In Continuous Software Development, Tools Are the Message for Documentation

Theo Theunissen[1], Stijn Hoppenbrouwers[1,2] and Sietse Overbeek[3]
[1]*HAN University of Applied Sciences, Department of ICT, Arnhem, The Netherlands*
[2]*Radboud University, Institute for Computing and Information Sciences, Nijmegen, The Netherlands*
[3]*Utrecht University, Department of Information and Computing Sciences, Utrecht, The Netherlands*

Keywords: Agile, Continuous Software Development, DevOps, Documentation, Lean.

Abstract: In Continuous Software Development, a wide range of tools are used for all steps in the life cycle of a software product. Information about the software product is distributed across all those tools and not stored in a central repository. To better understand the software products, the following media elements must be taken into account: the types of information, the tools, tool-stacks and ecosystems to manage the (types of) information, and the amount of structure. In the tile, "tools" refers to the phrase "the medium is the message", coined by McLuhan and Fiore (1967) pointing that the medium should be subject of investigation as well as the content of the message. In this paper the tools include tool stacks, ecosystems, the types of information and amount of structure; they define the content of the message. Our approach to present relevant information to different stakeholders is rooted in understanding and utilizing these aspects. In this respect, the amount of structural variety of information defines the value for information creation and retrieval, including the tools to process that information. Documentation is considered an information type that is processed through tools in a software development ecosystem.

1 INTRODUCTION

In traditional software development, the main tools for developers are limited to a small number of tools such as an Integrated Development Environment (IDE) and Source Code Management (SCM). In modern software development approaches such as Lean, Agile, and DevOps, an increase can be observed in the overall number of tools developers are using. There are many tools for project management, ranging from simple task management tools like Trello, to enterprise project management with Jira, or a wide range of IDEs such as VSCode, Eclipse, or IntelliJ. Even categories used for classifying tools are manifold, ranging from data management to development tools and from deployment tools to monitoring tools (Kersten, 2018; Beshawred, 2020). Information about software is scattered throughout all the tools used in a software development ecosystem. For most software products, there is no single repository that contains all information about the software. For instance, the core concept of a software product may be presented in PowerPoint-like tools. Information about stakeholder concerns, risks, constraints, and context may be documented in Word-like documents, modifications on source code are often maintained in git, and deployment documentation may be defined as executable infrastructure-as-code. These examples show that the type of information documented has a strong relationship with the tool it is stored in and used with. These tools often define the format of the information. The format can range from structured text to video. Documentation in modern software development concerns the creation of information about the software product, conveying knowledge about the software product, and in some cases even executing the documentation. The scattered information that is stored in the myriad of tools introduces issues of retrieval and comprehension of relevant information about the software with respect to the stakeholders involved. The phrase "tools are the message" concerns:

1. the **types** of information documented,
2. the amount of structural **variety**: whether the information is structured (source code, templates)

Theunissen, T., Hoppenbrouwers, S. and Overbeek, S.
In Continuous Software Development, Tools Are the Message for Documentation.
DOI: 10.5220/0010367901530164
In *Proceedings of the 23rd International Conference on Enterprise Information Systems (ICEIS 2021) - Volume 2*, pages 153-164
ISBN: 978-989-758-509-8

or unstructured (sketches, text),

3. the myriad of **tools** used, including tool categories organized into stacks and ecosystems, and
4. **comprehensibility** support of the software product.

To better understand why tools are the message, we introduce the umbrella term "Continuous Software Development (CSD)" that covers the characteristics of Lean, Agile, and DevOps software development approaches. First, it covers the values, principles, practices, processes, and tools for Lean, Agile, and DevOps. Second, CSD embraces the whole lifecycle of software as a product, from conception to end-of-life. This includes continuous design and architecting, and also development until retirement. Third, CSD takes into account the drivers behind the continuously changing state of the software product, such as progressive insights, contextual changes, new features, bug fixes, or other unforeseen factors. Last, information about the software is distributed across the many tools used in a software development ecosystem. "The tools are the message" will be explained by showing how the types of information documented relate to the tools used for information creation, retrieval, and execution.

The scientific contribution of this paper is the insight that the requirements for documentation in lean, agile and DevOps are present in values of these methods, as well as in the community of practice of industrial software engineers. We present the generic requirements and conditions for documenting and communicating contained in CSD knowledge. Our findings lead to approaches for knowledge preservation in CSD.

In the remainder of this paper, the following subjects will be addressed. In Section 2, the study design is presented. In Section 3 the data collection, data analysis and data interpretation are shown. In Section 4, the types of information are discussed. "Tools are the message" is discussed in Section 5. The paper ends with conclusions in Section 6.

2 STUDY DESIGN

In this study, we use multiple sources for data collection and data analysis to enhance its credibility. We will use Multivocal Literature Review (MLR) following Garousi et al. (2019) and we follow Yin (2002) for the case study design.

2.1 Multivocal Literature Review (MLR)

The motivation for conducting an MLR is that the investigation of documentation in CSD is relatively new. Many state-of-the-art values, principles, knowledge, practices, tools and processes are shared in, for example, blogs, (online) lectures, data sets, and (technical) reports. Therefore, it is important to take these sources into account besides the available body of literature on the topic (Garousi et al., 2019).

2.1.1 Quality Assessment Criteria for Data Sources

In Table 1, the columns with 'grey' and 'black' literature types are not ordered by rigour. Added to this list are git software repositories, as they contain information on documentation about software. The selected

Table 1: Types of literature, based on Garousi et al. (2019).

'White' literature	'Grey' literature	'Black' literature
• Published journal papers • Conference proceedings • Peer-reviewed books	• Preprints • Technical reports • Lectures • Data sets • Podcasts, Video • Blogs • git SCM	• Ideas • Concepts • Thoughts

repositories, displayed in Table 2, are open source with a hundred to a thousand contributors. They deliver high quality software that has often been running for years.

