One to Rule them All: A Study on Requirement Management Tools for the Development of Modern AI-based Software

Abdul-Rasheed Ottun*, Mehrdad Asadi†, Michell Boerger¶, Nikolay Tcholtchev¶,
João Gonçalves‖, Dušan Borovčanin§, Bartlomiej Siniarsk‡, and Huber Flores*
*University of Tartu, Estonia
†Vrije Universiteit Brussel, Belgium
‡University College Dublin, Ireland
§MAINFLUX LABS, Serbia
¶Fraunhofer Institute for Open Communication Systems, Germany
‖Erasmus University Rotterdam, The Netherlands
firstname.lastname@{ut.ee, vub.be, ucd.ie, mainflux.com, fokus.fraunhofer.de, eshcc.eur.nl}

***Abstract*—Modern system architectures are rapidly adopting AI-based functionality. As a result, new requirements about software trustworthiness must be considered during the entire software development life cycle of applications. While several requirement management tools are available to track and monitor requirements over time, it is still unknown to what extent these tools can cope with these new demands imposed by AI. In this paper, we contribute by performing a qualitative and quantitative analysis of different requirement management tools and their performance in managing AI-related requirements effectively. Through a rigorous analysis performed by a consortium formed by different industry and academic partners, we evaluate the suitability of five different requirement management tools. Our results indicate that while several tools are available for managing requirements, it is currently challenging to find a tool that can manage AI requirements mainly because tools do not comply with the required aspects imposed by regulatory entities. Lastly, we also shared our lessons learned and experiences from selecting requirement tools that can be used in team-based consortium projects.**

***Index Terms*—Requirement Engineering; XAI, AI Applications; Trustworthy AI**

I. Introduction

Modern system architectures are rapidly adopting AI-based functionality [1]. Indeed, advanced machine and deep learning models can improve the usability and performance of the applications that we use in our daily life activities. For instance, transportation systems can rely on AI for better navigation of autonomous vehicles, and healthcare can improve the accuracy of diagnosis for different diseases using AI [2]–[4]. A major challenge to integrating AI into the software development life cycle is that standard pipelines for building AI models can be easily hampered (through adversarial of situational changes) at any phase of the model construction [5]–[7], e.g, data poisoning during data collection. Besides this, the resulting AI model is black-box, meaning that the logic used to make a decision is obscured to users. Another important challenge is to make AI trustworthy for its adoption at scale [8]. Ensuring that AI is trustworthy requires taking into consideration several trustworthy properties as early as the conceptual designs defining the applications conceived. Trustworthy properties of AI are a set of characteristics that AI requires to be equipped with, for instance, explainability, interpretability, managing biases, data governance, privacy enhancement, and safety, to mention some. Tracking the fulfillment of these characteristics over time and their changes require the use of specialized management tools.

Regulations defined for the development of AI-based software, e.g., EU GDPR [9] and US AI Executive Order (EO) 13960 [10]; have introduced guiding principles imposing ethical, lawful, and robust considerations for the development and deployment of AI-based solutions. As early conceptual and blueprint decisions can influence the fulfilling of these regulations in practice, trustworthy properties are required to be considered as early as the requirements collection phase. Indeed, requirements can be greatly affected due to the sensitivity and privacy of data that is used to train/retrain AI models. For instance, the use of data in mobile applications becomes situational, requiring, in some cases, consent from surrounding individuals to use their data [11]. As requirements capture private related information from users, this should be reflected throughout the development process. Likewise, the type of AI algorithm selected can also influence the post de-facto verification of AI behavior, making it difficult to assess characteristics, such as transparency and resilience. Another example is tracking specifications regarding the resilience of AI models to the large spectrum of possible attacks that influence their decision process [12], e.g., model evasion and model stealing. As a result, the monitoring and tracking of AI requirements becomes critical for the auditability of accountability of developed AI-based software. While there is a large variety of (open source and off-the-shelf commodity) tools available

979-8-3503-2445-7/23/$31.00 ©2023 IEEE

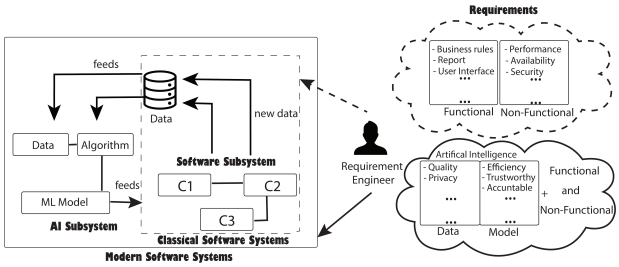

Fig. 1: Conceptual modern software architectures implementing machine learning [1] and set of evolving AI requirements to be tracked and monitored.

for tracking requirements and monitoring them over time [5], [13], not all the tool provide flexibility and adaptability to cope with dynamic changes in data and model characteristics. More importantly, besides the established regulations, other stakeholder considerations also become important for the selection of the management tool as the construction process of AI may require to apply different internal policies affecting each involved stakeholder.

We contribute by analyzing the suitability of different requirement management tools. As large software applications are typically built in large teams, located in different areas and belonging to different organizations. To evaluate the effectiveness of different management tools, we conduct the analysis in the context of an EU Horizon project formed by a consortium of industry and academic partners located across different EU and non-EU countries. This analysis is conducted using the Commercial Off-The-Shelf (COTS) methodology. By using COTS, we assessed whether popular and off-the-shelf commercial tools can be used easily to manage and track the requirements of AI-based software. In this analysis, five technical partners are selected, where one is appointed as a coordinator, defining the overall criteria for evaluating the tool. The coordinator also is responsible for assigning each partner a tool, such that each individual partner can evaluate the tool against the criteria. The results of our analysis demonstrate that new regulations imposed for the development of AI reduced the pool of options available for handling requirements. Besides this, our results demonstrate that while internal policies of organizations may be also a problem when selecting requirements management tools, it is much easier to bypass them when compared with rules imposed by regulatory entities. Lastly, we also share our lessons learned and experiences from selecting requirement tools that can be used in team-based consortium projects.

