Science of Computer Programming 232 (2024) 103047

Contents lists available at ScienceDirect

Science of Computer Programming

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

An iterative approach for model-based requirements engineering in large collaborative projects: A detailed experience report

Andrey Sadovykh [a,*], Bilal Said [a,*], Dragos Truscan [b,*], Hugo Bruneliere [c,*]

[a] *SOFTEAM, Docaposte - Paris & Nantes, France*
[b] *Abo Akademi University - Turku, Finland*
[c] *IMT Atlantique, LS2N (UMR CNRS 6004) - Nantes, France*

A R T I C L E   I N F O

Keywords:
Requirements engineering
Model-based engineering
Collaborative projects
Experience report

A B S T R A C T

In this paper, we report on our 7 years of practical experience designing, developing, deploying, using, and evolving an iterative Model-based Requirements Engineering (MBRE) approach and language in the context of five large European collaborative projects providing complex software-intensive solutions. Based on significant data sets collected both during project execution and via surveys realized afterward, we demonstrate that such a model-based approach can bring interesting benefits in terms of scalability (e.g., a large number of handled requirements), heterogeneity (e.g., partners with different types of RE background), adaptability and extensibility (e.g., to various project's needs), traceability (e.g., from the requirements to the software components), automation (e.g., documentation generation), consistency and quality (e.g., central model), and usefulness or usability (e.g., actual deployment and practical use). Along the way, we illustrate the application of our MBRE approach and language with concrete elements from these several European collaborative projects. More broadly, we discuss the general benefits and current limitations of using such a model-based approach and corresponding language, as well as the related lessons we learned during these past years.

1. Introduction

In many European countries, collaborative projects involving academic and industrial partners are a preferential way of implementing ambitious Research and Innovation Actions (RIAs) [1,2]. They also foster international collaborations and develop long-term partnerships between organizations from various countries. For example, the European Commission has several funding agencies (ECSEL, Horizon, ITEA, etc.) with various programs targeting different societal, economic, and scientific grand challenges [3]. As a result, collaborative projects are one of the primary sources of innovation in Europe [2]. The average number of involved organizations goes from 4.69 in Horizon 2020 [4] to 30 and 40 in ECSEL (now KDT) projects [5], even exceeding 100 organizations in some large ECSEL/KDT projects [6].

A key element of such European research projects is the heterogeneity and complementarity of the project's participants. The participating organizations can come from different application domains (e.g., railway, avionics, telecommunications, manufacturing), have different sizes (e.g., from small and medium enterprises to large industrial groups), have various maturity levels, or come

* Corresponding authors.
E-mail addresses: andrey.sadovykh@softeam.fr (A. Sadovykh), bilal.said@softeam.fr (B. Said), dragos.truscan@abo.fi (D. Truscan), hugo.bruneliere@imt-atlantique.fr (H. Bruneliere).

https://doi.org/10.1016/j.scico.2023.103047
Received 17 March 2023; Received in revised form 22 September 2023; Accepted 11 October 2023

Available online 23 October 2023

with different kinds of research backgrounds. They must work together to achieve a set of shared R&D goals, usually validated via several case studies that typically serve as a common sandbox for experimenting on newly designed and developed technologies. The objective is to provide evidence to the European Commission and the European community of the benefits and drawbacks, both scientific and economical, that the developed innovative technologies can offer.

In most cases, such collaborative research projects are driven by the needs of the industry. Therefore, the entire project activity is focused on identifying those needs and providing corresponding technologies to address them. However, the diversity and number of partners [7] amplify challenges related to 1) the elicitation of the needs from the industrial case study providers and 2) the identification of concrete solutions to be provided during the project (lasting typically three years or more), and 3) the creation of a roadmap for the development of these solutions. When such challenges are not addressed properly, they can negatively influence the outcomes of the project [8].