2.1.2 Inclusion and Exclusion Criteria

Search engines used for the MLR, based on popularity (NetApplications, 2020) and triangulation, are:

- Google. 70.0% popularity on Desktop, 94.0% on Mobile devices.
- Bing. 13.2% popularity on Desktop, 0.7% on Mobile devices.
- DuckDuckGo (DDG). 0.3% on Desktop, 0.2% on Mobile devices.
- Google Scholar (GS). Not mentioned by NetApplications (2020). It was used for triangulation to compare with academic search engines.

Google Scholar was added to verify the results from the other search engines. Furthermore, if Google Scholar returns few search results, then search engines like ACM, IEEE, ScienceDirect, Springer or WebOfScience often also show few results. Other search engines mentioned by NetApplications (2020) are powered by Google (AOL) or Bing (Yahoo) and therefore considered redundant. Only western search

Table 2: Description of Data sources and Types used in this paper and contribution to answering the research questions.

ID	Source	Type	Description	RQ1	RQ2	RQ3	RQ4
S1	bash (Open Source Community, 2020a)	SCM	Open source project for a Linux shell		✓		✓
S2	Mozilla open source project (Open Source Community, 2020b)	SCM	Open source project for Firefox, a web browser		✓		✓
S3	LaTeX open source project (Open Source Community, 2020c)	SCM	Open source project for a document editing tool		✓		✓
S4a	stackshare.io (Beshawred, 2020)	Data set	Website that collects user generated data on popular development and deployment stacks	✓			✓
S4b	thoughtworks.com/radar (Fowler, 2020)	Data set	An opinionated guide to technology frontiers	✓			✓
S4c	gartner.com (Gartner, 2020)	Data set	Interpreting technology hype	✓			✓
S5	Case study	Data set	Interviews with practitioners from the industry	✓	✓	✓	✓
S6	Documents	Data set	Documents that support the interviews	✓	✓	✓	✓

engines were included; Asian search engines were excluded.

Garousi et al. (2019) refers to the extensive assessment of sources, compared to 'white literature'. In Table 3, the inclusion criteria are presented. For definition of the numbers, the threshold is randomly selected, or top ranking is applied. The numbers for the thresholds are set to cover the research questions.

Table 3: Inclusion Criteria, following Kitchenham and Charters (2007); Garousi et al. (2019).

ID	Criteria
C1	Open source projects, including number of developers. The threshold is 10 developers.
C2	Open source projects, including number of years existing. The threshold is 10 years.
C3	Open source projects, including format of documentation. Range of formats over projects are selected to cover the variety of information.
C4	Community generated data data sets including metrics for stacks. The top three are taken into account.
C5	Community generated data data sets including metrics for companies.The top three are taken into account.
C6	Community generated data data sets including metrics for developers. The top three are taken into account.

In Table 3, C3 concerning the formats, refers to the following formats: HyperText Markup Language (HTML), Compiled HTML Help (CHM), Rich Text Format (RTF), Portable Document Format (PDF), LaTeX, PostScript, man pages, DocBook, Extended Markup Language (XML), and ePub.

2.1.3 Search Process

The search terms on all three search engines are: "`stacks tools technology`". In the test run, it became clear that the search results for Google Scholar were useless, unless the term 'software' was added. The final search string was "`stacks tools technology software`". Each time, an anonymous browser session was started to minimize hints from the search engines based on previous searches. Different search engines use different metrics to calculate the result set. The number designates an indication, not a rank.

Table 4: Inclusion Criteria, based on Garousi et al. (2019).

Search Engine	Number of hits	Top three results
Google	47.900.000	Stackshare homepage • Popular Tech Stacks from stackshare • Technology Stack: What it is and how to build one on mixpanel.com
Bing	141.000.000	Popular Tech Stacks from stackshare • Stackshare homepage • https://mopinion.com/tools-for-your-2019-marketing-technology-stack/
Duck Duck Go	Not calculated	Popular Tech Stacks from stackshare • Stackshare homepage • Top six stacks from fingent.com
Google Scholar	289.000	"What are developers talking about?" from Springer • "Singularity: rethinking the software stack" from ACM • "Managing the Life-Cycle of Linked Data with the LOD2 Stack" from Springer

2.2 Case Study

We follow Yin (2002) for the case study approach. A case study is appropriate for research that answers "why" and "how" questions. This study does not require control of behavioral events, and focuses on contemporary phenomena (Yin, 2002).

The following types of research are derived from Yin (2002).

- Explanatory.
  Description of cause-effect relations used with inductive reasoning and often used with descriptive statics. This type of research is applicable because, for understanding relations between real-life phenomena, the interviewees can elaborate on the phenomenon and their relation. Typical questions start with 'who', 'what', 'where', 'how many' and 'how much'.
- Descriptive.
  Takes the context into account by scientifically reporting observations about situations and events. In real-life situations, situations and events can not be considered without their context. Typical questions start with 'why' and 'how'.
- Exploratory.
  Defines and tests hypotheses for building new theories. For this paper, exploring and building new theories is not the primary objective, but the collected data can serve in follow-up research. Typical questions starts with 'why' and 'how'.

