The Analysis of Requirements Engineering Tools for the Needs of Practitioners

Mert Ozkaya *, Geylani Kardas, Mehmet Alp Kose

Posted Date: 8 October 2023

doi: 10.20944/preprints202310.0424.v1

Keywords: requirements engineering; tools; survey; practitioners

Article

The Analysis of Requirements Engineering Tools for the Needs of Practitioners

Mert Ozkaya [1,*], Geylani Kardas [2] and Mehmet Alp Kose [3]

[1] Computer Engineering Department, Yeditepe University, Istanbul, 34755, Turkey
[2] International Computer Institute, Ege University, Izmir, 35100, Turkey
[3] Independent Researcher, Istanbul, Turkey
* Correspondence: mozkaya@cse.yeditepe.edu.tr

Abstract: Many requirements engineering tools have been existing for gathering, documenting and tracing requirements that can even be further processed for such purposes as analysis and transformation. In this study, we analysed 56 different requirements engineering tools for a comprehensive set of features that are categorised into multiple viewpoints (i.e., project management, specification, collaboration, customisation, interoperability, methodology, knowledge management). The analysis results led to many interesting findings. Some of them are as follows: (i) the project planning and execution activities are rarely supported, (ii) multi-user access and versioning are highly supported, (iii) the top-popular specification technique is natural languages, while precise specification via modeling languages is rarely supported, (iv) requirements analysis is rarely supported, (v) requirements transformation is considered for generating documents only, (vi) tool customisation via the tool integration and API support is highly popular, while customising the notation set is rarely supported, (vii) exchanging requirements is popular in such standards as ReqIF and Excel/CSV, while no single standard are accepted by all the tools, (viii agile development is very common, while other methodologies (e.g., MDE and SPLE) are rarely supported, and (viii) user-guide, telephone, e-mail, videos are the top methods for sharing knowledge. The analysis results will be useful for different stakeholders including practitioners, tool vendors and researchers.

Keywords: requirements engineering; tools; survey; practitioners

1. Introduction

A requirement is considered as a statement that needs to be agreed by all the stakeholders and contributes to solving customers' problem [1–3]. Each requirement describes either what the system to be developed is expected to perform or any constraints on the system design and development (e.g., quality and platform constraints). Requirements engineering has been proposed as the branch of software engineering, which promotes the application of well-known techniques, practices and methods for eliciting, documenting, and analysing requirements [4–7]. With requirements engineering, the goal is to maximise the quality of the requirements that will affect all subsequent stages of software development. This involves such activities as understanding the customer and user needs completely and correctly, specifying requirements precisely in a verifiable way to avoid any issues (e.g., incomplete and inconsistent requirements), and associating requirements with other artefacts (e.g., test-cases and system architectural components). It should be noted that failing to apply the activities of requirements engineering may lead to wrong systems to be developed that do not satisfy the customers and therefore never be used. Indeed, many software projects fail because of the ill-defined application of the requirements engineering practices [8–10].

To perform the requirements engineering activities effectively, several tools are available on the market through which the requirements for any systems can easily be specified and managed collaboratively throughout the project lifecycle. Requirements engineering tools offer diverse features for the practitioners including the project management facilities, requirements traceability, automated

 doi:10.20944/preprints202310.0424.v1

analysis, document generation, test-scenario generation, multi-user collaboration, and data exchange in the ReqIF standard [11], API support for customising the tool.

While tens of different alternative requirements engineering tools have been existing, the literature lacks in any resources for practitioners which can be used for determining the existing requirements engineering tools, understanding the tools' support for a number of important features for practitioners, and comparing the tools with each other. As discussed in Section 5, the existing literature studies that attempt at comparing the requirements engineering tools either ignore many of the existing tools or focus on a small set of features.

Therefore, in this paper, we aim to analyse 56 different requirements management tools for a comprehensive set of features that we believe are very important for the practitioners in industry. We categorised the features that we considered into multiple viewpoints: project management, specification, collaboration, customisation, interoperability, methodology, knowledge management. The project management viewpoint is concerned with the support for initiating, planning and executing the projects. The specification viewpoint is concerned with the technique(s) supported for specifying requirements (e.g., natural languages and modeling languages), the support for analysing requirements automatically, and the support for transforming requirements into useful artifacts (e.g., code and test scenarios). The customisation viewpoint is concerned with customising and extending the requirements engineering tools for specific user needs. The interoperability viewpoint is concerned with the support for exchanging the requirements between different tools. The methodology viewpoint is concerned with the support for any well-known techniques and methodologies that can improve the requirements engineering activities such as agile development, product-line engineering, and model-based engineering. Lastly, the knowledge management viewpoint is concerned with the support provided by the tools for reducing the learning curve.

The analysis results are believed to be highly useful for different stakeholders. Practitioners can use the analysis results to determine many of the existing requirements engineering tools, their weak and strong points in terms of the practical set of features, compare different tools and choose the one(s) that best meet their needs. Tool vendors can use the results to understand to what extent their tools satisfy the needs of practitioners and any gaps that can be improved. Lastly, using the analysis results, researchers in the field of requirements engineering can conduct further empirical studies to better understand practitioners' perspectives towards the requirements engineering tools and even propose research and development projects that could improve the requirements engineering tools from different aspects.

2. Research Methodology

2.1. Identifying and Filtering the Tools

To determine the existing requirements engineering tools, we performed a comprehensive literature search. That is, we firstly performed a *Google search* and used several keywords including "requirements engineering tools/platforms", "requirements management tools/platforms", "requirement modeling tools/platforms", "requirement tools/platforms", "requirements modeler", "SDLC tools", "modeling tools/platforms". We also searched for the web-sites that list some requirements engineering tools and those web-sites are given in Table 1. So, we managed to obtain more than 80 different requirements engineering tools in total.

Among all the requirements engineering tools determined and collected, we excluded the tools that do not have any accessible web-sites through which the knowledge about the tools can be accessed and the tools can be downloaded and installed. A few tools are supported with web-sites that are not in English and some tools' web-sites provide restricted access for non-EU countries. After eliminating all those tools, we ended up with 77 tools whose web-sites are accessible.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

Table 1. The web-pages that present a list of requirements engineering tools

Name	URL	Access Date
List of requirements engineering tools	https://en.wikipedia.org/wiki/List_of_requirements_engineering_tools	09.04.2023
7 Requirements engineering tools to make your life easy	https://www.zumvie.com/7-requirements-engineering-tools-to-make-your-life-easy/	09.04.2023
10 Best Requirements Management Tools & Software Of 2023	https://thedigitalprojectmanager.com/tools/requirements-management-tools/	09.04.2023
Top 20+ Best Requirements Management Tools	https://www.softwaretestinghelp.com/requirements-management-tools/	09.04.2023
13 BEST Requirements Management Tools & Software (2023)	https://www.guru99.com/requirement-management-tools.html	09.04.2023
Software Requirements Engineering Tools	https://ecomputernotes.com/software-engineering/softwarerequirementsengineeringtools	09.04.2023
Top Requirements Management Tools List	https://blog.testlodge.com/requirements-management-tools-list/	09.04.2023

Then, we analysed those 77 different tools to determine if we can experiment with the tools using the materials available on their web-sites. So, we excluded any tools that fail to satisfy one of those criteria at least: *(i)* the support for a free, evaluation version that can be used immediately and *(ii)* the support for a set of fully-fledged user-guide documents and any supporting materials (e.g., videos, case-studies, etc.). We observed that while a few of the tools are open-source tools, most of them are commercial that require monthly (or annual) payments to be installed and used. Also, some tools support web-based access only, some support desktop application only, and some support both. Concerning the open-source tools, we checked their web-sites to see if we can find any link for downloading and installing the executable files. For any open-source tool whose web-site does not include any valid download link, we tried to obtain the resources that can aid in understanding the tool's support for the features of interest (e.g., guides, videos, tutorials, white papers, etc.). In the cases where we could not download any executable files or access adequate level of resources, we eliminated those open-source tools. Concerning the commercial tools, we checked their web-sites for a free, evaluation version. Some of the tools' web-site do not provide such kind of access. So, we tried to use their resources available (if any) to understand the tools' support for the features of interest. We eliminated any tools whose free, evaluation version cannot be accessible and resources are inadequate.

So, we ended up with a list of 56 different tools given in Table 2, which we can analyse for the features of interest.

Table 2. The requirements engineering tools

Tool	Web-site	Supported platforms	Open-source	Year
Accompa PM	www.web.accompa.com	Web	No	2007
acunote	www.acunote.com/	Web	No	2006
Agile Requirements Designer	www.broadcom.com/products/software/continuous-testing/agile-requirements-designer	Web and On-premise	No	1960s
agosense.fidelia	www.agosense.com/	Web and On-premise	No	2009
Aha!	www.aha.io/	Web	No	2013
Aligned Elements	www.aligned.ch/features/requirement-management	Web and On-premise	No	2006
Quality Center - Dimensions RM	www.microfocus.com/en-us/products/dimensions-rm/overview	Web	No	1976
Auros IQ	www.aurosks.com/	Web	No	2001
Axosoft	www.axosoft.com/	Web and On-premise	No	2014
Azure DevOps	www.azure.microsoft.com/	Web and On-premise	No	2005
Balsamiq Wireframes	www.balsamiq.com/	Web and On-premise	Yes	2008
Business Optix	www.businessoptix.com/	Web	No	2010
Cameo Systems Modeler	www.3ds.com/products-services/catia/products/no-magic/cameo-systems-modeler/	Web and On-premise	No	1998