II. Background and Related Work

Requirement management (RM) tools support many requirement engineering processes such as gathering, elicitation, analysis, and verification within the software development lifecycle [14]. These tools are typically used to track and monitor the evolution of requirements during the whole development process. At the same time, RM tools provide a way to document and quantify the building process of software applications [15]. Several studies have categorized different tools based on their functionalities and features [14], [16], [17]. As modern system architectures evolve, RM tools also provide over time new features [18] that are identified by different stakeholders involved in the development of applications [19], e.g., software developers and data scientists, among others.

On the other hand, regulatory entities also have introduced new aspects and properties that computing software has to consider when implementing AI-based functionality. For instance, as classical software architectures are augmented to consider new paradigms such as machine and federated learning [20], new features in RM tools are required that take into consideration the collection of data at scale. Besides this, systems and applications implementing AI are required to adopt trustworthy properties, requiring analysis of other dimensional aspects of data used for training AI models, such as fairness and transparency [13]. Indeed, trustworthy AI depicts a set of trustworthy properties required for computing programs to be considered trustful [8] and several studies have identified new types of requirements to consider in these modern systems [13]. In this work, we revisit the list of requirements to be considered in AI-based systems, and unlike others, we analyze whether off-the-shelf RM tools can cope with these new considerations.

III. Evolution in Software Requirements

Classical Software Requirements: Majority of classical software requirements focus on providing functional and non-functional features according to the client's needs. As shown in Table I, these requirements ensure that the applications meet the behavior and contain features required during the software development process. Functional requirements range from use case business rules and transactional correctness to user interfaces that enable the client to achieve a higher user experience. On the other hand, non-functional requirements are quality attributes that determine how the system should behave after deployment. For instance, even if the functionality is as expected, having poor performance after deployment may lead to a negative user experience and jeopardize system safety. Table I describes the classical software requirements in detail.

GDPR and imposed regulations: Europe is leading the verification of AI-based solutions, such that AI is trustworthy to users. The General Data Protection Regulations (GDPR) stipulates the guidelines for dealing with personal data within the European Union (EU), putting forward fairness, security, privacy, trust, transparency, and explanation considerations during software and AI-based solution development. Practically, the guidelines have compliance implications on data and software architecture [21], [22] as they obligate these

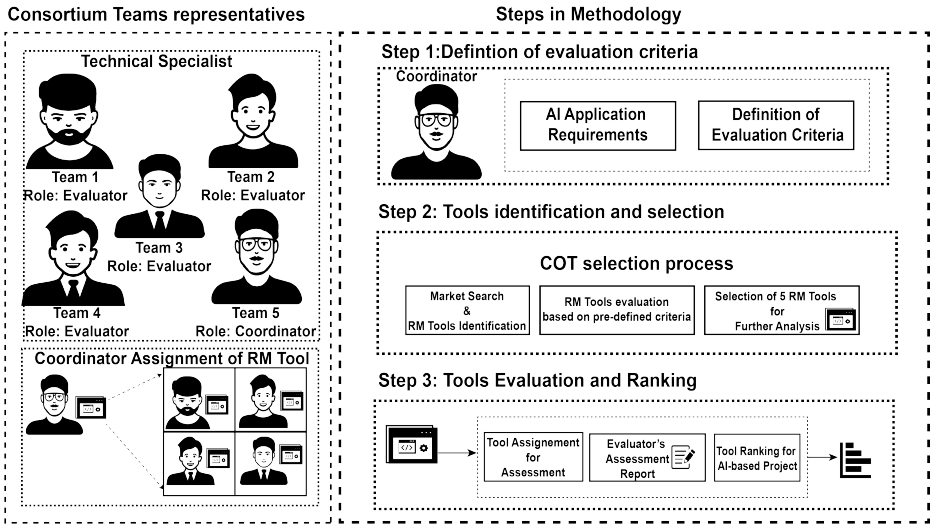

Fig. 2: Evaluation methodology.

Requirement type	Feature	Description
Functional	Business rules	Transactional functions that must be performed in the end product based on user stories
	External interfaces	API design and accessibility to external APIs
	Access control	Authentication of end users
Non-functional	Performance	Response time of application
	Availability	Continuous availability of the system for users.
	Scaleability	Ability to perform as expected under load.
	Usability	user-friendly and easy to use system.
	Security	Authorization and resource access control

TABLE I: Generic requirements of applications

considerations for validating and verifying that the software solutions and systems are rightly and correctly developed [23], [24]. As a result, software engineers, requirement engineers, and other practitioners now have to consider a new set of requirements like data traceability, minimization, rectification, and erasure, system security, and privacy that have been imposed by the regulations as part of system requirements during the system design and development. Similarly, these principles are also described in the US AI ACT, and other countries have also considered similar regulations, for instance, China, Japan, Brazil, and Canada.

Modern Software Requirements: The inclusion of AI functionality in applications has changed significantly the elicitation of requirements when designing software [1]. As shown in Figure 1, the current software systems leverage AI (e.g., machine learning), leading to a revolution in intelligent software design. Since AI modules within software mainly rely on data, modern software requirements change accordingly. As shown in Table II, in addition to current classical software requirements, AI-enabled systems raise a set of new requirements related to data and the internal model logic of AI. For instance, AI functionality highly relies upon the quantity and quality of data fed to the system. Moreover, keeping an individual's privacy during data collection should be considered by AI-driven software engineers.

IV. Analysis of RM tools: Methodology

Assumptions: The analysis of RM tools is conducted in the context of a project that aims to design and develop an AI-based solution. Our project brings together specialists from the academy and industry to build a software solution that can be used to analyze AI models running in modern applications. During the development of our solution, we assume that each team can access the overall code and contribute to the development of the application. In addition to this, once a tool is selected for the joint development of the solution, we also assume that requirements are tuned over time solely by the coordinator of the analysis, such that issues about accessing data by other teams do not infringe our analysis criteria.