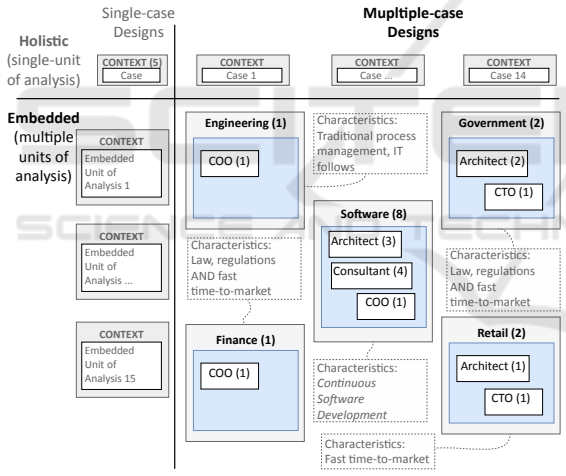

Figure 1: Units of analysis -individual practitioners- with cases from multiple types of organizations, including multiple departments, concerning documentation in CSD.

Figure 1 depicts the case design and units of analysis.

2.2.1 Units of Analysis

The units of analysis are individual practitioners from (non-)profit national and international organizations in senior positions. Professionals in IT-industry are relevant because IT is the domain of research. The reason for doing individual interviews is that they can provide in-depth information, and individual or even opinionated perspectives on matters of concern. Also, for practical reasons, it is easier to arrange a meeting with individuals than with groups. Concerning seniority, practitioners have an overview of the continuity of software projects over the years, including software products' evolution. This includes changing the technology stack, organizational change, and the IT tooling landscape. There is no particular reason for including national or international organizations. Practical reasons such as availability are decisive in selecting organizations. The teams' and organizations' size is relevant because knowledge about the software product is more present in larger teams, including historical knowledge about decisions, bugs, and bug fixes.

2.3 Objectives and Research Questions

The *objective* of this study is defined in the main research question:

> *The objective is to investigate the necessary and sufficient conditions to organize information scattered throughout a CSD ecosystem into comprehensible documentation for designated stakeholders.*

The related research questions are:
- RQ1: **Which tools** are used in the software development ecosystem?
- RQ2: What is the **variety** of information that is stored in tools?
- RQ3: Which information is stored with what tool?
- RQ4: How can this scattered information be organized into **comprehensible documentation**?

3 RESULTS

In this section, the data collection methods, data analysis methods and data interpretation methods are presented, together with the data.

3.1 Data Collection

Table 2 presents the sources and types for the data used in this paper. From a wide range of popular open source projects, S1-S3 were selected because of their contribution to answering RQ2 concerning variety of information and RQ4 concerning organizing information into comprehensible documentation. Sources S4-S6 contribute to answering RQ1, namely which tools are used in which ecosystem. From the publicly available data sets that present the popularity of tool stacks, stackshare.io was the only viable option. Alternatives, such as stack.g2.com or www.producthunt.com show limited sets. Sites

such as alternativeto.net show only alternatives for a specific tool. The data from stackshare was compared with information collected from the interviews. Sources S5-S6 contribute to answering all research questions and were selected to validate and extend the data collected from S1-S4.

3.1.1 Data Collected from the Source Code

The source code from the open source project was selected because repositories with a long development period also have numerous developers. New developers build on existing code and sometimes need to work through a wide range of standards and guidelines, architecture, decisions, UI, or enforced coding standards by the use of code linters (a linter is a static analysis tool that warns for or prohibits deployment when code is not following styles or constructs, or includes programming errors). Examples are mediawiki and Linux.

The repositories presented in Table 2 are selected because they differ in the **type of medium** for documentation. This contributes to answering RQ2.

3.1.2 Data Collected from the Data Sets

Three sources are used: S4-S6 in Table 2. The tools from stackshare.io are used to get an overview of popular tools and tool stacks. The data retrieved from the interviewees verifies the popularity of tools in tool stacks. Furthermore, the number of tools in tool stacks mentioned by interviewees is higher and gives insight into the selection of tools.

3.1.3 Interviews

We held 14 interviews (S5 in Table 2) with 15 subjects in 5 different cases. The cases were selected based on the expected motivation for documentation, ranging from fast time-to-market (TTM) to following regulations, as mentioned by Bass et al. (2015). Figure 1 depicts the units of analysis (interviewees) with multiple cases (organizations with different motivations for documentation).

3.2 Data Analysis

In this section, filtering, grouping, ordering, and visualization of data is presented. The filtering applies to a selection of data that falls within this study's scope and contributes to answering the research questions. Discussions and insights in the interviews that do not contribute are not taken into account. The grouping of the data applies to combine results in aggregated classes with common properties. Part of the grouping matches with the subsequent research questions. The ordering refers to the relevance of visual properties. With the visualization, a quick and comprehensive overview is presented from relevant data.

In Figure 2, the types of organization are represented. Based on Bass et al. (2015), it was expected that the motivation for documentation ranges from fast TTM, such as in retail, to required documentation because of regulations, such as in government or finance. Software companies are defined having IT as their core competence, and having more than 80% of their revenue generated by IT. Modern companies use their digital infrastructure platform as their source of revenue, but none of the interviewed companies generates revenue from a platform.

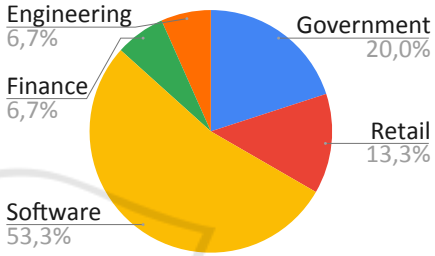

Figure 2: Types of organization from the interviewees.

In Figure 3, the functions of the interviewees are represented. All 15 interviewees have senior positions, either as some kind of manager or some kind of technician. Consultants are technical consultants, not business consultants although the senior consultants have a wider span of control and skills than IT only. C-level interviewees all had an education in IT and were seasoned IT practitioners.

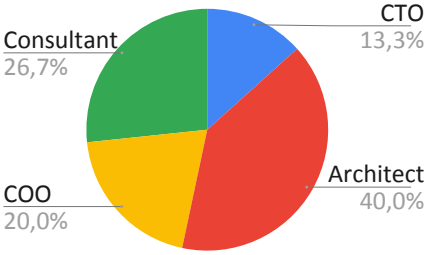

Figure 3: Functions of interviewees.