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

Table 2. *Cont.*

Tool	Web-site	Supported platforms	Open-source	Year
Capella	www.eclipse.org/capella/	On-premise	Yes	2001
CaseComplete	www.casecomplete.com/	Web	No	2012
ClickUp	www.clickup.com/	Web and On-premise	No	2017
CodeBeamer ALM	www.codebeamer.com/	Web and On-premise	No	1998
Cradle	www.threesl.com/cradle/	Web and On-premise	No	1987
Doc Sheets	www.docsheets.com	Web and On-premise	No	2000
Eclipse Papyrus	www.eclipse.org/papyrus	On-premise	Yes	2019
Enterprise Architect	www.sparxsystems.com/	On-premise	No	2000
Helix RM	www.perforce.com/products/helix-alm	Web and On-premise	No	1995
innoslate	www.innoslate.com/	Web and On-premise	No	1993
Innovator for Business Analysts	www.innovator.de/en/	On-premise	No	1986
in-STEP BLUE	www.microtool.de/en/products/in-step-blue/	Web and On-premise	No	1997
iRise	www.irise.com/	Web and On-premise	No	2016
Jama Connect	www.go.jamasoftware.com/	Web and On-premise	No	2007
Kovair ALM	www.kovair.com	Web	No	2006
MagicDraw	www.3ds.com/products-services/catia/products/no-magic/magicdraw/	On-premise	No	1981
Matrix ALM/QMS	www.matrixreq.com/en/product	Web	No	2013
Modelio Analyst	www.modeliosoft.com/en/modules/analyst.html	Web and On-premise	No	2009
OpenProject	www.openproject.org/	Web and On-premise	No	2012
Orcanos	www.orcanos.com	Web and On-premise	No	2004
PivotalTracker	www.pivotaltracker.com/	Web	No	2006
Polarion Requirements	www.polarion.plm.automation.siemens.com/products/polarion-requirements	Web	No	2004
Psoda	www.psoda.com/	Web	No	2006
Rational DOORS	www.ibm.com/docs/en/ermd/	Web and On-premise	No	1993
Rational Rhapsody	www.ibm.com/products/uml-tools	On-premise	No	2011
ReqEdit	www.reqteam.com/	On-premise	No	2014
ReQtest	www.reqtest.com/	Web	No	2009
ReqView	www.reqview.com/	Web and On-premise	No	2015
RMsis	www.marketplace.atlassian.com/apps/30899/rmsis-requirements-management-for-jira	Web	No	2009
ReqChecker	https://reqchecker.eu/	On-premise	No	2016
RMTrack	www.rmtrack.com/	Web	No	2001
Scrumwise	www.scrumwise.com/	Web	No	2009
SpiraTeam	www.inflectra.com/SpiraTeam/	Web and On-premise	No	2006
StoriesOnBoard	https://storiesonboard.com/	Web	No	2015
SwiftKanban	www.nimblework.com/	Web and On-premise	No	1998
Targetprocess	www.targetprocess.com/	Web and On-premise	No	2006
TopTeam	www.topteamrequirements.com	Web and On-premise	No	1995
Tuleap Enterprise	www.tuleap.org	Web and On-premise	No	2011
Valispace	www.docs.valispace.com/	Web and On-premise	No	2016
Visual Paradigm	www.visual-paradigm.com	Web and On-premise	No	2002
Visure Requirements	www.visuresolutions.com/	Web	No	2002
Yodiz	www.yodiz.com	Web	No	2009
Xebrio	www.xebrio.com	Web and On-premise	No	2018

2.2. Identifying the Tool Features

To determine the features for which the requirements engineering tools shown in Table 2 are analysed, we considered the needs of practitioners who are involved in the software and systems development. Therefore, we used the results and feedback obtained from our previous survey on understanding practitioners' experiences in requirements engineering [12]. We also used our past expertise on analysing and comparing modeling languages and tools for the needs of practitioners

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

e.g., [13]. Therefore, we end up with several important features that can be used for analysing and comparing the existing requirements engineering tools. To facilitate the understandability, we categorised the features into multiple viewpoints, which are project management, collaboration, requirements specification, customisation, interoperability, methodology, and knowledge management.

2.2.1. Project Management

With the project management viewpoint, we focus on three project management stages, which are project initiation, project planning, and project execution [14]. Project initiation can be considered as the first stage of project management and concerned with the activities that are performed once the project is approved including defining the problem, solution, and scope in detail, assigning project manager and creating teams, and organising a physical area. Project initiation is followed by the project planning which is concerned with drawing a gantt chart for planning the tasks and their dependencies, and other issues such as resource planing, budget planning, staff planning, and risk planning. Project execution is concerned with the stage where the project plans are realised and any progresses are reported.

2.2.2. Collaboration

With the collaboration viewpoint, we focus on the capability of multiple users using the same tool with different access rights collaboratively. We consider the collaboration support in terms of multi-user access, user role definitions, user access-right definitions, and requirements versioning. The multi-user access support enables multiple users to access the same project and manage the requirements together at the same time. The support for the user roles enables for defining the roles of the users such as customer, analyst, system engineer, developer, manager, etc. The support for the user access-rights enables for defining the access-rights for the users such as read-only access, edit access, authorisations, etc. The support for versioning enables for keeping different versions of the requirements in a repository that can be accessed, compared, and even deleted by the users.

2.2.3. Requirements Specification

With the requirements specification viewpoint, we are concerned with the specification of requirements, the analysis of requirements specifications, and the transformation of requirements specifications into some useful artefacts such as code, test scenarios and documentation.

Concerning the requirement specifications, we consider three important techniques suggested by Taylor et al. too [15] - natural languages, boxes-and-lines, and modeling languages. Natural languages and boxes-and-lines represent the informal techniques for specifying requirements and promote any stakeholders with no technical knowledge to be involved in the requirements specification process. However, informal requirement specifications can be ambiguous and thus interpreted differently by different stakeholders due to lacking in precise definitions behind (i.e., syntax and semantics).

An alternative way for specifying requirements is modeling languages. Modeling languages can be supported with precise (and sometimes formal) definitions (i.e., syntax and semantics) and thus enable for the precise communications of requirements among stakeholders. Also, with some modeling tool support, it can be easier to analyse the requirements specifications and even generate executable code and test scenarios. Modeling languages can be general-purpose (e.g., UML [16], SysML [17], and BPMN [18]) or domain-specific (e.g., AADL for embedded systems [19]). Modeling languages are supported by several tools (e.g., UML tools [13] and AADL's OSATE[1]), which can aid in specifying requirements, analysing the specifications and transforming them into code.

---

[1] https://osate.org/

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

Concerning the requirements analysis, we consider the tool support for a pre-defined set of properties for which the requirement specifications can be analysed *(i)* defining new (i.e., user-defined) properties for which the requirements specifications can be analysed, *(ii)* simulating (i.e., executing) the requirement specifications, and *(iii)* completeness and consistency checks [20]. The requirement consistency here can be used for checking if two requirements conflict with each other (e.g., implying the same functionality), while the requirement completeness can be for checking if all the requirements are specified and each requirement is specified with no missing information.

Concerning the requirements transformation, we consider the support for generating skeleton code from requirements, generating test scenarios from requirements, and generating documentations in different formats (e.g., HTML, PDF, Excel, Word, etc.).

2.2.4. Customisation

With the customisation viewpoint, we are concerned with the capability of extending the requirements engineering tools with some additional features. We consider customisation in terms of user-defined modeling viewpoints, integration with external tools, API support, and DSL support. With the user-defined modeling viewpoint definition support, users use the modeling elements supported by the requirements engineering tool and define a viewpoint in terms of the rules and constraints specific to their concern. Then, users can use their viewpoint definition to specify domain-specific requirements [21]. With the integration support, users can integrate the requirements engineering tools with some external tools exhibiting different capabilities such as test automation, design, simulation, verification, project management, versioning, and repository management. With the API support, users can extend the requirements engineering tools by developing new functionalities. With the DSL support, users can develop their own domain-specific modeling languages for specifying requirement models in terms of domain-specific concepts, relationships and their symbols. Indeed, some tools can enable the use of UML profiling mechanism for extending UML for domain-specific purposes [22].

2.2.5. Interoperability

With the interoperability viewpoint, we are concerned with the tools' capabilities for accepting requirements documented in different formats (e.g., MS Word, MS Excel, CSV, XMI, and XML.) and exchanging the requirements with other tools and technologies using some well-accepted standards (e.g., the ReqIF standard).

2.2.6. Methodology

With the methodology viewpoint, we are concerned with the tools' support for the important software development methodologies that facilitate the software development and therefore aid in better managing the requirements engineering processes. We consider the agile software development, software product-line engineering, and model-driven software engineering methodologies, as we strongly believe that these three methodologies have great importance for developing quality systems and can be integrated with each other for better solutions [23–25]. Agile methodology promotes the customers to be always involved in the requirements engineering processes where any change requests on the requirements are always accepted and the requirements are analysed through working prototypes [26]. Agile development is supported with such techniques as Scrum, Kanban, lean management, through which the agile principles can be applied effectively. Software product-line engineering (SPLE) is another methodology that promotes the development of software products which have commonalities (e.g., the same functionalities) [27]. SPLE focuses on defining a product line architecture by determining the commonalities and variabilities of the artifacts generated in different stages of software development including artifacts in the requirement stage (e.g., use-case models, interaction and behaviour models). Whenever a product needs to developed, the variants of the product line architecture need to be reused and associated with the concrete needs and requirements of the product. Model-driven engineering (MDE) promotes specifying abstract models (e.g., requirements

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

models) for different perspectives of system development (e.g., structure, interaction, and behaviour) that can further be processed via tool support (e.g., model simulation, code generation, document generation, and versioning) [28].

2.2.7. Knowledge Management

With the knowledge management viewpoint, we are concerned with the knowledge that the requirements engineering tools share via their web-sites so as to reduce the time for learning and using the tools. We consider different methods for sharing the knowledge, including telephone, e-mail, forum, live-chat, help-desk, user-guide, blog, white papers, mailing lists, case-studies, videos, training, and coaching/consulting.

2.3. Collecting and Analysing the Data

To collect data, we used the MS Excel office tool and created separate sheets for different viewpoints introduced in Section 2.2 for which the requirements engineering tools analysed. For each tool, we used the tool itself (if we managed to download) and benefited any helper documents (e.g., web-site, user-guides, tutorials, white papers, blogs, and videos) so as to collect data for each feature of each viewpoint considered. We stored the collected data in the appropriate sheets of the Excel file. Note that to maximise the correctness and completeness of the data collected, each author performed the above process individually and then the collected data by the authors have been compared to detect any discrepancies. So, we ended up with a single Excel file that includes the tool data which have been justified and confirmed by each author.