Methodology: Our analysis is conducted by relying on the principles of COTS methodology. Figure 2 describes the overall process. By using COTS, we selected a reasonable number of RM tools and applied clear steps to conduct individual evaluations of each tool by different teams. To use COTS, first, we defined specific criteria for establishing the baseline features for the RM tools to be considered. Subsequently, we searched to identify relevant RM tools that align with the predefined criteria. Right after, we then systematically assessed each tool by relying on individual evaluations conducted by individual teams over assigned tools. Next, we explain the step-by-step procedure in detail as follows:

Requirement type	Feature	Description
Data	Privacy	Keeping private data anonymous
	Quality	Ensure quality of data before model fitting
	Bias-free	Completeness of data
Model	Efficiency	To perform efficient in terms of energy / time
	Robustness	To be robust against adversaries
	Trustworthiness	Transparent, explainable, and accountable AI
System	Resilience	Perform as expected in case of circumstance change
	Reliability	The probability that AI-based system performs correctly for a time period.
	Safety	Minimize harmful consequences

TABLE II: AI-based application requirements

A. Step 1: Definition of evaluation criteria

After identifying the essential functionalities of our AI-based application within the context of our project. We adopted the comprehensive and formal guideline provided by **ISO/IEC TR 24766:2009** [25] framework. The framework enabled us to understand the mapping of requirement management activities to the capabilities of any RM tools as the standard is the classification framework for evaluating relevant features of RM tools [16]. In addition, this framework availed us with crucial insights into the specific capabilities that candidate RM tools should possess. Following that, we established workshops and discussion sessions to elaborate on other criteria that RM tools must possess to serve our purpose for effective support of AI-based application development. Overall, we conducted six (6) workshops with a duration of forty (40) minutes. These criteria are defined as follows:

(a) Extensibility: RM tools that are open source and publicly accessible are relevant to our overall objective. This allows the development team to modify and extend the code as needed. Although these tools rely on community production and peer review, they are often more affordable and flexible and do not require additional production team costs to manage requirements. Moreover, this feature helps us to have an extensibility feature that enables us to adapt the RM tools and extend them to meet the specific needs of different stakeholders. Furthermore, the criterion demands that the RM tools to be considered must have detailed and comprehensible documentation about object models, interfaces, and API such that data, requirement artifacts, and functions can be adapted and extended.

(b) Textual Requirements: Textual requirements are text statements about the features or functionality of the application under development. The ability of users to use text is crucial for RM tools. The needs and expectations of stakeholders are easily captured by way of text. The captured information provides the basis for application design and development.

(c) Exportability: Exportability criterion relates to the ability to export data in different formats. This capability ensures that users can access and share requirements information outsides of the RM tool environment with third-party applications or relevant stakeholders without access to the tools.

(d) GDPR Compatibility: AI-based applications use a lot of sensitive data as such, compliance with privacy regulations is crucial for RM tools deployed for managing the requirements of AI-based applications. RM tools must be designed to address privacy and data protection concerns and comply with GDPR and other relevant regulations throughout the requirement management process and application development lifecycle. RM tools to consider must possess functionalities that can guarantee the confidentiality and integrity of sensitive information. At the basic level, the tools must sufficiently pass the GDPR principles described in Table IV.

(e) Local Installation: Since it is important to keep the confidential data of the project on local servers, it is important to have the tools installed locally and not on outsourced and untrusted servers.

(f) Requirements Traceability: Traceability of elements is vital for requirement monitoring. It encompasses the specification and the tracking of the relationships between system elements [31], [32].

B. Step 2: Tools Identification and Selection

In this step, we focused on tool identification and assessment. To identify a suitable set of RM tools, we conducted a search of available options in the market. We discovered a variety of RM tools are available with different features and functionalities. However, considering the time limitations and the overwhelming number of options, it was impracticable to assess all the identified tools. Hence, we adopted the COTS method alongside the criteria defined in Step 1 for selecting a cross-section of the identified tools for further evaluation of their capabilities in managing requirements in the context of AI-based system development.

C. Step 3: Tools evaluation

After several iterations of assessment of identified tools based on pre-defined criteria, we selected the five (5) tools that can be adapted for our AI-based application for further evaluation with stakeholders involved in the application development. Each of the selected tools was evaluated in detail against the criteria described in Step 2. We randomly distributed the tools among different project stakeholders and asked them to evaluate the assigned tool based on the proposed criteria. The criteria of the evaluation were also explained in detail to each of the teams assessing the tools. Table III shows the evaluation outcomes and demonstrates if the tools comply with the defined criteria.

Criteria	Tools				
	OSRMT [26]	RMTOO [27]	OpenReq [28]	Doorstop [29]	CAIRIS [30]
Extensibility	✓	✓	✓	✓	✓
Local Installation	✓	✗	✗	✓	✓
Traceability	✓	✗	✗	✓	✓
GDPR Compliance	✗	✗	✗	✗	✓
Export Capability	✗	✗	✓	✓	✓

TABLE III: Requirement Management tools criteria.

GDPR Principle	Description
Lawfulness, Fairness, and Transparent	Data with privacy properties are processed only if it's recognized as personal data. Use cases involving personal data are associated with a necessary goal or requirement.
Purpose Limitation	Use cases involving personal data are associated with a necessary goal concerned with that personal data.
Data Minimisation	Data with privacy properties are accounted for in processes.
Accuracy	Personal data has an Integrity security property.
Storage Limitation	Personal data in data stores is processed.
Integrity & Confidentiality	Personal information has confidentiality, integrity, and privacy properties that must be preserved.

TABLE IV: GDPR Principles and Descriptions [30].