Figure 4 presents on the x-axis the number of tools in a tool stack, based on numbers from stackshare.io. The left y-axis represents the number of found tool stacks with the number of tools. Combinations most found numbers three. There are 95567 combinations of three tools (blue line). The red line depicts the number of most popular combinations. A further

analysis of the data is in the Appendix[1]. The table represents popular combinations and the number of combinations. The most popular combination is GitHub with nginx and Redis. This combination is mentioned 47 times by the stackshare.io. If the combination of tools is 11 or more, there are fewer combinations of tools in tool stacks. Another interesting observation is that productivity tools are more popular than communication or documentation tools. The first appearance of a communication tool is the chat application Slack, with a number of 5 tools or more. Trello, as task management tool, is mentioned when a tool stack is built up of 8 tools or more. The first time a documentation-like tool (Markdown) is mentioned is when a tool stack consists of 13 tools or more.

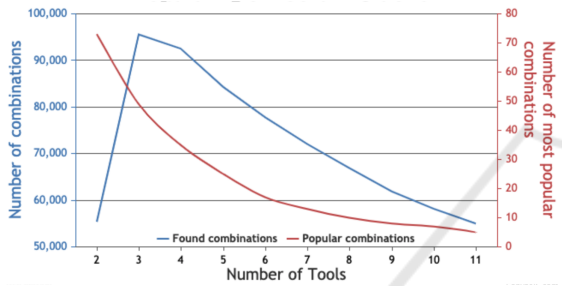

Figure 4: Tools and tool combinations, based on stackshare.io and theotheunissen.nl/tools-are-the-message.

Based on the interviews, the number of tools was higher than the popularity index in Figure 4. Interviewees mentioned a full range of tools that are either prescribed and supported or tools that are allowed for individual developers. An example of this is that the tool stack from JetBrains is prescribed and supported but developers have the option to use another IDE (VSCode). Another example is Postman for the testing of REST API endpoints. This tool is allowed, can be highly productive for individual developers but is not supported by the company. Communication and collaboration tools are not allowed to be individual choices but are prescribed by the company.

Figures 5 and 6 were initially compiled from Theunissen et al. (2020). The figures were presented in the interviews for validation and extended with other types of information.

3.3 Data Interpretation

With data interpretation, the research questions are answered based on the analyzed data. Also, deviations from expectations are discussed.

One deviation from the expectation is the motivation for documentation, which affects multiple cases in the design. Bass et al. (2015) mention a range of motivations between fast TTM for retail to following regulations for government. This range defines the multiple cases in the design. However, although government and finance have to apply to strict regulations, the need for fast TTM is immanent. For governmental organizations in the Netherlands, national elections held at least every four years result in new policies that have to be implemented at short notice. For governmental organizations, no trade-off is possible. These organizations have to comply to law and regulations, and follow policies. The workaround is to automate development as much as possible and fill the gaps with manual operations. Financial companies also have to conform to regulations and standards such as AMLD5[2], GDPR[3] or the Gramm-Leach-Bliley Act [4]. The notion of 'fast TTM' applies to keeping up with regulations. Fast TTM does not refer to adding up-selling and cross-selling features in a web shop for the financial industry. The effect on the multiple cases in the case study design is that the cases for governmental organizations and the financial industry are identical and not different cases. However, this does not affect the answering of the research questions.

A second observation is that interviewees mention many more tools than are mentioned on stackshare. The sources contributing to answering RQ1 are the online user generated tool stacks with tools from stackshare.io (S4a). These results where verified, validated, and extended by the interviews (S5) and supporting documents (S6). Figure 4 shows the most popular sets consist of three tools. There are 95567 stacks with three tools. However, 69262 stacks are only mentioned once with three tools, making the number of popular stacks a long tail. Interviewees mentioned over 20 tools. This deviation has a small effect on RQ1 -which tools are used- because the range of tools is retrieved and confirmed with interviewees. For RQ3, which type of information with what tool, it has a larger effect because interviewees could elaborate on the motivation. Stackshare.io only shares numbers, not motivations. The interviewees all mentioned more tools than could be extracted from stackshare and the tools from stackshare are also mentioned by interviewees. The effect from the interviewees is only qualitative because they can motivate choices, describe relations between tools, or refer to organizational policies.

[1]Appendix on https://theotheunissen.nl/tools-are-the-message

[2]https://eur-lex.europa.eu

[3]https://gdpr-info.eu

[4]https://ftc.gov

4 TYPES OF INFORMATION

In this section, the types of information in CSD are presented. There is a distinction between information and documentation. The term **information** is used to refer to any (set of) symbols that

1. makes a difference (Shannon and Weaver, 1949; Peirce, 1992) and
2. causes one or more effects (Pawłowski et al., 2009).

The term **documentation** is used for any written, verbal, visual artifact or activity that transfers knowledge between stakeholders, related to the software product (Wagenaar et al., 2018). Documentation stems from the etymological meaning for (Cicero et al., 2001):

1. teaching (Latin: *docere*),
2. pointing out, or
3. instructing with evidence and authority.

The types of information are the distinctive properties for what is actually documented, at what point in the process in CSD, how it is stored, and why it is relevant to keep the information. Figure 5 shows the types of information, tool categories, and examples of tools.