After collecting data, we focussed on analysing the collected data so as to obtain some important findings about the tools. To this end, created a table for each viewpoint using the MS Excel office tool, and thus the collected data for the tools that support the features of the viewpoint in question are displayed. Also, we drew some charts to indicate some important findings about the tools' support for the viewpoint features. Note that during the data analysis process, we had also the need for re-collecting the data about some tools as we determined some ambiguities and inconsistencies.

3. The Analysis of the Requirements Engineering Tools

In this section, we discuss the analysis results of the 56 different requirements engineering tools given in Table 2. We consider each viewpoint introduced in Section 2.2 and discuss the tool support for the viewpoint features.

3.1. Project Management

As shown in Figure 1, 46% of the requirements engineering tools support all the features of project management considered (i.e., project initiation, project planning, and project execution). 38% of the tools just support the initiation phase of the projects, which includes such activities as creating projects and assigning members to the projects. However, those tools do not support planning the project tasks with, e.g., Gantt chart, and the execution stage. 16% of the tools do not provide built-in support for project management. Note that those tools with no built-in support may provide the integration with project management tools such as Jira[2].

3.2. Collaboration

All the requirements engineering tools enable the multiple users to work together on the same project. The only exception here is ReqChecker. 78% of the tools analysed further support assigning

[2] https://www.atlassian.com/software/jira

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

roles to the users and giving them appropriate access rights. Note that the tools that have been observed to ignore the user role and access-right definitions are given in Table 3.

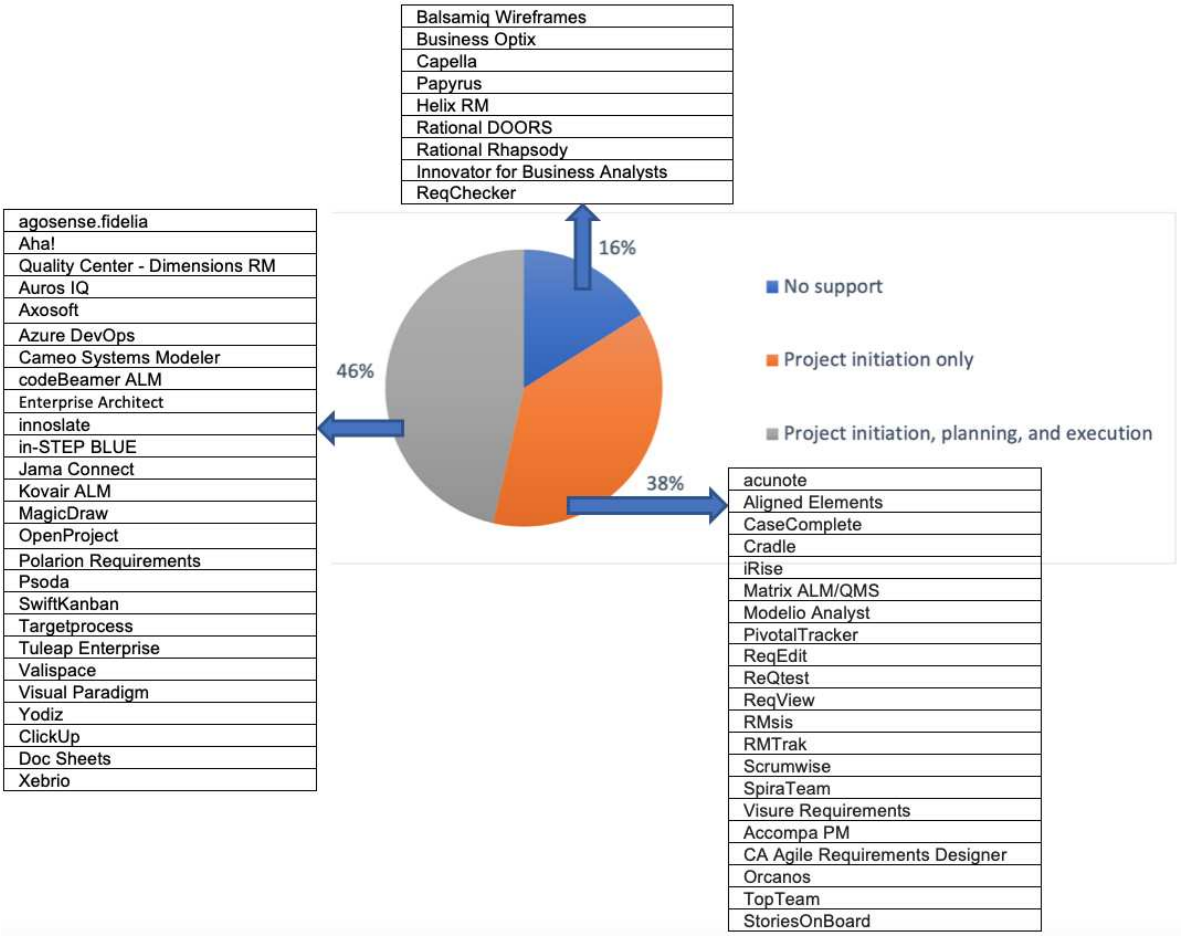

Figure 1. The support for the project management

Table 3. The requirements engineering tools that do not support either user roles or user-access rights

Requirements Engineering Tools	Multi-user Access	User Roles	User-access Rights
acunote	Yes	No	Yes
agosense.fidelia	Yes	No	No
Balsamiq Wireframes	Yes	No	No
Business Optix	Yes	No	No
CaseComplete	Yes	No	No
Eclipse (IDE) Papyrus	Yes	No	Yes
Psoda	Yes	Yes	No
ReqView	Yes	No	No
Valispace	Yes	No	Yes
Orcanos	Yes	No	No
Xebrio	Yes	No	Yes

Almost all the requirements engineering tools (93%) support versioning - ScrumWise, RMTrack, and Psoda are the only exceptions. Table 4 shows the requirements engineering tools' versioning support in terms of the support for built-in central versioning system and the integration with external versioning systems (i.e., GIT, SVN, and Mercurial). So apparently, 74% of the tools supporting versioning provide built-in versioning systems for versioning requirements. The support for the integration with external versioning systems is quite rare (3-25%) - GIT is supported by 25% of the tools, SVN is supported by 20% of the tools, and Mercurial is supported by just 3%. Acunote is the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

tool that supports all external versioning systems considered - GIT, SVN, and mercurial. Polarion Requirements is the tool that provides an internal versioning system and supports the integration with GIT and SVN at the same time.

Table 4. The requirements engineering tools that support versioning

Requirements Engineering Tools	Built-in Versioning	GIT	SVN	Mercurial
acunote	No	Yes	Yes	Yes
agosense.fidelia	No	Yes	No	No
Aligned Elements	Yes	No	No	No
Auros IQ	Yes	No	No	No
Axosoft	No	Yes	No	No
Azure DevOps	No	Yes	No	No
Balsamiq Wireframes	Yes	No	No	No
Business Optix	Yes	No	No	No
Cameo Systems Modeler	Yes	No	No	No
Capella	No	Yes	Yes	No
CaseComplete	No	No	Yes	No
codeBeamer ALM	No	Yes	Yes	Yes
Cradle	Yes	No	No	No
Eclipse (IDE) Papyrus	Yes	Yes	No	No
Enterprise Architect	Yes	No	Yes	No
Helix RM	Yes	No	No	No
innoslate	Yes	No	No	No
Innovator for Business Analysts	Yes	No	No	No
in-STEP BLUE	Yes	No	No	No
iRise	Yes	No	No	No
Jama Connect	Yes	No	No	No
Kovair ALM	Yes	No	No	No
MagicDraw	Yes	No	No	No
Matrix ALM/QMS	Yes	No	No	No
Modelio Analyst	Yes	No	No	No
OpenProject	Yes	No	No	No
PivotalTracker	Yes	No	No	No
Polarion Requirements	Yes	Yes	Yes	No
Quality Center - Dimensions RM	Yes	No	No	No
Rational DOORS	Yes	No	No	No
Rational Rhapsody	Yes	No	No	No
ReqChecker	Yes	No	Yes	No
ReqEdit	Yes	No	No	No
ReQtest	Yes	No	No	No
ReqView	Yes	No	Yes	No
RMsis	Yes	No	No	No
SpiraTeam	Yes	No	Yes	No
StoriesOnBoard	No	Yes	No	No
SwiftKanban	No	Yes	No	No
Targetprocess	No	Yes	No	No
Tuleap Enterprise	No	Yes	Yes	No
Valispace	Yes	No	No	No
Visual Paradigm	Yes	No	No	No
Visure Requirements	Yes	No	No	No
Yodiz	No	Yes	Yes	No
Accompa	Yes	No	No	No
CA Agile Requirements Designer	Yes	No	No	No
ClickUp	No	Yes	No	No
Doc Sheets	Yes	No	No	No
Orcanos	Yes	No	No	No
TopTeam	Yes	No	No	No
Xebrio	Yes	No	No	No

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

3.3. Requirements Specification

As Figure 2 shows, most of the languages (73%) support the informal specifications of requirements using natural languages (e.g., English sentences). A few of those tools (Dimensions RM, Auros IQ, Balsamiq Wireframes, Helix RM, and Jama Connect) supplement the natural languages with simple boxes-and-lines for the requirement specifications.

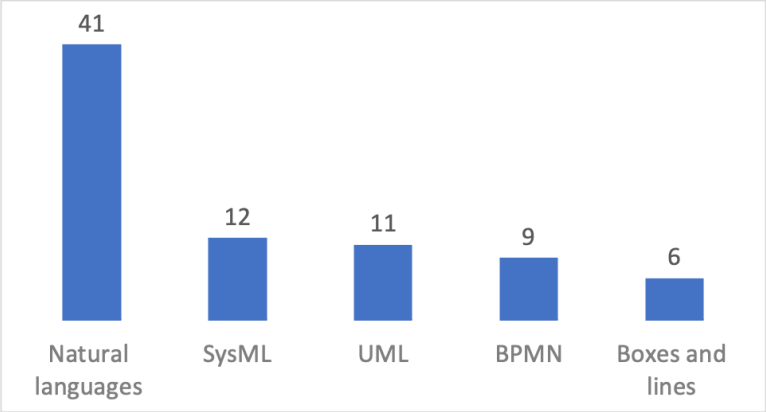

Figure 2. The requirements specification techniques supported by the requirements engineering tools

Some of the requirements engineering tools given in Table 5 support the precise specifications of requirements using well-known modeling languages including SysML, UML, and BPMN.