D. Threats to validity

The criteria for the selection of the RM tools and evaluators is based on the consortium's requirement management needs and the stakeholders, the potential domain experts and practitioners, who are the users of the tool within the context of our EU Horizon project. There are quite a large number of RM tools available in the market, not all of which would fit the peculiarity of our project needs. We then relied on the expertise and experience of the consortium stakeholders to establish the criteria for the selection and evaluation of candidate tools to ensure that the necessary criteria required to meet the project demands were sufficiently satisfied. To minimize the potential reactivity threat, where one evaluator's judgment could influence another, each team installed the assigned tool locally and extensively assessed it over a period of three months. This allows for teams' independent judgment within a reasonable period of observation based on the established criteria. However, we acknowledge the limited generalizability of our findings due to the relatively small sample of RM tools considered, the number of evaluators involved, and the specific evaluation context. Despite these and other limitations, our evaluation provides valuable insights into our project's specific needs and contributes to the understanding of RM tools' capabilities in the context of AI-based application development.

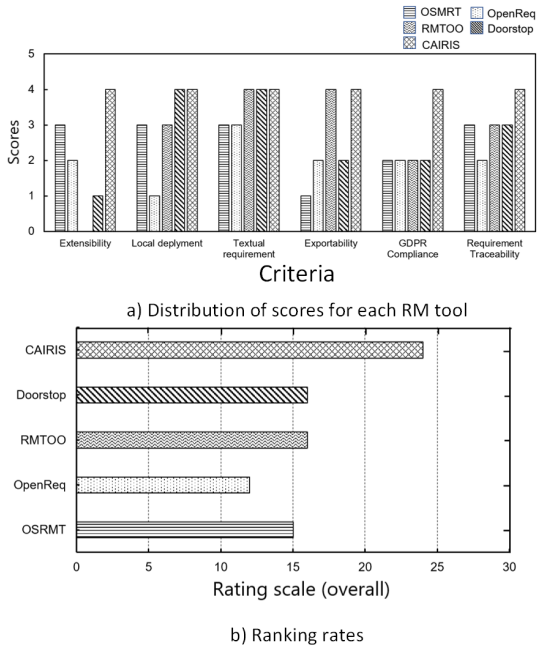

Fig. 3: Evaluation of tools based on defined criteria

V. Quantitative Analysis of the Tools

Next, a quantitative analysis is conducted to quantify each relative aspect that conforms the overall analysis criteria.

Experimental setup: We conducted a survey that captures the evaluation of each team with its respective assigned tool. We utilized a questionnaire as the main instrument for systematically collecting evaluation information about the specified criteria. We designed our questionnaire following the standard requirement evaluation questionnaire as presented in [33], [34], which also follows the ISO/IEC TR 24766:2009 standard, making emphasis in the key aspects defined in our criteria. Five-point Likert-like Scale response (5: Strongly Agree, 4: Agree, 3: Cannot say, 2: Disagree, 1: Strongly Disagree) [35] was provided to quantify the questions. Examples of representative questions include, *Can we extend the tool based on the documentation and guidelines?*; *Please rank the tool based on the "Ease of Local Deployment criterion."* and *Is it easy to deploy the tool on our local servers?*.

Results: Although most of the tools that we have considered were open-sourced, we can observe from Figure 4(a) that about 60% of tools have inadequate documentation or have poor guidelines that result in a low score of ease of extensibility. Moreover, as shown in Figure 4(b), Local Deployment has mostly received an agreement of 80% among partners, but only OpenReq did not comply with this feature, resulting in a Disagreement of 20%. It is also observable that traceability was among the features that were considered in most of the

tools (40% Agree and 20% Strongly Agree, see Figure 4(c)). It is also worth noting that RM tools require more attention to clarify how the tool adheres to the GDPR principles since the evaluators were not able to analyze this aspect clearly (80% Neutral - see Figure 4(d)).

Figure 3(a) shows the distribution of scores by RM tools based on predefined criteria. The figure confirms the results of Figure 4 and shows how features are ranked for each tool. From Figure 3(b), it is also observable that CAIRIS achieved the highest scores of analyzed criteria and, most notably, its compliance with GDPR because of its rich documentation and elaboration on GDPR principles. The tools that come next in rankings are Doorstop and RMTOO. The lower rate to RMTOO and Doorstop was mainly because of the partners' Disagreement with their ease of extensibility and GDPR compliance, resulting from poor documentation and interfaces. Accordingly, as shown in the figure, OpenReq is the tool that received the least score, and it is because of the lack of maintenance since 2019. It seems that the deployment of the tool on local premises was not easy because of outdated implementation and was not recommended for further usage in AI-based application requirement management.Taken together, our results suggest that CAIRIS provides the most optimal support for tracking and monitoring the requirements of AI-based software.

VI. Qualitative analysis of the tools

Each team provided first a qualitative analysis of the assigned tool. This analysis contained a description of the tool and qualitative information regarding how the tool fulfilled the established criteria.

(a) OSRMT (Team 1): The open-source requirements management tool (OSRMT) is an open-source requirement management tool with GNU General Public License version 2.0 (GPLv2). It offers full capabilities for defining and managing requirements for software development. It can be installed as a single-user desktop application or a multiple-user application with a centralized server.

- **Extensibility:** OSMRT is available free and open to any user. The tool's license allows users to use the tool for any purpose, access the source code and modify it for specific use. This avails OSMRT with a large community of users to collaborate and contribute towards its development and potential for wider adoption.
- **Traceability:** The tool offers traceability capabilities that enable users to create relationships between artifacts and requirements allowing for requirement analysis and change management. Stakeholders can visualize the traceabilities to aid their understanding and decision-making.
- **GDPR compliant:** Information about the tool's compliance with General Data Protection Regulation (GDPR) is not available in its GitHub repository.
- **Textual requirement:** OSRMT supports textual requirements in various forms. Users have the ability to create and manage five fundamental artifacts as baseline entities: Features, Requirements, Design Modules, Implementations, and Test Cases. These artifacts serve as essential components within the requirements management process, enabling effective organization and traceability of the software development project.
- **Exportability:** OSMRT allows the export of artifacts to XML only. However, exporting to another format such as Excel/CSV, or PDF is not possible.

