.
[3] R. Wohlrab, P. Pelliccione, E. Knauss, M. Larsson, Boundary objects and their use in agile systems engineering, J. Softw.: Evol. Process 31 (5) (2019) e2166.
[4] R. Ouriques, K. Wnuk, T. Gorschek, R.B. Svensson, The role of knowledge-based resources in Agile Software Development contexts, J. Syst. Softw. 197 (2023) 111572, http://dx.doi.org/10.1016/j.jss.2022.111572.
[5] D. Akoumianakis, N. Vidakis, G. Vellis, D. Kotsalis, G. Milolidakis, A. Plemenos, A. Akrivos, D. Stefanakis, Transformable boundary artifacts for knowledge-based work in cross-organization virtual communities spaces, Intell. Decis. Technol. 5 (1) (2011) 65–82.
[6] B.A. Bechky, Crossing Occupational Boundaries: Communication and Learning on a Production Floor, Stanford University, 1999.
[7] K. Henderson, Flexible sketches and inflexible data bases: Visual communication, conscription devices, and boundary objects in design engineering, Sci. Technol. Human Values 16 (4) (1991) 448–473.
[8] C.P. Lee, Boundary negotiating artifacts: Unbinding the routine of boundary objects and embracing chaos in collaborative work, Comput. Support. Coop. Work (CSCW) 16 (3) (2007) 307–339.
[9] N. Levina, E. Vaast, Turning a community into a market: A practice perspective on information technology use in boundary spanning, J. Manage. Inf. Syst. 22 (4) (2014) 13–37.
[10] G. Vidotto, D. Massidda, S. Noventa, M. Vicentini, Trusting beliefs: A functional measurement study, Psicol.: Int. J. Methodol. Exp. Psychol. 33 (3) (2012) 575–590.
[11] J.K. Blomkvist, J. Persson, J. Åberg, Communication through boundary objects in distributed agile teams, in: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, 2015, pp. 1875–1884.
[12] J. Lansing, A. Sunyaev, Trust in cloud computing: Conceptual typology and trust-building antecedents, ACM Sigmis Database: Database Adv. Inf. Syst. 47 (2) (2016) 58–96.
[13] T. Gustavsson, Visualizing inter-team coordination, in: Proceedings of the Evaluation and Assessment in Software Engineering, 2020, pp. 306–311.
[14] T.L. Huber, M.A. Winkler, J. Dibbern, C.V. Brown, The use of prototypes to bridge knowledge boundaries in agile software development, Inf. Syst. J. 30 (2) (2020) 270–294.
[15] P. O'Raghallaigh, S. Lane, F. Adam, D. Sammon, Difficult knowledge boundaries during requirements determination, J. Decis. Syst. 29 (sup1) (2020) 270–284.
[16] J.R. Whitson, Voodoo software and boundary objects in game development: How developers collaborate and conflict with game engines and art tools, New Media Soc. 20 (7) (2018) 2315–2332.
[17] S.L. Star, J.R. Griesemer, Institutional ecology,translations' and boundary objects: Amateurs and professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39, Soc. Stud. Sci. 19 (3) (1989) 387–420.
[18] S.L. Star, Chapter 2 - The structure of ill-structured solutions: Boundary objects and heterogeneous distributed problem solving, in: L. Gasser, M.N. Huhns (Eds.), Distributed Artificial Intelligence, Morgan Kaufmann, San Francisco (CA), 1989, pp. 37–54.
[19] D. Méndez Fernández, W. Böhm, A. Vogelsang, J. Mund, M. Broy, M. Kuhrmann, T. Weyer, Artefacts in software engineering: a fundamental positioning, Softw. Syst. Model. 18 (5) (2019) 2777–2786.
[20] S. Leigh Star, This is not a boundary object: Reflections on the origin of a concept, Sci. Technol. Human Values 35 (5) (2010) 601–617.
[21] E. Subrahmanian, I. Monarch, S. Konda, H. Granger, R. Milliken, A. Westerberg, et al., Boundary objects and prototypes at the interfaces of engineering design, Comput. Support. Coop. Work (CSCW) 12 (2) (2003) 185–203.
[22] A. Zaitsev, U. Gal, B. Tan, Boundary Objects and Change in Agile Projects, ACIS, 2014.
[23] D.H. McKnight, L.L. Cummings, N.L. Chervany, Initial trust formation in new organizational relationships, Acad. Manag. Rev. 23 (3) (1998) 473–490.
[24] G.M. Marakas, R.D. Johnson, J.W. Palmer, A theoretical model of differential social attributions toward computing technology: when the metaphor becomes the model, Int. J. Human-Comput. Stud. 52 (4) (2000) 719–750.
[25] N. Paravastu, D. Gefen, S.B. Creason, Understanding trust in IT artifacts: An evaluation of the impact of trustworthiness and trust on satisfaction with antiviral software, ACM SIGMIS Database: DATABASE Adv. Inf. Syst. 45 (4) (2014) 30–50.
[26] H. Koelmann, Perspectives on information technology artefacts in trust-related interactions, in: International Conference on Human-Computer Interaction, Springer, 2020, pp. 445–457.
[27] J.B. Thatcher, D.H. McKnight, E.W. Baker, R.E. Arsal, N.H. Roberts, The role of trust in postadoption IT exploration: An empirical examination of knowledge management systems, IEEE Trans. Eng. Manage. 58 (1) (2010) 56–70.
[28] C.B. Seaman, Qualitative methods in empirical studies of software engineering, IEEE Trans. Softw. Eng. 25 (4) (1999) 557–572.
[29] P. Runeson, M. Höst, Guidelines for conducting and reporting case study research in software engineering, Empir. Softw. Eng. 14 (2) (2009) 131–164.
[30] M.B. Miles, A.M. Huberman, J. Saldaña, Qualitative Data Analysis: A Methods Sourcebook, Sage publications, 2018.
[31] Swedish Research Council, Good Research Practice, Swedish Research Council, Stockholm, Sweden, 2017, p. 86.
[32] P. Allmark, J. Boote, E. Chambers, A. Clarke, A. McDonnell, A. Thompson, A.M. Tod, Ethical issues in the use of in-depth interviews: Literature review and discussion, Res. Ethics 5 (2) (2009) 48–54, http://dx.doi.org/10.1177/174701610900500203.
[33] N. Levina, E. Vaast, The emergence of boundary spanning competence in practice: Implications for implementation and use of information systems, MIS Q. (2005) 335–363.
[34] E.O. Luttikhuis, J. de Lange, E. Lutters, R. ten Klooster, Evolving product information in aligning product development decisions across disciplines, Procedia CIRP 29 (2015) 573–578.

[35] P.R. Carlile, Transferring, translating, and transforming: An integrative framework for managing knowledge across boundaries, Organ. Sci. 15 (5) (2004) 555–568.
[36] E.-M. Kern, W. Kersten, Framework for internet-supported inter-organizational product development collaboration, J. Enterp. Inf. Manag. (2007).
[37] E. Hutchins, The social organization of distributed cognition, in: L.B. Resnick, J.M. Levine, S.D. Teasley (Eds.), Perspectives on Socially Shared Cognition, American Psychological Association, 1991, pp. 283–307.
[38] B. Flyvbjerg, Five misunderstandings about case-study research, Qual. Inquiry 12 (2) (2006) 219–245.