Table 5. The requirements engineering tools that support modeling languages for the requirements specification

Requirements Engineering Tools	Modeling Languages
Business Optix	BPMN
Cameo Systems Modeler	SysML
Capella	SysML
CaseComplete	UML
Cradle	SysML
Eclipse (IDE) Papyrus	UML, SySML
Rational Rhapsody	UML, SySML
innoslate	LML, SysML
Innovator for Business Analysts	ArchiMate, BPMN, SysML
in-STEP BLUE	UML, SySML, and natural languages
MagicDraw	UML, SysML, BPMN, OWL, OCL, MARTE, SOAML
Visual Paradigm	UML, BPMN, ArchiMate, DFD, ERD, SoaML, SysML, CMMN
CA Agile Requirements Designer	Flowchart
TopTeam	UML, SysML, BPMN

Table 6 shows the requirements engineering tools (36%) that support the analysis of the requirement specifications. Most of those tools that support the requirements analysis require practitioners to use modeling languages that lead to the precise requirement specifications for facilitating the analysability. A few of the tools that support the requirement specifications in natural languages (i.e., innoslate, ReqView, SwiftKanban, and Visure Requirements) use artificial intelligence and natural language processing technologies for checking the informal requirements in natural languages (e.g., determining ambiguous requirements and similar requirements).

Checking requirements for pre-defined properties is the most popular feature, satisfied by many of the tools. Also, some of the tools that support pre-defined properties further enable users to define their own properties. Model simulation support is provided by the tools that essentially enable for the precise specifications of the behaviour requirements via some modeling languages (e.g., UML, SysML

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

and BPMN). The completeness and consistency checks for the requirements are also quite popular. Lastly, Cameo Systems Modeler is the only tool that supports all the analysis properties considered.

Table 6. The requirements engineering tools that support the requirements analysis

Requirements Engineering Tools	Pre-defined properties	User-defined properties	Simulation	Consistency	Completeness
agosense.fidelia	X				
Auros IQ	X				X
Business Optix			X		X
Cameo Systems Modeler	X	X	X	X	X
Capella	X	X	X		X
Cradle	X	X		X	X
Eclipse (IDE) Papyrus			X		
Enterprise Architect	X	X	X		
Rational DOORS					X
Rational Rhapsody	X	X	X		
innoslate	X		X	X	X
Innovator for Business Analysts	X				
MagicDraw	X	X	X		X
ReqEdit	X				
ReqView				X	
SwiftKanban				X	
Valispace				X	X
Visual Paradigm			X		
Visure Requirements	X			X	
ReqChecker	X			X	X

Table 7 shows the requirements engineering tools (75%) that support the requirements transformation. So apparently, almost all those tools support generating documents from requirement specifications in different formats such as Word, Excel, PDF, and HTML. Code generation from requirement specifications is supported by a few tools only, which are Cameo, Papyrus, Enterprise Architect, MagicDraw, Modelio, and Visual Paradigm. Note that those tools that support code generation are all software modeling and design tools that generate code from precise models specified with modeling languages (e.g., SysML and UML). Test scenario generation from the requirement specifications is supported by a small amount of tools, which are Quality Center - Dimensions RM, Auros IQ, RMsis, SpiraTeam, TopTeam, and CA Agile Requirements Designers.

3.4. Customisation

Customisation is considered in terms of the support for user-defined modeling viewpoints, integration with external tools, API support, and DSL support.

Concerning the user-defined modeling viewpoint, Cameo Systems Modeler, Enterprise Architect, Rational Rhapsody, MagicDraw, Modelio Analyst, and Visual Paradigm are the only tools that enable the users to define their own custom modeling viewpoints for specifying requirements. With those tools, users can re-use and extend the existing concepts from popular modeling languages (e.g., UML and SysML) and define rules and constraints on those concepts. By doing so, users can later use the viewpoints for addressing the particular needs and concerns.

Concerning the integration with external tools, most of the requirements engineering tools (87%) support the integration with different types of tools including test automation tools, project management tools, versioning and repository tools. The exceptions here are Modelio Analyst, OpenProject, ReqEdit, ReqView, RMTrack, CA Agile Requirements Designer, and TopTeam.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

Table 7. The requirements engineering tools that support the requirements transformation

Requirements Engineering Tools	Code Generation	Test Scenario Generation	Document Generation
Aligned Elements			X
Quality Center-Dimensions RM		X	
Auros IQ		X	
Axosoft			X
Azure DevOps			X
Cameo Systems Modeler	X		
Capella			X
CaseComplete			X
codeBeamer ALM			X
Cradle			X
Eclipse (IDE) Papyrus	X		X
Enterprise Architect	X		X
Helix RM			X
ational DOORS			X
innoslate			X
n-STEP BLUE			X
iRise			X
Kovair ALM			X
MagicDraw	X		X
Matrix ALM/QMS			X
Modelio Analyst	X		X
penProject			X
Polarion Requirements			X
Psoda			X
ReqEdit			X
ReQtest			X
ReqView			X
RMsis		X	X
SpiraTeam		X	X
Tuleap Enterprise			X
Valispace			X
Visual Paradigm	X		X
Visure Requirements			X
Yodiz			X
Accompa PM			X
CA Agile Requirements Designer		X	X
Doc Sheets			X
Orcanos			X
TopTeam		X	X
Xebrio			X
ReqChecker			X
StoriesOnBoard			X

Concerning the API support, 70% of the requirements engineering tools considered offer an API for the users to extend the tools with specific features. Those tools provide web-sites that guide on how to use the API and perform any extensions. Figure 3 shows the tools that offer APIs and those that do not.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

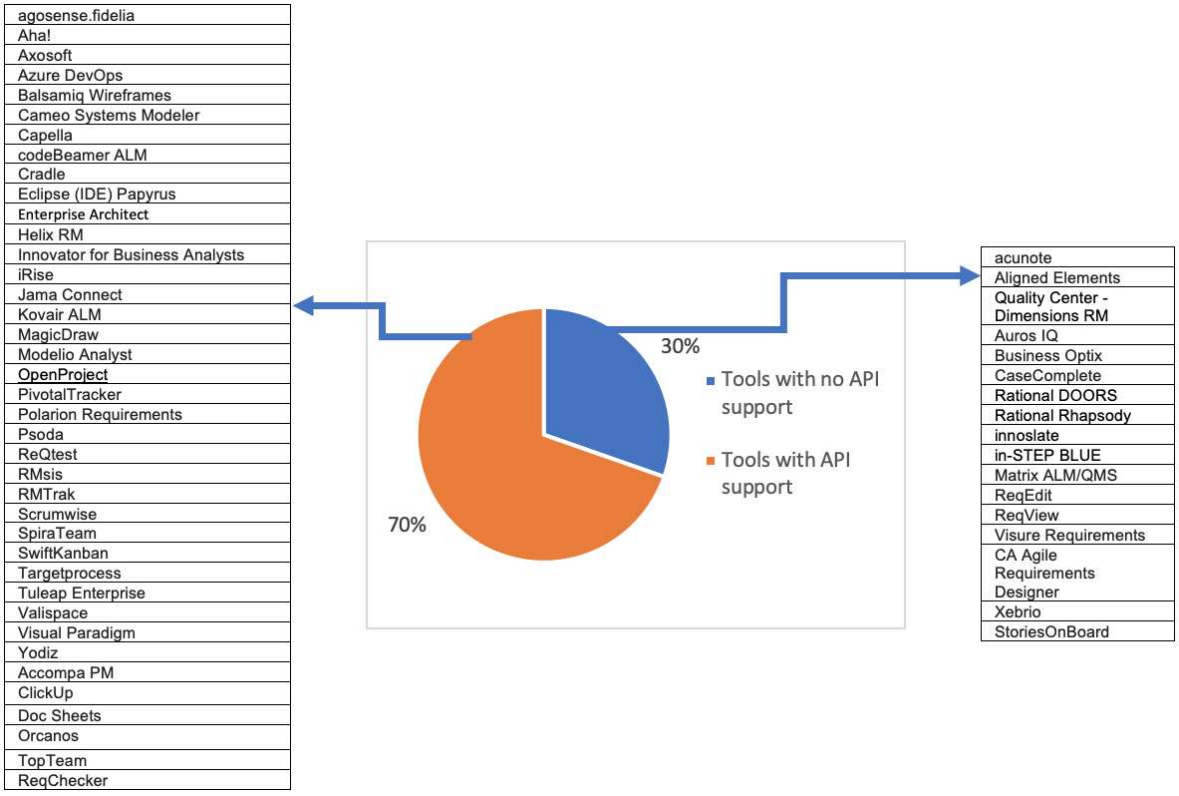

Figure 3. The requirements engineering tools that support API for enabling extension

Concerning the DSL development support, most of the requirements engineering tools do not support the development of domain-specific languages for specifying requirements. The only exceptions here are some of the tools that support UML - Cameo Systems Modeler, Eclipse Papyrus, Enterprise Architect, Rational Rhapsody, MagicDraw, Modelio Analyst, and Visual Paradigm. Those tools that support UML modeling enable users to benefit UML's profiling mechanism for extending UML with domain-specific concepts.

3.5. Interoperability

Interoperability is considered in terms of the support for accepting requirements in different data formats and exchanging requirements with other tools.

Most of the requirements engineering tools considered (89%) accept requirements in different data formats. The top-supported data formats are Excel (32%), CSV (31%), and Word (20%), which are followed by other formats including XML, XMI, Json, and PDF. Figure 4 shows the tools that support the different data formats for accepting requirements.