(b) RMTOO (Team 2): RMTOO is a requirement management tool that is implemented in Python and designed to fully support the Linux operating system. The tool utilizes a command line interface, which consolidates input and output operations within a single environment, thus enhancing the efficiency of requirements handling. While RMTOO primarily caters to Linux, its implementation in a computer-independent programming language also enables its usage on other operating systems. Local installation can be accomplished by employing a virtual machine along with the necessary dependencies, such as Python version 3.5 and above, Latex, grahpviz, and gnuplot [27].

- **Extensibility:** RMTOO is available to the public for use under the GNU General Public License version 3.0 (GPLv3) license, an improved version of GPLv2. This implies that the license is not restricted only to the codes but also extends to any hardware or related works that use the source code of RMTOO. However, the compatibility of RMTOO is relatively strict due to its license as GPLv3 adheres to stricter compatibility guidelines which have an implication on the extensibility of the tool's source code to other projects. This necessitates that the source code can only be combined with projects with similar licenses i.e. GPLv3 or another compatible license. Thereby restricting the options for combining RMTOO with other projects that are without compatible licenses.
- **Traceability:** Linkage between requirement artifacts is visually represented using various colors in RMTOO. In RMTOO, requirement artifacts are distinctly identified with unique names and connections are color-coded, providing a visual form of traceability. However, the specifics regarding how RMTOO implements and facilitates traceability cannot be found in its repository [27].
- **GDPR compliant:** The available documentation and repository for RMTOO do not provide explicit information concerning the privacy and security measures implemented to protect user data. As a result, the tool's compliance with the GDPR cannot be verified or confirmed due to the absence of specific details regarding data privacy practices.
- **Textual requirement:** RMTOO is a comprehensive text-based requirement management tool, enabling users to seamlessly create requirement artifacts using text prompts throughout the entire process. The tool empowers users to efficiently generate requirement artifacts by utilizing text-based inputs, facilitating a streamlined and intuitive experience for requirement creation. Through its text-

based interface, RMTOO ensures that users can effortlessly capture, organize, and manage requirements using textual representations, ensuring clarity and consistency throughout the requirement management lifecycle.
- **Exportability:** RMTOO offers versatile output capabilities, allowing users to generate output in multiple file formats, including PDF, HTML, and visualization formats. Users can leverage this flexibility to select the format that best suits their needs for sharing and presenting requirements information. Moreover, RMTOO facilitates seamless export of the generated output into any of the supported formats, allowing users to utilize the output in their preferred format for further analysis, distribution, or documentation purposes.

(c) OpenReq (Team 3): OpenReq requirement management tools aim to utilize modern recommender algorithms to develop intelligent recommendation and decision technologies that support different phases of requirement engineering such as elicitation, specification, analysis, management, and negotiation. It proposes to function as a support tool that relies heavily on artificial intelligence (AI) for driving innovative and efficient requirement management. However, the tool is at the moment still a prototype that is yet to evolve into a full fledge finished RM tool.

- **Extensibility:** OpenReq is an open-source platform, which means it is freely available and can be customized and extended to meet specific project needs. The open-source nature of the platform encourages community contributions, fosters innovation, and allows for continuous improvement over time.
- **Traceability:** Information about the traceability functionality of Open Req is not available in the existing repository and demo [36]
- **GDPR compliant:** The current prototype has no data management and protection information.
- **Textual requirement:** OpenReq possess detailed and well-developed textual capabilities that enable users to create, organize, and manage requirement as well as related artifacts. In addition, the textual capabilities of the tool support collaboration among stakeholders and enable the recommender algorithms model to learn and decision-making.
- **Exportability:** The tool supports the generation of PDF output and visualization. However, the exportability of output is unknown at the moment.

(d) Doorstop (Team 4): Doorstop is essentially a Python library that helps store and manage textual files with source code for requirement management control. It requires Python 3.7+ and a version control system for requirements storage. The library can be installed locally using the pip function in the terminal and could be accessed via the terminal or imported in Python script via the import command. The source code is available on GitHub [37].

- **Extensibility:** Doorstop is distributed under the GNU Lesser General Public License (LGPLv3), which implies a "weak copyleft" license. This designation indicates that users are permitted to utilize, distribute, and modify the library's code under this license. However, users are obligated to include a complete copy of the license and the original copyright notice. In the case of derivative works, the original source code must be included or made accessible, and the derived work must be licensed under the same LGPLv3 license and not previous versions [37].
- **Traceability:** Every Doorstop file is stored inside the version control repository. Every file is assigned a unique name sequentially numbered, allowing easier linking and historical review.
- **GDPR compliant:** Doorstop prioritizes security in its available documentation. However, the documentation did not provide details about compliance with GDPR.
- **Textual requirement:** Doorstop facilitates the use of human-readable text files, for instance, YAML files, which can be easily interpreted by individuals and accessed using standard text editors. Furthermore, these text files can be parsed effortlessly using Python libraries.
- **Exportability:** Requirement artifacts can be exported for editing and exchange with other systems in different formats such as YAML (.yml), Comma-separated Values (.csv), Tab-separated Values (.tsv), and Microsoft Excel file (.xlsx). It is also possible to create requirements using any of the formats.

(e) CAIRIS (Team 5): CAIRIS (Computer Aided Integration of Requirement and Information Security) is an open-source platform for eliciting, specifying, and validating secure and usable systems. CAIRIS was developed by Shamal Faily to aid designers in integrating security and usability requirements during applications development [30]. CAIRIS is compatible with mainstream operating systems, Windows and Linux, and installable through GitHub, Virtualbox, and Vagrant.