5 TOOLS ARE THE MESSAGE

Marshall McLuhan coined the phrase "the medium is the message" (McLuhan and Fiore, 1967). He pointed out that the medium should be subject of investigation as well as the content of the message. The subject for this study concerns both the tools as well as the content of the message, so note that tools are part of the message and not the only message. "The tools are the message" thus refers to a different message being communicated if tools make use of different media types such as written, verbal, or visual. The primary concern for this study is the tools, not the types of information. However, the types of information do have a strong relationship with the tool in which the information is created, retrieved or updated. In this section, the aspects that establish that tools are (part of) the message will be discussed. In Section 5.1 the tools, tool-stacks and software development ecosystems are discussed. Tools define which tools, stacks or ecosystems are used when information is created, captured, understood or processed. In Section 5.2 the variety of information is discussed. The variety refers to the amount of structure information has. This amount of structure is defined by creation and retrieval of information with tools. In Section 5.3, the relation between the types of information, tool (stacks), and the variety of information will be presented as "tools are the message".

5.1 Tools, Tool Stacks, and the Software Development Ecosystem

In this section, the tools in relation to other tools are described. The tools, tool stacks and their popularity can be found in Sources S4 (websites), S5 (interviews), and S6 (supporting documents to the interviews), displayed in Table 2.

In CSD, tools are organized into stacks, and tool stacks are organized into software development ecosystems.

In Figure 6, an overview is presented for the relation between Software Development Ecosystems with Tool stacks and Tools, including components and examples. In the next paragraphs, an explanation will be presented of the figure.

1. **Tools.** A tool is defined as a concept, technology, software system, template, framework, or library to design, develop, and maintain a software product. The level of freedom can define the difference between a template, framework, and library. A (technology) template is like a form where the only freedom exists in filling in the value of variables, such as a Python Django template. A framework is a comprehensive set of methods that prescribe the purpose and usage of functions and methods. The maximum freedom is with libraries that provide a set of functions that can be used to speed up development, e.g., jQuery.
   Examples for concepts are the nature of technology, such as the processes, knowledge, and artifacts, including impact on society (Kipperman, 2009). For technologies: Technology Radar or Gartners Hypecycle. Examples for software systems are typical tools like Confluence and IntelliJ.
2. **Tool Stacks.** The organization of tools into tool stacks for 1) development stacks including back-end, front-end or full-stack, 2) solutions stacks, or 3) company stacks. The *front end* is where the end-user interacts with the system. For web applications, a limited set of technologies is available. For mobile application development, there is no dedicated technology to create the user interface, other than supporting tools to create wireframes or graphic tools to create low-fidelity or high-fidelity User Interface (UI)s. The stack for *back end* development concerns databases, applications, and servers. This also includes the infrastructure the software is running on. This might be

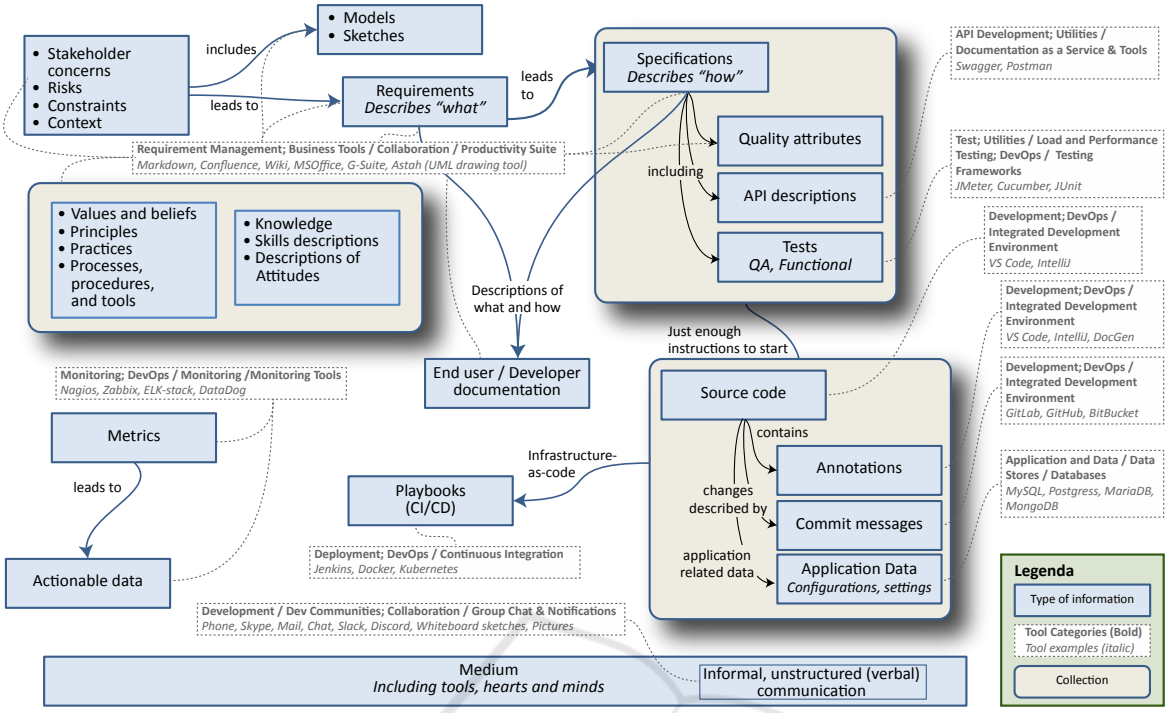

Figure 5: Types of Information, including mapping to tool categories and tools. The medium applies to the communication of information. Metrics and actionable data apply to all information for insight and control of processes.