77% of the requirements engineering tools enable exchanging requirements specifications (i.e., importing and exporting) in some data formats. The top-supported data format is ReqIF (25%), which is followed by Excel (23%) and CSV (23%). Figure 5 shows the tools that support different data formats for the requirements exchange.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

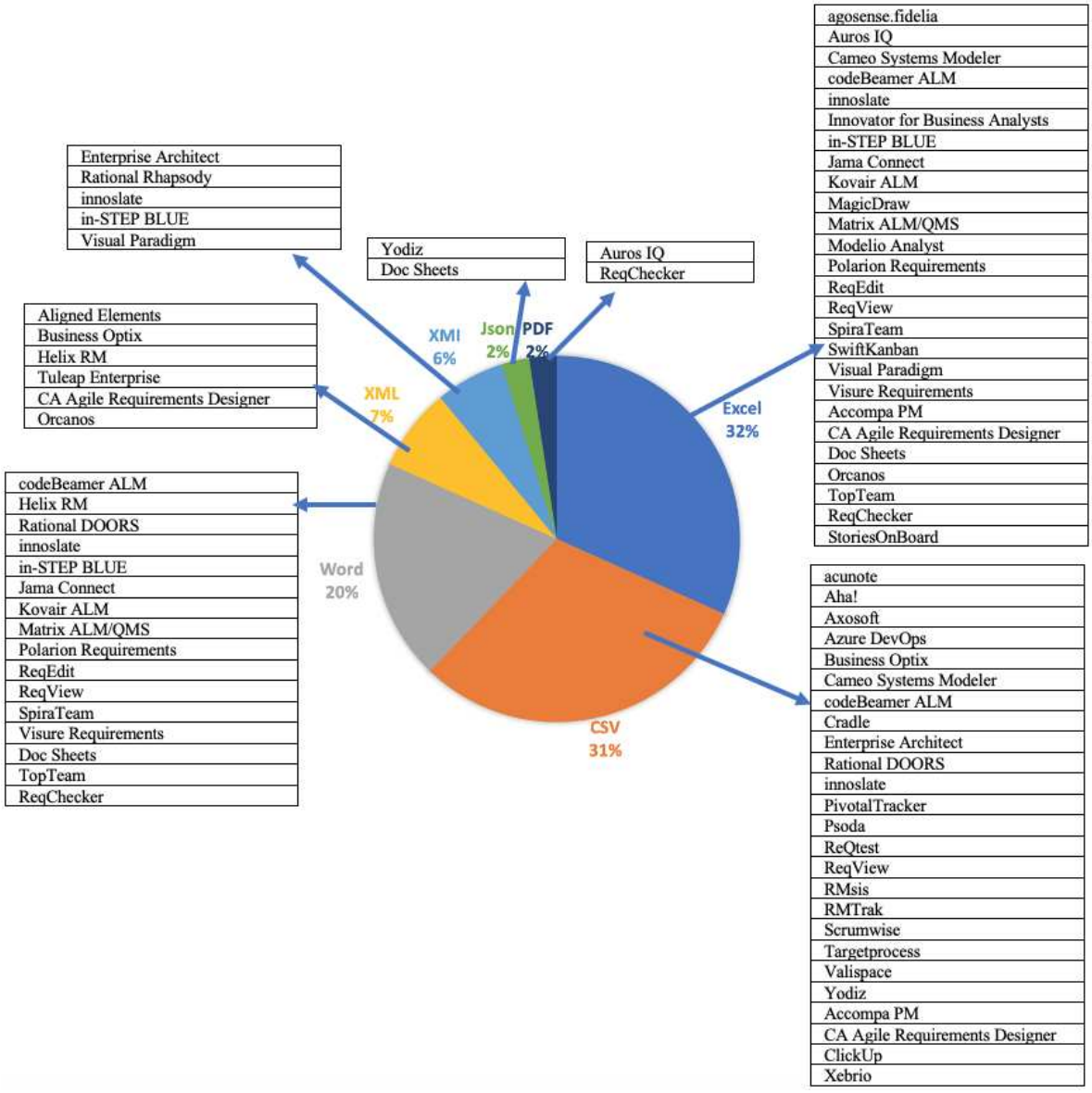

Figure 4. The requirements engineering tools that accept requirements in different formats

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

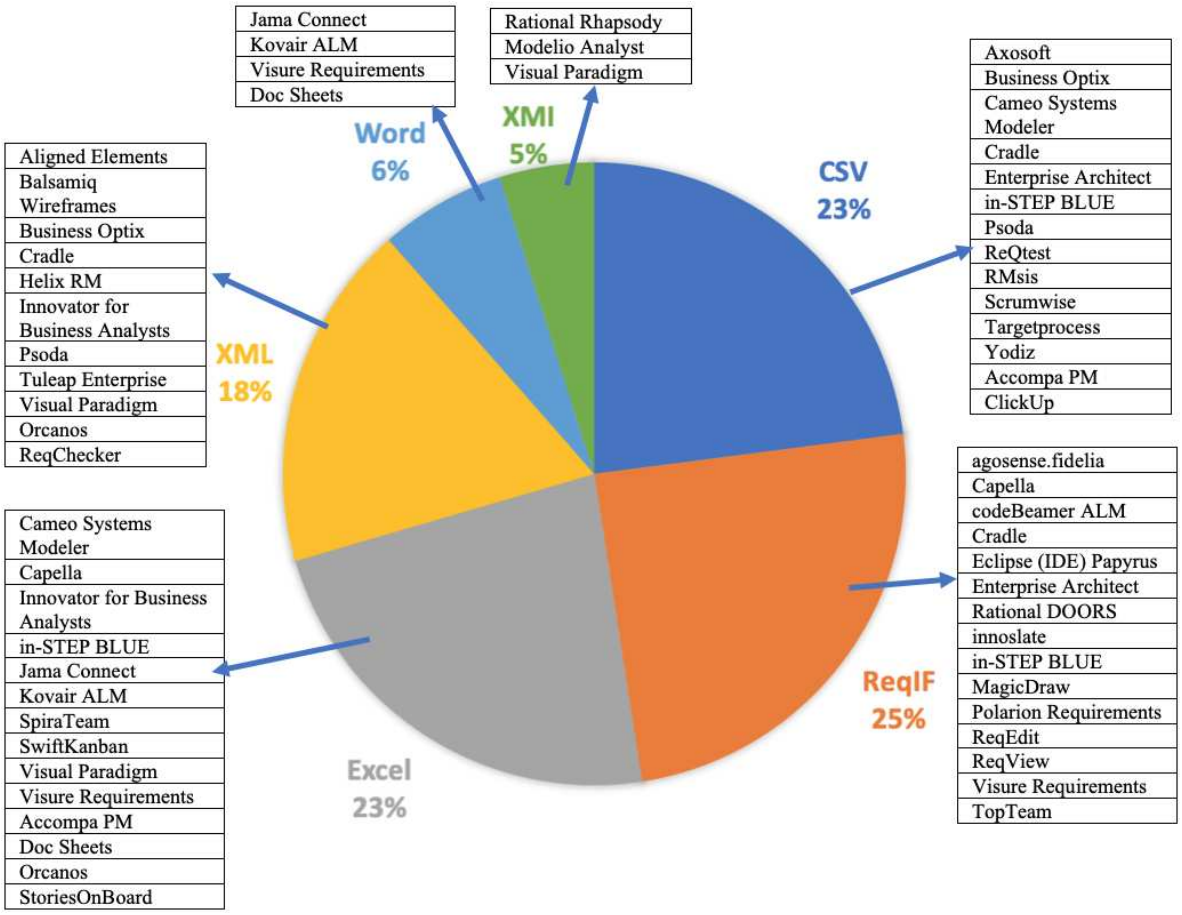

Figure 5. The requirements engineering tools that support exchanging requirements in different formats

3.6. Methodology

We consider three important software development methodologies, which are model-driven engineering, agile software development, and product-line engineering. 70% of the requirements engineering tools considered support at least one of those three methodologies and those tools are given in Table 8. Agile software development is highly popular among the requirements engineering tools (52%), which enable the management of requirements using the agile development principles. 32% of the tools support model-driven engineering and enable the specifications of models using modeling languages and performing other facilities such as model validation, model simulation, and model transformation (e.g., generating code from models). Product-line engineering is rarely supported in comparison with agile software development and model-driven engineering - Cameo Systems Modeler, codeBeamer ALM, and MagicDraw are the only tools that support the specifications of software systems with the principles of product-line engineering. Note that none of the requirements engineering tools support all the three methodologies at the same time.

3.7. Knowledge Management

We consider the knowledge management in terms of the requirements engineering tools' support for telephone communication, e-mail communication, forum, live-chat, help-desk, user guide, blog, white papers, mailing list, case-studies, videos, training, and coaching/consulting. Table 9 shows the support provided by the requirements engineering tools. So apparently, each tool enables users to access any tool-related knowledge to some extent. Indeed, each tool provides a user-guide for the users to learn how to use the tool. Likewise, most of the tools (73-77%) provide telephone details and e-mail addresses through which users may ask their questions. 73% of the tools provide videos for the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

users to see and learn how to use the tool. Help-desk, training support, blog resources are also quite popular among the tools (50-57%). However, live-chat, mailing list and coaching are rarely supported (7-27%).

Enterprise Architect, Rational Rhapsody, Targetprocess, and Visual Paradigm are the tools that support the greatest number of methods for sharing knowledge. Note that none of the tools support all the features considered for the knowledge management.