- **Extensibility:** CAIRIS is available under the GNU Affero General Public License (AGPL). Under this license, the tool is modifiable and distributable by users. CAIRIS source codes are freely accessible. Users can collaboratively contribute to development. Being an open-source tool, the source codes can be reviewed, modified, improved, and regularly updated by a large community of developers and users. Developers and users can build on the existing functionalities of CAIRIS and extend them to suit their distinct needs.
- **Local Installation:** CAIRIS can be installed on local servers, desktops, or laptops and supports various operating systems. Users can install CAIRIS locally through GitHub, Docker, and Vagrant. Multiple users can install CAIRIS locally in different locations and can collaborate on the same project.
- **Traceability:** In CAIRIS, traceability is automated. CAIRIS employs the IRIS meta-model to automate traceability and model relationships between elements in the requirement process. This automated traceability feature

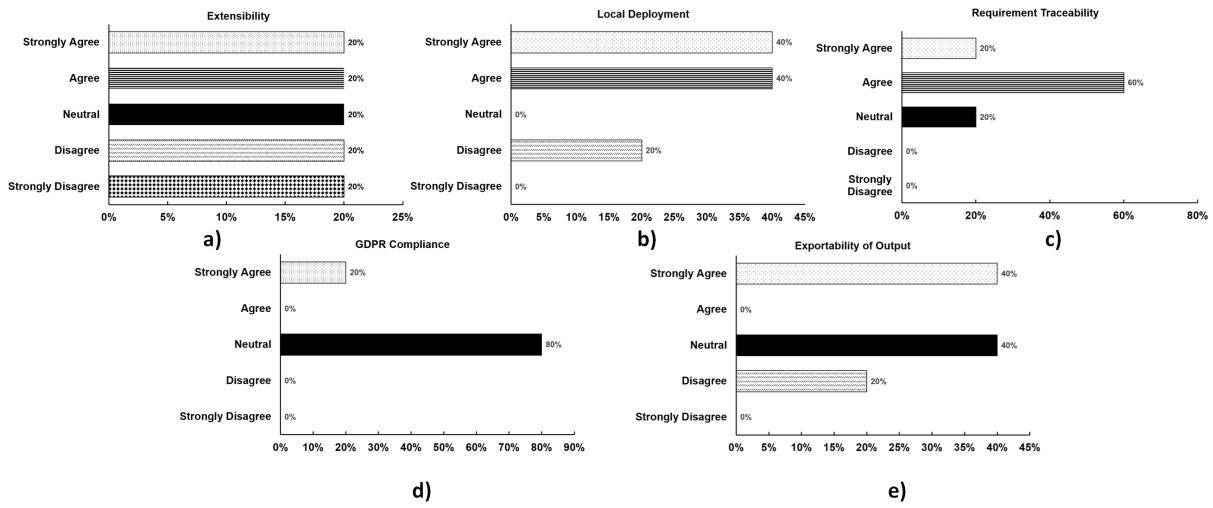

Fig. 4: An overview of criteria evaluation among RM tools

simplifies requirement management by allowing users to identify dependencies and assess the impact of changes on the application as a whole.

– **GDPR compliant:** Compliance with data protection law and privacy principles are important aspects of the CAIRIS requirement management tool. The developers of CAIRIS have incorporated model validation checks for GDPR compliance into CAIRIS. They defined three new types of roles to CAIRIS (Data Controller, Data Processor, and Data Subject) and steps for introducing personal data assets into a CAIRIS model.
– **Textual requirement:** In CAIRIS, text can be created, modified, and managed through the tool's interface. CAIRIS' text requirements are stored in the database, allowing for traceability between text requirements and other elements, which helps ensure that the application being developed meets identified needs and expectations.
– **Exportability:**
CAIRIS has export capabilities that enable users to export project data in multiple formats like CSV, JSON, PDF, and XML. In addition, users can also export specific reports like risk assessments, threats, and vulnerabilities and share data with external stakeholders. The export capabilities also serve as means for users to maintain a backup of their data in a format that can be easily imported to CAIRIS.

VII. Guidelines and Recommendations

In the light of our analysis, we next define some guidelines for the selection of tools and provide some general recommendations regarding the expected functionality that new versions of the tools must include for modern software.

Extendability: Most of the RM tools that comply with the guidelines and regulations about building AI software are also open source. While their source code is available online, extending the tool with certain functionality may require low-level programming instrumentation and re-compilation of the source code. This may be a difficult task that can potentially increase the elicitation process of requirements. Besides this, instrumenting code may also become a security breach in the development life cycle.

Supplementary features: Vulnerabilities are weaknesses that can be exploited to compromise the security of a system. AI-based systems are susceptible to different attacks. Hence, the prioritization of risk and vulnerability assessment features for RM tools dedicated to AI-based systems is required. The objective of vulnerability assessment during system development is to facilitate risk management. The process of addressing the vulnerabilities entails **identifying, assessing, and prioritizing potential security vulnerabilities** [38]. The process can be challenging for developers. However, deploying RM tools with risk and vulnerability assessment capabilities can strengthen and adequately manage the process as part of the requirement management process during development. For instance, RM tools such as CAIRIS may be equipped with in-built functionality that allows users to create a vulnerability form detailing the name, description, type of vulnerability, the likelihood of occurrence, impact level, and the exposed asset. Another key supplementary functionality that RM tools could improve is concurrent management [39], [40]. Indeed, RM tools for concurrent or real-time editing by multiple users working on the same can aid in tuning requirements in collaborative projects. High resilience to concurrent changes in requirements can ensure that any changes one user makes are automatically reflected and updated in real-time across all user interfaces, irrespective of geographical location.

Maintainability: Several open-source tools lack detailed documentation, increasing their learning curve. RM tools with minimal (or non-existent) documentation can delay the process of defining requirements as the development team first needs to

study in detail the functionality of the tool and assess whether it is suitable for the project or not. RM tools that handle the requirements of modern applications must clearly define the level of support to handle AI specifications.

Version control: AI-based applications and systems development processes differ significantly from classical application development due to a series of iterative activities and tuning involved in model training for achieving stable and optimal model performance. It is essential to capture and track changes in metadata, data, parameters, configuration, etc, that transpire throughout the development to monitor different versions of model performance that are observed. The inability to log, track and trace these changes using a requirement tool can result in costly consequences. Therefore, the availability of version control capability is crucial when selecting requirement management tools for any AI-based developmental projects.