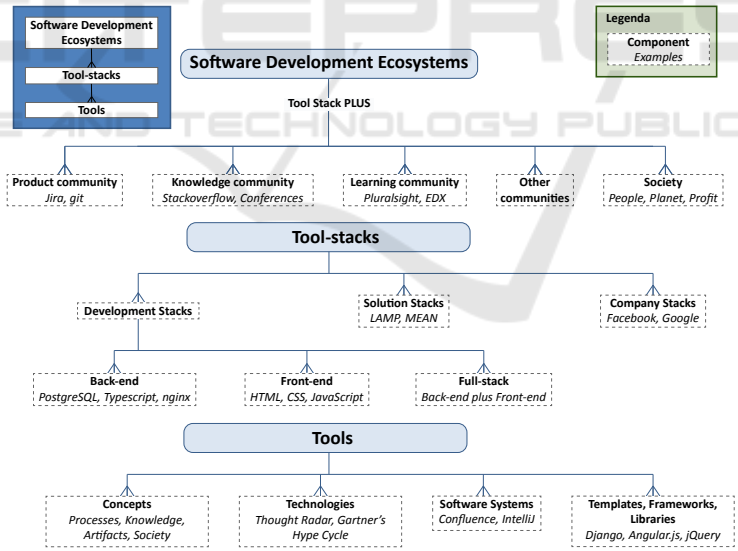

Figure 6: Relations between Software Development Ecosystems with Tool stacks and Tools, including components and examples.

on-premise, in the cloud, or hybrid. A myriad of tools for the back end is available. At stackshare, over 2,000 tools can be found related to back-end development (Beshawred, 2020)[5]. The *full stack* can be viewed as the sum of tools for the back end and front end. However, this simple sum of tools also includes processes, knowledge, and competences, and other types of information and not just the tools but also to understand and manage the complexity of this sum of tools. This fits with the

[5]For this paper, the data can be found on https://theotheunissen.nl/tools-are-the-message

processes in DevOps, where developers and maintainers are the same people. A *solution stack* is a specific set of tools to solve a problem where each tool contributes to the solution, and tools are mutually exclusive. Tools can be exchangeable, for instance MySQL with MariaDB, or Apache with nginx. A *company stack* is a stack of tools used by a company for its specific situation, focusing on fast TTM or following legislation. In this respect, the reason for documentation is relevant. It can lead to very little documentation in case of fast TTM or to mandatory documentation because of regulatory reasons, as in the aviation industry, medicine industry, or tax administration.
Examples for 1) are github and intelliJ. Examples for 2), the solution stack, are Linux, Apache, MySQL, PHP (LAMP), MongoDB, Express.js, Angular (MEAN). For 3), the company stacks, examples are for Facebook: Hack, React, Cassandra, and GraphQL or Google with Dart, Go, Angular.js, and Material Design. The stack as a concept is discussed by Jansen et al. (2009), referring to Software Ecosystems (SECO). This is a set of business functioning as a unit interacting with a shared market for software and services, including relationships. SECO already points to a context, but the community takes the context explicitly into consideration as is discussed in the next paragraph.

3. **Software Development Ecosystems.** This is an extension of the tool stacks with communities, optionally outside the team or company. A *product community* refers to information about specific tools, including concepts, technologies, software systems, and templates, frameworks, and libraries. The community for *knowledge* can be based either on websites where developers gather to exchange questions, answers, and contemplations, or on conferences where developers from industry and scientists meet. There are also meetups specific for industry developers, that are also attended by scientists. This makes clear that knowledge sharing involves, besides cognitive activity, also social activity. The *learning community* includes sites for Massive Online Open Courses (MOOC)s, either academic or commercial, to learn about concepts and practicing code, but also classic learning environments for students at universities.
Examples are product or tool sites. Knowledge sharing sites like Stackoverflow, Reddit, or Quora; online courses from Pluralsight, EDX, or Udemy.

The classification of tools into tool stacks varies from an unorganized landscape such as Kersten (2018) to layers for Application and Data, Business Tools, DevOps, and Utilities such as Stackshare.io (Beshawred, 2020). Other classifications take into account the processes, infrastructure, productivity (4+1 from Kruchten (1995), or C4 from Brown (2014)).

The way the tools are organized into tool stacks, and tool stacks are organized into software development systems, shows that this "scattering" has a high level of organization. The tools are not randomly picked to serve a purpose, but are combined to support design, development, and maintenance for a software product with specific requirements from the industry such as fast TTM or regulations for documentation.

"Tools are the message" concerning the tools, tool stacks and software development ecosystems relates to the type of information that is stored with each tool. The tools in "tools are the message" define how the information is created, stored, retrieved, and communicated. The tool stack is comprised of a set of tools in use by a development team. The software development ecosystem includes the community around individual tools.

5.2 Variety of Information and Contribution to Knowledge Transfer

In this section, the variety of information is discussed, including the relation with tools. This Section is closely related to RQ2: the variety of information. The data sources are S1-S3 (open source projects), S5-S6 (case studies) as represented in Table 2.

The information in the tools in CSD has a certain amount of structure. Figure 7 presents the amount of structure measured by the creation or capturing on the upper x-axis and the retrieval for human communication varying to automated processing on the right y-axis. The tools to create or retrieve the information are on the lower x-axis. The Creation dimension on the upper x-axis varies from constructing to capturing information. "Capturing" refers to the ingestion of information into a storage medium that is not created with a software development tool, e.g. whiteboard sketches, drawings, or models. A photograph might be saved in Jira or Confluence for later usage. The system does captures chat messages or email messages that have a low degree of structure. Probably the only structure an email message has is the subject and other standard headers such as the addressees. "Manufacturing" refers to the manual creation of information, such as source code. The Retrieval dimension on the right y-axis refers to the usage of the information and varies from the ease of human understanding