Table 8. The requirements engineering tools that support different methodologies

Requirements Engineering Tools	Model-driven Engineering	Agile Software Development	Product-line Engineering
acunote	No	Yes	No
agosense.fidelia	No	Yes	No
Aha!	No	Yes	No
Quality Center - Dimensions RM	Model-driven testing	Yes	No
Auros IQ	No	Yes	No
Axosoft	No	Yes	No
Azure DevOps	No	Yes	No
Business Optix	Model specification and simulation	Yes	No
Cameo Systems Modeler	Model specification and validation	No	Yes
Capella	Model specification and validation	No	No
CaseComplete	Model specification	No	No
codeBeamer ALM	No	Yes	Yes
Cradle	Model specification and validation	Yes	No
Eclipse (IDE) Papyrus	Model specification and simulation	No	No
Enterprise Architect	Model specification, validation, and transformation	Yes	No
ational Rhapsody	Model specification, validation, simulation, and transformation	No	No
innoslate	Model specification and simulation	No	No
Innovator for Business Analysts	Model specification	No	No
iRise	Model specification	Yes	No
Kovair ALM	Model specification	Yes	No
MagicDraw	Model specification, validation, and transformation	No	Yes
Modelio Analyst	Model specification, validation, and transformation	No	No
penProject	No	Yes	No
PivotalTracker	No	Yes	No
Psoda	No	Yes	No
ReQtest	No	Yes	No
RMsis	No	Yes	No
Scrumwise	No	Yes	No
SpiraTeam	No	Yes	No
SwiftKanban	No	Yes	No
Targetprocess	No	Yes	No
Tuleap Enterprise	No	Yes	No
Visual Paradigm	Model specification, validation, simulation and transformation	Yes	No
Visure Requirements	No	Yes	No
Yodiz	No	Yes	No
Accompa PM	No	Yes	No
CA Agile Requirements Designer	Model-driven testing	No	No
ClickUp	No	Yes	No
Doc Sheets	No	Yes	No
TopTeam	Model-driven testing	No	No

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

Table 9. The requirements engineering tools that support knowledge management

Requirements Engineering Tools	Tel.	E-mail	Forum	Live-chat	Help Desk	Guide	Blog	White paper	M. List	Case-study	Videos	Training	Coaching
acunote	X	X				X	X		X				
agosense.fidelia	X	X			X	X	X						
Aha!	X	X		X		X	X						
Aligned Elements	X	X				X	X						
Quality Center - Dimensions RM	X	X	X		X	X	X	X	X			X	
Auros IQ	X	X				X	X						
Axosoft		X				X	X						
Azure DevOps			X			X	X						
Balsamiq Wireframes	X	X	X		X	X		X			X		
Business Optix	X	X			X	X		X		X	X	X	
Cameo Systems Modeler	X		X		X	X				X	X		X
Capella			X			X		X	X	X	X	X	X
CaseComplete	X	X			X	X					X		
codeBeamer ALM		X				X	X	X		X	X	X	X
Cradle	X	X				X	X	X			X	X	
Eclipse (IDE) Papyrus		X	X		X	X			X	X	X		
Enterprise Architect	X	X	X		X	X		X		X	X	X	X
Helix RM	X	X				X				X	X	X	X
Rational DOORS	X		X		X	X	X	X		X	X	X	X
Rational Rhapsody	X		X		X	X	X	X		X	X	X	X
innoslate	X	X		X	X	X	X				X	X	
Innovator for Business Analysts	X	X				X					X	X	
in-STEP BLUE	X	X				X	X	X			X		
iRise						X		X			X		
Jama Connect	X		X		X	X	X				X		
Kovair ALM	X	X			X	X		X		X	X		
MagicDraw	X	X	X		X	X				X	X		X
Matrix ALM/QMS	X			X	X	X	X				X		
Modelio Analyst	X	X	X			X		X		X	X	X	X
OpenProject	X	X	X			X	X				X	X	X
PivotalTracker		X				X	X				X		X
Polarion Requirements	X		X	X		X		X			X	X	X
Psoda	X	X				X	X			X	X	X	
ReqEdit	X	X			X	X					X		
ReQtest		X				X	X					X	
ReqView	X				X	X				X	X	X	X
RMsis	X	X			X	X	X				X		
RMTrak	X	X			X	X					X		
Scrumwise		X		X		X							
SpiraTeam	X	X	X		X	X					X	X	
SwiftKanban	X	X				X	X			X	X	X	X
Targetprocess	X	X	X	X	X	X	X	X		X	X	X	
Tuleap Enterprise	X					X	X	X		X	X	X	X
Valispace		X		X	X	X	X	X		X	X		
Visual Paradigm	X	X	X		X	X	X	X		X	X	X	
Visure Requirements	X	X			X	X	X	X			X	X	
Yodiz		X	X		X	X	X				X		
Accompa	X	X				X	X			X	X		
CA Agile Requirements Designer	X	X	X	X	X	X		X			X	X	

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

Table 9. *Cont.*

Requirements Engineering Tools	Tel.	E-mail	Forum	Live-chat	Help Desk	Guide	Blog	White paper	M. List	Case-study	Videos	Training	Coaching
ClickUp				X	X	X		X		X		X	
Doc Sheets	X	X			X	X	X					X	
Orcanos	X	X	X	X	X	X	X			X	X	X	
TopTeam	X	X				X							
Xebrio		X			X	X		X			X		
ReqChecker					X	X						X	
StoriesOnBoard		X				X	X	X		X			

4. Discussions

4.1. Summary of Findings

In Section 3, we analysed 56 different requirements engineering tools for a number of features that are categorised as the project management, specification, collaboration, customisation, interoperability, methodology, and knowledge management viewpoints. In the rest of this section, we summarise the key findings from our analysis.

The project management activities are not the priority for many tools. Only 46% of the requirements engineering tools considered provide built-in support for managing projects in terms of the project initiation, project planning, and project execution activities.

Multi-user collaboration support is provided by almost all the tools. All the requirements engineering tools considered support multi-user access - except ReqChecker. 78% of those tools further enable assigning users with different roles (e.g., editor, reader, developer, tester, manager, etc.) and configuring the access rights for the user roles.

Most of the tools provide their built-in versioning control system. 74% of the requirements engineering tools provide built-in versioning systems, while the support for external versioning systems (e.g., GIT, SVN, and Mercurial) remain very low.

The top-popular requirement specification technique is the natural language. 73% of the requirements engineering tools support the requirements to be specified in natural languages. Using those tools, requirements are specified informally in plain text format via some well-defined user-interfaces.

The precise specification of requirements that can easily be processed is rarely supported. 25% of the requirements engineering tools that support software modeling and design enable users to specify requirements precisely using well-accepted modeling languages such as SysML, UML, and BPMN. Using those tools, it is possible to specify requirements precisely, analyse requirements, and further transform the requirements.

Analysing requirement specifications to detect issues is rarely supported. Only 30% of the requirements engineering tools analyse the requirement specifications and most of those tools enable analysis thanks to their precise modeling support through the modeling languages. Note that a few tools use artificial intelligence and natural language processing techniques for checking the informal requriement specifications in natural languages.

Requirements transformation is mainly considered for generating documents from requirements. 75% of the requirements engineering tools support the requirement transformation and most of those tools only support generating documents in formats such as Word, Excel, HTML, and PDF. However, generating skeleton code and test-scenarios from requirements are rarely supported.

Tool customisation is highly popular by means of the external tool integrations and API support. 87% of the requirements engineering tools support the integration with many external tools including the test automation tools, project management tools, and versioning tools. Also, 70% of the tools provide their own APIs through which users can develop their own tool extensions.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

Extending the notation set for the requirement specifications is rarely supported. While the tool customisations/extensions are supported by most of the requirements engineering tools, the support for extending the notation set for specifying domain-specific requirements (e.g., defining modeling viewpoints and developing DSLs) is rare.

Most tools accept requirements in different formats and Excel/CSV are the top-popular formats. 89% of the requirements engineering tools enable importing requirements in different data formats including Excel, CSV, Word, XMI, Json, and PDF. The top import formats are Excel and CSV (31-32%), which is followed by Word (20%).

Many tools exchange requirements and ReqIF, Excel and CSV are the top-popular data formats. 77% of the requirements engineering tools enable exchanging (i.e., importing and exporting) the requirement specifications. The top data exchange formats are ReqIF (25%), Excel (23%), and CSV (23%).

The top-supported software development methodology is agile. 52% of the requirements engineering tools considered enable users to manage their requirements using agile principles and techniques. Model-driven engineering is supported by 32% of the requirements engineering tools and product-line engineering is supported by just 3 tools.

User-guide, telephone details, e-mail addresses, and videos are the four top-popular techniques for sharing tool-related knowledge. While all the requirements engineering tools considered provide user-guide documents, 73-77% of the tools provide telephone details, e-mail addresses and videos.

4.2. Lessons Learned

Most requirements engineering tools promote the requirements to be specified in natural languages. While using natural languages is important for reducing the learning curve and enabling even non-technical users to specify their requirements quickly, processing natural language specifications can be very hard (if not impossible). The use of natural languages cause the tools to avoid processing requirements and focus more on the requirements gathering and documentation rather than the requirements analysis and transformation. The natural language processing techniques and technologies are rarely adopted by most of the tool vendors for developing and integrating requirements analysis tools. The tools with software modeling and design support (e.g., Cameo, Visual Paradigm, Magic Draw, and Enterprise Architect) do enable the automated analysis and simulation of requirement specifications and their transformation into skeleton code. However, those tools require users to use modeling languages with precise definitions. Note that while the precision here facilitates the analysability and transformation of models, non-technical users may not find it easy to learn and use the modeling languages for their requirement specifications.

While the requirements engineering tools enable the tool extensions via external tool integrations and API support, most of the tools ignore extending the notation sets used for specifying requirements (e.g., enabling DSL development). Only a small number of tools enable users to extend the well-known UML language via UML's profiling mechanism and define a domain-specific notation set. However, it should be noted that domain-specific modeling is highly important in increasing the productivity and maximising the quality of the requirements engineering [29]. Indeed, many industries develop their own domain-specific languages using meta-modeling technologies [30,31].

We also learned that most of the requirements engineering tools support the exchange of requirements data among different tools. However, no any single standard (e.g., ReqIF) has been adopted by the all of the tools for the data exchange. Also, none of the tools share any case-studies that demonstrate exchanging the requirements among different tools. So, it is not so clear how effectively the requirements exchange work between different tools. Another interesting lesson is that while ReqIF is considered as a well-regarded standard for exchanging requirements, only 25% of the requirements engineering tools support ReqIF. The rest of the requirements engineering tools support some other formats including Excel, CSV and XML.

 doi:10.20944/preprints202310.0424.v1

Most of the requirements engineering tools support the agile principles and such techniques as Scrum, Kanban, and lean. While the agile techniques help in managing the projects effectively and developing software systems that meet the customer needs, agile development does not essentially address the important principles of software engineering such as reusability, maintainability, automated code (or test scenario) generation and early detection of errors. Unfortunately, such methodologies as product line engineering and model-driven engineering that are believed to enhance the software quality are rarely supported by the existing tools.