Data management: Meta-data defines the attributes available in the dataset and it is expected that RM tools can handle such descriptor representation for the dataset. Considering that AI-based systems rely heavily on big amounts of data, it is crucial to consider also whether the changes on raw data, e.g., new data contributions, data format; can be also tracked by the RM tools. Dataset version has a direct link to model version and its performance.

Deployment: Besides having specific characteristics, it is also important to re-assess the functionalities of the RM tools once is deployed for its usage. Indeed, it is possible that vendors providing the underlying infrastructure to host the tool do not comply with certain regulations, invalidating completely the usage of the tool. For instance, a private cloud vendor could avoid specifying how the replication process is handle with their cloud technology. For instance, replication just in the same location (Europe) or to different locations (EU to US).

VIII. Discussion

Room for improvement: While our work provides several insights on selecting a RM tool for handling AI-based requirements, our work is limited in the context of a single project. Thus, we are interested on replicating our study to other projects, such that it is possible to generalize our findings. Besides this, there are several factors that may impact this type of studies. For instance, the number of teams in a project, and the amount of resources available to explore the suitability of a RM tool.

Pre-defined evaluation frameworks: As demonstrated in our work, selecting a RM tool for AI-based solution development can pose challenges due to the plethora of available options which consumes a significant amount of time. However, employing evaluation frameworks like ISO/IEC TR 24766:2009 and the International Council on Systems Engineering (INCOSE) list can potentially assist in navigating the selection process by offering a comprehensive list of features and facilitating the comparisons of these features across different tools. It is important to exercise caution when utilizing these frameworks, as they serve as a guide and the listed features should still be evaluated based on the specific requirements of the development project.

Stakeholder challenges: The complexities involved in fulfilling stakeholders' diverse expectations and preferences can make the selection of RM tools difficult. When developing an AI-based solution, there are different types of stakeholders to consider, each with unique needs and priorities. For instance, stakeholders' expectations could vary regarding the tool functionalities, user experiences, traceability, report generation, customization, resource sharing, etc. This diversity makes it difficult to align the capabilities of the various tools to consider with the stakeholders' expectations. However, effective engagement of stakeholders during the tool selection process can overcome the challenges.

Tool Integration and compatibility: The requirement management process is not a standalone process in the AI-based product development processes and other activities. Integrating the RM tool into existing processes and systems is essential for effectively managing requirements. Analyzing the compatibility of the requirement management tool can ensure seamless integration of the tool into existing systems. Similarly, the requirement management process should integrate easily with existing procedures to facilitate efficient exchange of requirements data, artifacts, and information among systems and stakeholders. Hence, considering the fitness of tools in the light of their integration and compatibility with existing processes and systems can influence choice-making when selecting a requirement management tool.

Lessons learned: Several tools in the market are designed in line with the standard capabilities to handle **ISO/IEC TR 24766:2009** requirement definitions. Our assessment of RM tools for AI-based system development projects reveals that not all RM tools can effectively support such development due to some of the unique requirements of the development, for instance, data privacy and transparency requirements. Managing the requirements thus requires specific capabilities that some RM tools lack. Moreover, while standards and regulations exist to establish the standard capabilities of RM tools, the standard needs continuous review and update to adapt to the emerging reality of the integration of AI into development projects. In addition, projects involving different stakeholders, such as technical and socio-technical experts, require to consider selecting an RM tool that can handle the definition of requirements from specific domain angles and terminology.

IX. Summary and Conclusions

In this paper, we presented a rigorous qualitative and quantitative analysis of different requirement management tools. The goal of the analysis was to evaluate whether existing requirement management tools can cope with the demands of tracking and monitoring requirements for AI-based applications, which are subject to specific characteristics imposed by regulatory entities. The analysis is conducted in

the context of a consortium that consists of several academic and industry partners. By applying the COTS method, five different tools were assessed, each by one individual partner from the consortium. Our results suggest that new regulations make it difficult to find a requirement management tool for developing AI-based software. We also share our lessons learned and experiences from selecting requirement tools that can be used in team-based consortium projects.

Acknowledgment

This research is part of SPATIAL project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No.101021808.

References

[1] H. Muccini *et al.*, "Software architecture for ml-based systems: what exists and what lies ahead," in *IEEE/ACM 1st Workshop on AI Engineering-Software Engineering for AI (WAIN)*. IEEE, 2021, pp. 121–128.

[2] R. Shah *et al.*, "Iot and ai in healthcare: A systematic literature review." *Issues in Information Systems*, vol. 19, no. 3, 2018.

[3] I. H. Sarker *et al.*, "Ai-driven cybersecurity: an overview, security intelligence modeling and research directions," *SN Computer Science*, vol. 2, pp. 1–18, 2021.

[4] A. Haydari *et al.*, "Deep reinforcement learning for intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst*, vol. 23, no. 1, pp. 11–32, 2020.

[5] A. Vogelsang *et al.*, "Requirements engineering for machine learning: Perspectives from data scientists," in *IEEE 27th International Requirements Engineering Conference Workshops (REW)*. IEEE, 2019, pp. 245–251.

[6] K. Ahmad *et al.*, "What's up with requirements engineering for artificial intelligence systems?" in *IEEE International Requirements Engineering Conference (RE)*, 2021, pp. 1–12.

[7] L. M. Cysneiros *et al.*, "Software transparency as a key requirement for self-driving cars," in *IEEE 26th international requirements engineering conference (RE)*, 2018, pp. 382–387.

[8] J. M. Wing, "Trustworthy ai," *Communications of the ACM*, vol. 64, no. 10, pp. 64–71, 2021.

[9] F. Lagioia *et al.*, "The impact of the general data protection regulation (gdpr) on artificial intelligence," 2020.

[10] Cio.gov - executive order (eo) 13960. [Online]. Available: https://www.cio.gov/policies-and-priorities/Executive-Order-13960-AI-Use-Case-Inventories-Reference

[11] C. B. Fernandez *et al.*, "Implementing gdpr for mobile and ubiquitous computing," in *Proceedings of the 23rd Annual International Workshop on Mobile Computing Systems and Applications*, 2022, pp. 88–94.