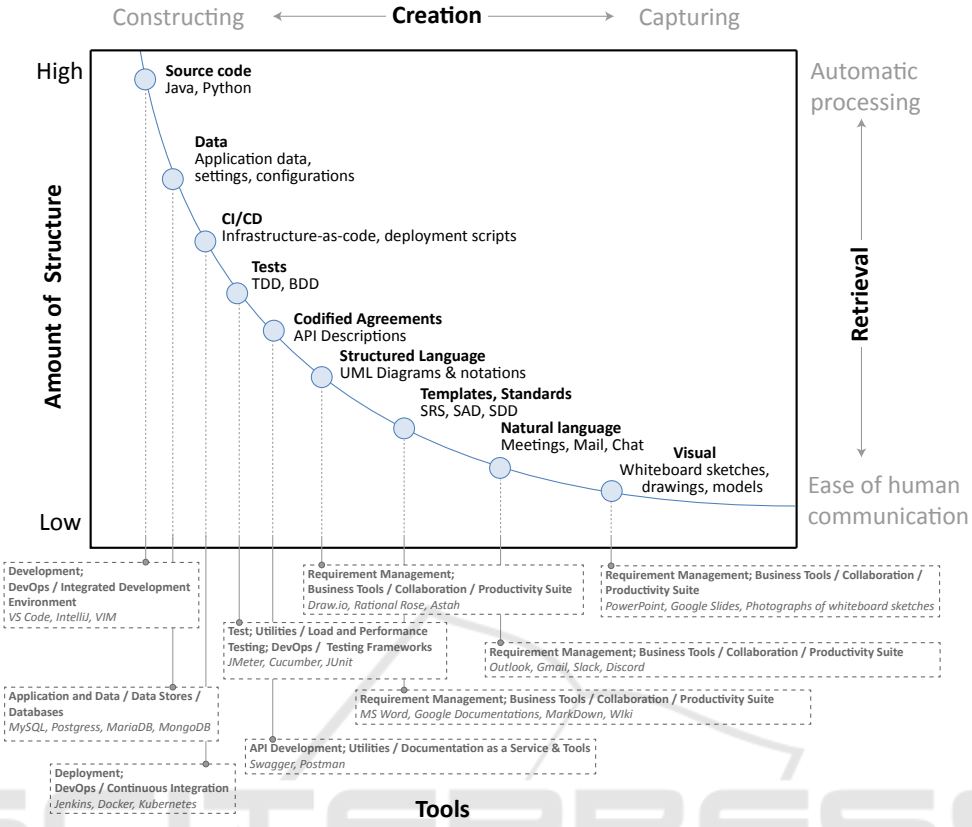

Figure 7: Amount of Structure measured by Creation, Retrieval, and Tools used to create or retrieve the types of information.

to automatic processing. “Ease of human understanding” identifies the cognitive load required to understand the information involved. “Automatic processing“ refers to the syntax, grammar, and semantics to compile source code.

“Tools are the message” concerning the variety of information relates to the amount of structural characteristics from the information that is captured, stored, or communicated. Tools enable the creation, storage, and communication of information while enforcing more or less structure.

5.3 Types of Information, Variety of Information, and Tools in a Software Development Ecosystem

This Section elaborates on RQ3: which information is stored in what tool. The data sources for this section are S5-S6 (interviews).

In previous sections, an overview was presented of types of information, and of tools including tool stacks and software development ecosystems. Also, the variety of information was discussed. All this makes clear that tools are significant in understanding the information and documentation that is created or retrieved. The main difference concerns the usage of the information stored in the tools, especially if the creators of the information are not the users of the information. This is most manifest when creator and user are not the same person. Examples are teams that are geographically distributed across the world or team members that were not involved in the conceptualization of the information.

“Tools are the message” (Section 5.1, Figure 5) in relationship with the types of information (Section 4), the variety of information (Section 5.2, Figure 7), and tools show that for creation and retrieval of information, tools make a difference.

For example, the presentation of a concept is better done in PowerPoint-like tools than source code only. The implementation of an algorithm can better be communicated with the actual source code accompanied with a model/sketch than in text. The creation and retrieval of information is closely related to these activities. Figure 5 presents the composition of “Tools are the message” by the types of information, variety of information and tools including tool stacks.

Considering major open source software repositories, most of the repositories show simple text doc-

umentation, including Mark Down. However, there is a close connection between the software product and the *format* of the information. For instance, for the Linux utility "bash", traditional UNIX man pages are used (Open Source Community, 2020a). For LaTeX, the information about the software is in tex format (Open Source Community, 2020c), and for the Firefox browser, the information is in HTML-format (Open Source Community, 2020b). These specific examples make clear that the software product and format of the information are strongly related.

5.4 How to Organize Scattered Information into Comprehensible Documentation

This section addresses RQ4. The data sources for this section are S1-S6 (all sources), as represented in Table 2. S1-S3 (open source projects) are used to gain insight in what (values, architecture, interfaces) is documented in what type of information (text, commit message, pictures) in what tool (Confluence, Github, RDBMS). S4 is used to understand the community of software practitioners. S5-S6 (case studies) are used for verification, validation and extending findings.

In CSD, "tools are the message" refers to the types of information, the variety of information and tools in software development ecosystems. The tools require or produce information with certain formats. When combining these aspects, the effort needed to comprehend the information about the software product will increase, and understanding will go down, in particular for geographically distributed teams, across buildings or across the globe, or in case of decision making not involving all team members. However, not all information about the software product is relevant for all stakeholders all the time. For customers, who pay for the development of the software product, metrics and actionable data are relevant for productivity of the software product team. For end users, a user manual should be present. If part of the end product, many websites do not have manuals but are supposed to be user friendly. For managers, metrics for relevant Key Performance Indicator (KPIs)s should be present. For developers, user stories and codified Application Programming Interface (API) descriptions should be present.