Software development life-cycle starts with the requirements engineering and continues with other processes to be performed including design, implementation and testing under the guidance of some process models (e.g., waterfall, V-model, incremental model, spiral model, etc.) [32]. To develop the right systems right, the artifacts produced in the requirements should be linked with the artifacts produced in design and implementation, which enable tracing from requirements till to the implementation (and vice versa). However, none of the requirements engineering tools considered enable round-trip engineering (i.e., going from requirements to design and then design to code, and backwards). Indeed, none of the tools enable receiving requirements in natural languages, specifying and verifying design models, linking requirements with the design models, and transforming all those into test-scenarios and code.

Another lesson is to do with the knowledge management as many of the requirements engineering tools show inadequate support for sharing their tools' knowledge. Indeed, given 13 different criteria considered for the knowledge management, more than half of the tools support 6 of those criteria at most and therefore those tools may require some learning curve for the users as the users may not easily figure out how to use the tools and their particular features as they wish.

4.3. Threats to Validity

Internal validity is concerned with causal relationships between the results of the analysis and any independent variable (i.e. cause) that leads to the results [33]. In this study, nonprobabilistic sampling was used and the requirements engineering tools were chosen non-randomly. That is, the tools to be analyzed were searched from the Internet systematically and filtered according to the exclusion criteria as discussed in Section 2.1. In addition, the tools were analyzed for various requirements engineering features categorized in seven viewpoints. Derivation and the formalization of these features and composing viewpoints were based on our previous research on understanding practitioners' experiences in requirements engineering [12].

Three experienced researchers have been involved in the analysis of the requirements engineering tools. To minimise any instrumentation biases here, we ensured that the three researchers have analysed the same set of tools independently from each other in the same systematic way that is discussed in Section 2.3. The results obtained from the researchers have been compared to detect any inconsistencies on the data analysis.

External validity threats concern the generalizability of the analysis results, that is, the degree to which the examined studies are the representative of the reviewed topic [33]. The set of requirements engineering tools analyzed in our study may not be representative of the entire set of all available tools. However, this threat was mitigated by an extensive search on the Internet using various keywords as listed in Section 2.1.

Construct validity relates to how well an analysis helps in achieving the research objective. Our goal was to analyze the capabilities of the existing requirements engineering tools according to the practitioners' needs. For this purpose, we categorized various requirement engineering features under seven viewpoints and used them during our analysis. The analyzed data were compared, existing inconsistencies were determined and these parts were re-analyzed together with all authors until they reached a consensus on them. This method also contributed to minimizing the risk on the construct validity of the conducted research. Additionally, we need to assure that all relevant requirement engineering tools are found adequately. For this purpose, well-known terms/concepts related to

 doi:10.20944/preprints202310.0424.v1

the requirement engineering tools and platforms were used to create search strings, several search iterations were provided and hence the adequate coverage of the all available tools was achieved.

Finally, to minimize the conclusion validity threat, the research methodology of this study was designed and validated carefully to minimize the risk of excluding relevant requirement engineering tools. Benefiting from our previous experience on conducting other analysis studies (e.g. [34] and [35]), the search methodology in here was formalized and applied in a way that only a very small number of relevant requirement engineering tools could be missed, and a manageable quantity of irrelevant ones could be included. Furthermore, the findings of the performed analysis were assessed within the context of the set of tool features provided at the beginning of the study.

5. Related Work

In this section, we discuss the similar studies that compare a set of requirements engineering tools for a number of features.

In [36], the authors analysed 13 important requirements engineering tools by observing their practical use in client environments. The authors essentially considered the agile methodologies, collaboration, and test-driven requirements engineering. The authors also provided an interesting guidance on how to use the requirements engineering tools.

In [37], the authors analysed 7 requirements engineering tools that have been considered to address the security requirements. The authors focus on understanding those tools' weak points in terms of gathering and documenting security requirements in a precise way that can be performed by stakeholders with limited technical knowledge and further validated.

In [38,39], the authors surveyed among the vendors of 38 different requirements engineering tools. The authors' survey consists of 146 questions and with those questions, the tool vendors were expected to rank their tools support for different capabilities. Moreover, the authors performed three separate scenarios to better understand the particular features that are more important for practitioners and the tools' support for those features.

In [13], the author analysed 58 different UML modeling tools for a number of requirements that are considered important for practitioners. Note however that the author here focussed on analysing the tools with UML support rather than the tools that support requirements management.

In [40], the authors analysed 21 different requirements engineering tools for a number of common requirements management features so as to understand which features are more popular and which are less popular. The authors did not focus on providing any precise information about the tools' support for any particular features

In [41], the authors analysed 13 different requirements engineering tools for a set of features about the requirements management and traceability. The authors just indicated which tool support which of the features considered with Yes/No answers. The authors did not focus on collecting precise data about the features considered and performing thorough analysis of the tools.

In [42], the authors considered 12 different requirements engineering tools and collected data about the tools' support for a number of features so as to understand tools' capabilities. However, the authors basically shared the collected data and ignored any analysis of the collected data for revealing any interesting results or lessons.

In [43], the authors considered 10 different tools with regard to their support for the artificial intelligence based requirements gathering and requirements management techniques and activities. The authors basically introduced the tools' support for some features with Yes/No style answers (i.e., *Low, Medium* and *High*), which lack in any thorough discussions of the analysis results or lessons learned.

In [44], the authors considered 8 different requirements engineering tools and categorised the tools as heavyweight, middleweight, and lightweight tools. The authors focussed on a set of features and analysed each tool to understand how many of those features of interest are supported.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 October 2023 doi:10.20944/preprints202310.0424.v1

In [45], the authors conducted a survey among 117 students to understand their thoughts on 9 highly-used UML modeling tools. The authors focus on understanding from the perspectives of the students the main benefits and drawbacks of the tools.

According to our observations, the above-mentioned studies existing in the literature differ from our study discussed in this paper. Firstly, unlike our study, many of the existing studies focus more on understanding the tools that support certain features without collecting and analysing any precise data about those features. For instance, the support for the requirements transformation could be considered more precisely if the requirements transformation was addressed in terms of code generation, test-scenario generation, and document generation as we do in our paper. Moreover, as indicated in Table 10, our study is distinguished with its consideration of a comprehensive set of tools that are analysed for a comprehensive set of features which are grouped into multiple viewpoints. Also, as indicated in Table 11, our study is further distinguished with its consideration of the project management, collaboration, requirements specification, customisation, interoperability, methodology, and knowledge management viewpoints at the same time each of which is addressed with a cohesive set of features by our study.

Table 10. The similar studies in the literature

Study	Year	Number of tools	Number of features
[36]	2021	13	17
[37]	2013	7	9
[38,39]	2012	38	23
[13]	2019	58	9
[40]	2017	21	8
[41]	2011	13	9
[42]	2003	12	13
[43]	2022	10	10
[44]	2022	8	7
[45]	2019	31	Student Survey
Our Study	2023	56	20

Table 11. The analysis of the similar studies with regard to the viewpoints focussed in our study

Study	Project Man.	Specification	Collaboration	Customisation	Interoperability	Methodology	Knowledge Man.
[36]	No	No	Yes	No	Yes	Yes	Yes
[37]	No	Yes	No	No	No	Yes	No
[38,39]	Yes	Yes	No	No	Yes	No	No
[13]	Yes	Yes	Yes	Yes	Yes	No	Yes
[40]	No	Yes	No	Yes	No	No	No
[41]	No	Yes	No	Yes	No	No	No
[42]	No	Yes	Yes	No	Yes	No	No
[43]	No	Yes	Yes	No	No	No	No
[44]	No	Yes	Yes	Yes	No	No	No
[45]	No	No	No	No	No	No	No
Our Study	Yes	Yes	Yes	Yes	Yes	Yes	Yes

6. Conclusion

In this study, we analysed 56 different requirements engineering tools for a comprehensive set of features that are categorised into multiple viewpoints (i.e., project management, specification, collaboration, customisation, interoperability, methodology, and knowledge management). The analysis results reveal many important lessons. Most requirements engineering tools promote the requirements to be specified in natural languages. While using natural languages is important for reducing the learning curve, processing natural language specifications can be very hard (if not impossible). The requirements engineering tools enable the tool extensions via external tool

integrations and API support, however, most of the tools ignore extending the notation sets used for specifying requirements. Most tools support the exchange of requirements data among different tools. However, no any single standard (e.g., ReqIF) has been adopted by the all of the tools for the data exchange. Most tools support the agile principles and such techniques as Scrum, Kanban, and lean, while any other methodologies such as product-line engineering and model-driven engineering that can significantly improve the quality of software development are rarely supported. None of the tools enable round-trip engineering - i.e., receiving requirements in natural languages, specifying and verifying design models, linking requirements with the design models, and transforming all those into test-scenarios and code.

The analysis results are expected to be useful for practitioners, tool vendors, and the researchers in this field. Indeed, practitioners can use the results to compare the existing requirements engineering tools and find out the one(s) that best meet their needs. Tool vendors can use the results to determine the gaps that can be improved. Researchers can conduct further empirical studies to better understand practitioners' perspectives and propose new projects that improve the existing requirements engineering tools.

In the future, our first goal is to conduct a series of interviews with a group of practitioners from diverse industries so as to receive their feedback about the analysis results and validate the lessons learned. We will also design a survey that primarily focuses on practitioners' challenges on the requirements engineering tools.