[12] W. Shao and X. Wang, "Modeling data requirements for machine learning systems," in *IEEE 13th International Conference on Software Engineering and Service Science (ICSESS)*, 2022, pp. 97–100.

[13] D. Kaur *et al.*, "Requirements for trustworthy artificial intelligence–a review," in *Advances in Networked-Based Information Systems: The 23rd International Conference on Network-Based Information Systems (NBiS-2020) 23*. Springer, 2021, pp. 105–115.

[14] M. Hoffmann *et al.*, "Requirements for requirements management tools," in *Proceedings 12th IEEE International Requirements Engineering Conference, 2004.* IEEE, 2004, pp. 301–308.

[15] G. Kotonya and I. Sommerville, *Requirements engineering: processes and techniques*. Wiley Publishing, 1998.

[16] J. M. C. De Gea, J. Nicolás, J. L. F. Alemán, A. Toval, C. Ebert, and A. Vizcaíno, "Requirements engineering tools: Capabilities, survey and assessment," *Information and Software Technology*, vol. 54, no. 10, pp. 1142–1157, 2012.

[17] W. Abbas, W. H. Butt *et al.*, "Systematic literature review on requirement management tools," in *2022 International Conference on Emerging Trends in Smart Technologies (ICETST)*. IEEE, 2022, pp. 1–6.

[18] A. Finkelstein and W. Emmerich, "The future of requirements management tools." Oesterreichische Computer Gesellschaft (Austrian Computer Society), 2000.

[19] M. Kelanti, J. Hyysalo, P. Kuvaja, M. Oivo, and A. Välimäki, "A case study of requirements management: Toward transparency in requirements management tools," in *Proceedings of the eighth international conference on software engineering advances (ICSEA 2013)*, 2013, pp. 597–604.

[20] A.-R. Ottun, P. C. Mane, Z. Yin, S. Paul, M. Liyanage, J. Pridmore, A. Y. Ding, R. Sharma, P. Nurmi, and H. Flores, "Social-aware federated learning: Challenges and opportunities in collaborative data training," *IEEE Internet Computing*, 2022.

[21] K. Hjerppe, J. Ruohonen, and V. Leppänen, "The general data protection regulation: requirements, architectures, and constraints," in *IEEE 27th International Requirements Engineering Conference (RE)*, 2019, pp. 265–275.

[22] A. Alhazmi and N. AG Arachchilage, "A serious game design framework for software developers to put gdpr into practice," in *Proceedings of the 16th International Conference on Availability, Reliability and Security*, 2021, pp. 1–6.

[23] H. Li, L. Yu, and W. He, "The impact of gdpr on global technology development," pp. 1–6, 2019.

[24] H.-M. Heyn, E. Knauss, A. P. Muhammad, O. Eriksson, J. Linder, P. Subbiah, S. K. Pradhan, and S. Tungal, "Requirement engineering challenges for ai-intense systems development," in *IEEE/ACM 1st Workshop on AI Engineering-Software Engineering for AI (WAIN)*, 2021, pp. 89–96.

[25] ISO/IEC, "Information technology – systems and software engineering – guide for requirements engineering tool capabilities," 2009, first ed. 2009.

[26] A. Smith, "Open source requirements management tool,," https://github.com/osrmt/osrmt, 2019.

[27] A. Florath, "rmtoo – requirements management tool," https://github.com/florath/rmtoo, 2020.

[28] A. Felfernig *et al.*, "Openreq: recommender systems in requirements engineering," in *Proceedings of the Workshop Papers of I-Know 2017: Co-Located with International Conference on Knowledge Technologies and Data-Driven Business 2017 (I-Know 2017): Graz, Austria, October 11-12, 2017*, 2017, pp. 1–4.

[29] J. Browning and R. Adams, "Doorstop: text-based requirements management using version control," 2014.

[30] S. Faily and D. Ki-Aries, "Usable and secure requirements engineering with cairis," in *2019 IEEE 27th International Requirements Engineering Conference (RE)*. IEEE, 2019, pp. 502–503.

[31] O. C. Gotel and C. Finkelstein, "An analysis of the requirements traceability problem," in *Proceedings of IEEE international conference on requirements engineering*. IEEE, 1994, pp. 94–101.

[32] A. Shah *et al.*, "An evaluation of software requirements tools," in *Eighth International Conference on Intelligent Computing and Information Systems (ICICIS)*, 2017, pp. 278–283.

[33] d. G. J. M. Carrillo *et al.*, "Commonalities and differences between requirements engineering tools: A quantitative approach," *Computer Science and Information Systems*, vol. 12, no. 1, pp. 257–288, 2015.

[34] Commonalities and differences between requirements engineering tools: A quantitative approach. Accessed on 17 November 2023. [Online]. Available: https://www.um.es/giisw/EN/re-tools-survey/part2.pdf

[35] T. Nemoto and D. Beglar, "Likert-scale questionnaires," in *JALT 2013 conference proceedings*, 2014, pp. 1–8.

[36] V. Biryuk, "Openreq," https://https://github.com/orgs/OpenReqEU/repositories?page=2&type=all, 2019.

[37] J. Browning and R. Adams, "Doorstop," https://github.com/doorstop-dev/doorstop, 2014.

[38] S. Faily and S. Faily, "Usable and secure software design: The state-of-the-art," *Designing Usable and Secure Software with IRIS and CAIRIS*, pp. 9–53, 2018.

[39] D. Damian *et al.*, "Awareness meets requirements management: awareness needs in global software development," in *Proc. of the Int'l Workshop on Global Software Development, International Conference on Software Engineering (ICSE 2003)*, 2003.

[40] M.-A. D. Storey *et al.*, "On the use of visualization to support awareness of human activities in software development: a survey and a framework," in *Proceedings of the ACM symposium on Software visualization*, 2005, pp. 193–202.