The best way to introduce new team members to the software product is definitely not to demonstrate all tools with their variety of (un)structured information, types of information and tool (stacks). A better solution is to provide stakeholders with an overview according to their specific interest. For stakeholders, this might be a PowerPoint-like presentation with the mission and vision. For managers, this might be the metrics on a KPIs dashboard. For developers, this might be the requirements, specifications, or "how to's". For end-users, this might be the user manual. Comprehensive 'yellow pages' with an overview of what is relevant for who should be available. In an investigation of the 20 major public software products, we see that the various types of information are well presented.

6 CONCLUSIONS

The phrase "tools are the message" is taken from McLuhan and Fiore (1967) "the medium is the message". He proposed investigation of the medium instead of only the message. In this paper, the tools, including the type of information, and variety, are the "medium".

Tools are the message refers to three aspects. These aspects are

1. the types of information
2. tools, including tool stacks and software development ecosystems
3. the amount of structure

Based on the research results, five conclusions are drawn. First, there is a *strong relationship between the type of information and the tool*. The type of information refers to the properties of the information in terms of what is stored (content), how it is stored (format), why it is stored (relevance), and when in the process the information is stored. Tools are the concepts, technologies, software systems, and frameworks to design, develop, deploy, and maintain the software product.

The second conclusion is that *tools are organized into tool stacks, and tool stacks are organized into software development ecosystems*. A tool stack is an organized set of tools to produce a software product. The software development ecosystem includes communities outside the team, such as product communities or knowledge communities.

The third conclusion is that the *variety of tools refers to the amount of structure for information*. This amount of structure is defined by the creation and retrieval of the information, together with the tools for creation and retrieval.

The fourth conclusion is that the *combination of these three aspects makes a difference in creating, retrieving, communicating, and understanding the message*. There is a difference in communication and

comprehension when understanding a software product's core concept through presenting with PowerPoint or through source code. The same applies when communicating codified agreements for the communication between subsystems through detailed endpoints, including input and output types, or through a whiteboard sketch.

The fifth conclusion is that the combination of these three aspects, including the core message, creates complexity concerning finding and understanding relevant information. However, not all stakeholders, including developers, require all information at any time. As a group of stakeholders, developers require information to start, continue, and deploy an iteration. This *focus on information from specific tools will decrease the complexity and make it easier to comprehend relevant information*.

REFERENCES

Bass, L., Weber, I., and Zhu, L. (2015). *DevOps: A software architect's perspective*. Addison-Wesley Professional, 1st edition.

Beshawred, Y. (2020). Open source & saas tools — stackshare. https://stackshare.io/categories. (Accessed on 07/06/2020).

Brown, S. (2014). *Software Architecture for Developers Technical leadership by coding, coaching, collaboration, architecture sketching and just enough up front design*. Number 7:1-6. Leanpub.

Cicero, M. T., Sutton, E. W., and Rackham, H. (2001). *De Oratore: Books I-II*. Number 348 in The Loeb classical library. Havard Univ. Press, Cambridge, Mass., reprinted edition.

Fowler, M. (2020). Technology radar — an opinionated guide to technology frontiers — thoughtworks. https://www.thoughtworks.com/radar. (Accessed on 10/15/2020).

Garousi, V., Felderer, M., and Mäntylä, M. V. (2019). Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. *Information and Software Technology*, 106(May):101–121.

Gartner (2020). Hype cycle research methodology. https://www.gartner.com/en/research/methodologies/gartner-hype-cycle. (Accessed on 10/15/2020).

Jansen, S., Finkelstein, A., and Brinkkemper, S. (2009). A sense of community: A research agenda for software ecosystems. *2009 31st International Conference on Software Engineering - Companion Volume, ICSE 2009*, (June):187–190.

Kersten, M. (2018). A cambrian explosion of DevOps tools. *IEEE Software*, 35(2):14–17.

Kipperman, D. (2009). Teaching through technology concepts. *Strengthening the position of technology education in the curriculum*.

Kitchenham, B. and Charters, S. (2007). Guidelines for performing Systematic Literature Reviews in Software Engineering. *Engineering*, 2(4ve):1051.

Kruchten, P. B. (1995). The 4+1 View Model of Architecture. *IEEE Software*, 12(6):42–50.

McLuhan, M. and Fiore, Q. (1967). The medium is the message. *New York Times*, 123:126–128.

NetApplications (2020). Search engine market share. https://netmarketshare.com/search-engine-market-share.aspx? (Accessed on 10/21/2020).

Open Source Community (2020a). bminor/bash: Unofficial mirror of bash repository. Updated daily. https://github.com/bminor/bash.

Open Source Community (2020b). Firefox Source Code Directory Structure — Firefox Source Docs documentation. Source code on https://github.com/mozilla/gecko-dev.

Open Source Community (2020c). latex2e/base/doc at master · latex3/latex2e. https://github.com/latex3/latex2e/tree/master/base/doc.

Pawłowski, M., Paterek, T., Kaszlikowski, D., Scarani, V., Winter, A., and Żukowski, M. (2009). Information causality as a physical principle. *Nature*, 461(7267):1101–1104.

Peirce, C. S. (1992). *The essential Peirce: selected philosophical writings*, volume 2. Indiana University Press.

Shannon, C. E. and Weaver, W. (1949). The mathematical theory of communication, 117 pp. *Urbana: University of Illinois Press*.

Theunissen, T., Van Heesch, U., and Avgeriou, P. (2020). A Mapping Study on Documentation in Continuous Software Development [Unpublished manuscript].

Wagenaar, G., Overbeek, S., Lucassen, G., Brinkkemper, S., and Schneider, K. (2018). Working software over comprehensive documentation – Rationales of agile teams for artefacts usage. *Journal of Software Engineering Research and Development*, 6(1).

Yin, R. (2002). *Case Study Research: Design and Methods, 3rd Edition (Applied Social Research Methods, Vol. 5)*. Sage Publications, Inc., third edit edition.