References

1. IEEE Standard for Application and Management of the Systems Engineering Process. *IEEE Std 1220-2005 (Revision of IEEE Std 1220-1998)* **2005**, pp. 1–96. doi:10.1109/IEEESTD.2005.96469.
2. (Ed.), R.A. Guide to the Systems Engineering Body of Knowledge (SEBoK), version 1.9, 2017. Online; accessed 01 Apr 2023 13:56:12 UTC.
3. Lethbridge, T.C.; Lagamiere, R. *Object-oriented software engineering - practical software development using UML and Java*; MacGraw-Hill, 2001.
4. Curcio, K.; Navarro, T.; Malucelli, A.; Reinehr, S.S. Requirements engineering: A systematic mapping study in agile software development. *J. Syst. Softw.* **2018**, *139*, 32–50. doi:10.1016/j.jss.2018.01.036.
5. Nuseibeh, B.; Easterbrook, S. Requirements Engineering: A Roadmap. Proceedings of the Conference on The Future of Software Engineering; Association for Computing Machinery: New York, NY, USA, 2000; ICSE '00, p. 35–46. doi:10.1145/336512.336523.
6. Laplante, P.A. *Requirements Engineering for Software and Systems, Third Edition*, 3rd ed.; Auerbach Publications: USA, 2017.
7. Pohl, K. *Requirements Engineering: Fundamentals, Principles, and Techniques*, 1st ed.; Springer Publishing Company, Incorporated, 2010.
8. Humphrey, W. Why Big Software Projects Fail: The 12 Key Questions. *Software Management* **2006**, pp. 21–26.
9. Charette, R. Why software fails [software failure]. *IEEE Spectrum* **2005**, *42*, 42–49. doi:10.1109/MSPEC.2005.1502528.
10. Hussain, A.; Mkpojiogu, E.O.C. Requirements: Towards an understanding on why software projects fail. *AIP Conference Proceedings* **2016**, *1761*, 020046, [https://aip.scitation.org/doi/pdf/10.1063/1.4960886]. doi:10.1063/1.4960886.
11. Ebert, C.; Jastram, M. ReqIF: Seamless Requirements Interchange Format between Business Partners. *IEEE Softw.* **2012**, *29*, 82–87. doi:10.1109/MS.2012.121.
12. Ozkaya, M.; Akdur, D.; Toptani, E.C.; Kocak, B.; Kardas, G. Practitioners' Perspectives towards Requirements Engineering: A Survey. *Syst.* **2023**, *11*, 65. doi:10.3390/systems11020065.
13. Ozkaya, M. Are the UML modelling tools powerful enough for practitioners? A literature review. *IET Software* **2019**.
14. Westland, J. *The Project Management Life Cycle: A Complete Step-By-Step Methodology for Initiating, Planning, Executing & Closing a Project Successfully*; Kogan Page, Limited, 2006.

15. Taylor, R.N.; Medvidovic, N.; Dashofy, E.M. *Software Architecture - Foundations, Theory, and Practice*; Wiley, 2010; pp. I–XXIV, 1–712.
16. Rumbaugh, J.E.; Jacobson, I.; Booch, G. *The unified modeling language reference manual*; Addison-Wesley-Longman, 1999; pp. I–XVII, 1–550.
17. Balmelli, L. An Overview of the Systems Modeling Language for Products and Systems Development. *J. of Obj. Tech.* **2007**, *6*, 149–177. www.sysml.org.
18. Völzer, H. An Overview of BPMN 2.0 and Its Potential Use. Business Process Modeling Notation - Second International Workshop, BPMN 2010, Potsdam, Germany, October 13-14, 2010. Proceedings; Mendling, J.; Weidlich, M.; Weske, M., Eds. Springer, 2010, Vol. 67, *Lecture Notes in Business Information Processing*, pp. 14–15. doi:10.1007/978-3-642-16298-5_3.
19. Feiler, P.H.; Gluch, D.P.; Hudak, J.J. The Architecture Analysis & Design Language (AADL): An Introduction. Technical report, Software Engineering Institute, 2006.
20. Zowghi, D.; Gervasi, V. On the interplay between consistency, completeness, and correctness in requirements evolution. *Inf. Softw. Technol.* **2003**, *45*, 993–1009. doi:10.1016/S0950-5849(03)00100-9.
21. IEEE Recommended Practice for Architectural Description for Software-Intensive Systems. *IEEE Std 1471-2000* **2000**, pp. 1–30. doi:10.1109/IEEESTD.2000.91944.
22. Selić, B.; Gérard, S. Chapter 2 - An Introduction to UML Profiles. In *Modeling and Analysis of Real-Time and Embedded Systems with UML and MARTE*; Selić, B.; Gérard, S., Eds.; Morgan Kaufmann: Boston, 2014; pp. 27–43. doi:https://doi.org/10.1016/B978-0-12-416619-6.00002-X.
23. Ringert, J.O.; Rumpe, B.; Schulze, C.; Wortmann, A. Teaching Agile Model-Driven Engineering for Cyber-Physical Systems. 39th IEEE/ACM International Conference on Software Engineering: Software Engineering Education and Training Track, ICSE-SEET 2017, Buenos Aires, Argentina, May 20-28, 2017. IEEE Computer Society, 2017, pp. 127–136. doi:10.1109/ICSE-SEET.2017.16.
24. Mohan, K.; Ramesh, B.; Sugumaran, V. Integrating Software Product Line Engineering and Agile Development. *IEEE Softw.* **2010**, *27*, 48–55. doi:10.1109/MS.2010.31.
25. Model-Driven and Software Product Line Engineering. In *Model-Driven and Software Product Line Engineering*; John Wiley & Sons, Ltd, 2012; chapter 4, pp. 101–138, [https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118561379.ch4]. doi:https://doi.org/10.1002/9781118561379.ch4.
26. Dingsøyr, T.; Nerur, S.; Balijepally, V.; Moe, N.B. A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software* **2012**, *85*, 1213–1221. Special Issue: Agile Development, doi:https://doi.org/10.1016/j.jss.2012.02.033.
27. Metzger, A.; Pohl, K. Software product line engineering and variability management: achievements and challenges. Proceedings of the on Future of Software Engineering, FOSE 2014, Hyderabad, India, May 31 - June 7, 2014; Herbsleb, J.D.; Dwyer, M.B., Eds. ACM, 2014, pp. 70–84. doi:10.1145/2593882.2593888.
28. Kent, S. Model Driven Engineering. Integrated Formal Methods, Third International Conference, IFM 2002, Turku, Finland, May 15-18, 2002, Proceedings; Butler, M.J.; Petre, L.; Sere, K., Eds. Springer, 2002, Vol. 2335, *Lecture Notes in Computer Science*, pp. 286–298. doi:10.1007/3-540-47884-1_16.
29. Wasowski, A.; Berger, T. *Domain-Specific Languages - Effective Modeling, Automation, and Reuse*; Springer, 2023. doi:10.1007/978-3-031-23669-3.
30. Kosar, T.; Bohra, S.; Mernik, M. Domain-Specific Languages: A Systematic Mapping Study. *Information & Software Technology* **2016**, *71*, 77–91. doi:10.1016/j.infsof.2015.11.001.
31. Leblebici, O.; Kardas, G.; Tuglular, T. A Domain-Specific Language for the Document-Based Model-Driven Engineering of Business Applications. *IEEE Access* **2022**, *10*, 104093–104110. doi:10.1109/ACCESS.2022.3210530.
32. Ruparelia, N.B. Software development lifecycle models. *ACM SIGSOFT Softw. Eng. Notes* **2010**, *35*, 8–13. doi:10.1145/1764810.1764814.
33. Wohlin, C.; Runeson, P.; Höst, M.; Ohlsson, M.C.; Regnell, B. *Experimentation in Software Engineering*; Springer, 2012. doi:10.1007/978-3-642-29044-2.
34. Mohamed, M.A.; Challenger, M.; Kardas, G. Applications of model-driven engineering in cyber-physical systems: A systematic mapping study. *J. Comput. Lang.* **2020**, *59*, 100972. doi:10.1016/j.cola.2020.100972.
35. Arslan, S.; Ozkaya, M.; Kardas, G. Modeling Languages for Internet of Things (IoT) Applications: A Comparative Analysis Study. *Mathematics* **2023**, *11*. doi:10.3390/math11051263.

36. de Gea, J.M.C.; Ebert, C.; Hosni, M.; Vizcaíno, A.; Nicolás, J.; Alemán, J.L.F. Requirements Engineering Tools: An Evaluation. *IEEE Softw.* **2021**, *38*, 17–24. doi:10.1109/MS.2021.3058394.
37. Yahya, S.; Kamalrudin, M.; Sidek, S. A review on tool supports for security requirements engineering. 2013 IEEE Conference on Open Systems (ICOS). IEEE, 2013, pp. 190–194. doi:10.1109/ICOS.2013.6735072.
38. de Gea, J.M.C.; Nicolás, J.; Alemán, J.L.F.; Álvarez, J.A.T.; Ebert, C.; Vizcaíno, A. Requirements Engineering Tools. *IEEE Softw.* **2011**, *28*, 86–91. doi:10.1109/MS.2011.81.
39. de Gea, J.M.C.; Nicolás, J.; Alemán, J.L.F.; Toval, A.; Ebert, C.; Vizcaíno, A. Requirements engineering tools: Capabilities, survey and assessment. *Inf. Softw. Technol.* **2012**, *54*, 1142–1157. doi:10.1016/j.infsof.2012.04.005.
40. Shah, A.; Alasow, M.A.; Sajjad, F.; Baig, J.J.A. An evaluation of software requirements tools. 2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS). IEEE, 2017, pp. 278–283. doi:10.1109/INTELCIS.2017.8260075.
41. Shahid, M.; Ibrahim, S.; Mahrin, M.N. An Evaluation of Requirements Management and Traceability Tools. *International Journal of Computer and Information Engineering* **2011**, *5*, 627 – 632.
42. Sud, R.R.; Arthur, J.D. Requirements management tools: A quantitative assessment. Technical Report TR-03-10, Department of Computer Science, Virginia Polytechnic Institute & State University, 2003.
43. Nadeem, M.A.; Lee, S.U.J.; Younus, M.U. A Comparison of Recent Requirements Gathering and Management Tools in Requirements Engineering for IoT-Enabled Sustainable Cities. *Sustainability* **2022**, *14*. doi:10.3390/su14042427.
44. Inam-Ul-Haq.; Abbas, W.; Butt, W.H. Systematic Literature Review on Requirement Management Tools. 2022 International Conference on Emerging Trends in Smart Technologies (ICETST). IEEE, 2022, pp. 1–6. doi:10.1109/ICETST55735.2022.9922932.
45. Agner, L.T.W.; Lethbridge, T.C.; Soares, I.W. Student experience with software modeling tools. *Softw. Syst. Model.* **2019**, *18*, 3025–3047. doi:10.1007/s10270-018-00709-6.