The main expected results in the Software and Systems Engineering area projects are generally large and complex integrated frameworks or tool sets. To allow for their actual design, development, and deployment, it is thus fundamental to support and manage as efficiently as possible the corresponding Requirements Engineering processes [9].

Requirements Engineering (RE) is the process of identifying, describing, using, and maintaining requirements within engineering processes [9,10]. There is a widely accepted consensus on the main steps of RE processes [11,12]: 1) requirements elicitation, 2) requirements analysis, 3) requirements specification, and 4) requirements validation. Requirement management is a transverse activity covering all these steps and ensuring their smooth execution. Even if requirements have been used for a long time in different disciplines, the RE advent mostly coincides with the software expansion in the 90s [13]. This resulted in the progressive growth of an active research community [14,15].

In large collaborative projects, the first major **challenge #1** is the elicitation, analysis, and specification of the requirements supporting the needs of the industrial case study providers, i.e., the case study requirements. This is not trivial since it requires the collaboration of all the partners, each having different application domains and technical backgrounds. Thus, having a centralized and accurately described mapping between case study requirements, on the one hand, and the framework and tool components requirements, on the other hand, can allow the technical coordination team of the project to better track the progress of the case studies development, spot further needs for technical solutions, and mitigate the risks. The second major **challenge #2** is to validate the requirements and create a roadmap for framework development by collecting, in addition to the previously gathered information, the development plans for individual tool components. This allows all partners to be aware of when different features of the framework will be implemented. This way, case study providers know when these tools can be evaluated against their case studies. In addition, having such a roadmap allows the technical coordination team to plan better and produce the different deliverables, demonstrations, and management or dissemination events. Appropriate RE approaches and tools are needed to address these two major challenges to maximize the benefits and achieve good technical results.

In this article, we propose a solution that combines Model-based Requirements Engineering (MBRE) with iterative project management practices to improve the support for large and diverse collaborative projects. Our main contributions are the following:

1. A *MBRE approach* for collaborative projects supporting the different stakeholders' activities in the project concerning both functional and non-functional requirements as well as data requirements;
2. An underlying *iterative RE process* with microtasks for improving both the collaborative collection of requirements (**challenge #1**) and their further analysis and exploitation (**challenge #2**);
3. A *dedicated modeling language* for RE, implemented in Modelio tool [16] and allowing to realize the proposed approach and process in practice;
4. An *evaluation of the approach and process* via quantitative and qualitative data from the large collaborative projects where the solution was applied;
5. A discussion on the main *lessons learned and corresponding identified challenges* as far as MBRE and, more generally, RE are concerned.

From an industrial perspective, the proposed solution is rooted in acknowledged software and system modeling standards such as UML [17] and SysML [18] to foster genericity, reusability, and interoperability, as well as by the recommendations of using iterative development in model-driven engineering practices [19,20]. The solution is also inspired by the microtask programming concept used in the context of software engineering crowd-sourcing. Finally, it is also partially inspired by the European Space Agency standard terminology and structure for RE documents [21].

The solution was incrementally designed, developed, refined, and applied in five large European projects over more than seven years: H2020 *DataBio*[1] 2017-2019 [22], ITEA3 *REVaMP2*[2] 2016-2019 [23], ECSEL *MegaM@Rt2*[3] 2017-2020 [24], H2020 *VeriDevOps*[4] 2020-2023 [25], and ECSEL *AIDOaRt*[5] 2021-2024 [26], providing various complex software solutions (frameworks, integrated tool sets, etc.). However, we have already faced similar challenges in coordinating RE processes and developments in more than fifteen similar European collaborative research projects.

1 https://cordis.europa.eu/project/id/732064.
2 http://www.revamp2-project.eu/.
3 https://megamart2-ecsel.eu/.
4 https://www.veridevops.eu/.
5 https://www.aidoart.eu/.

An initial version of our solution was published a couple of years ago [27]. Since then, we significantly extended it with the concept of *microtasks* for a facilitated iterative collaboration and more uniformity in the produced requirements. We also added the modeling of *requirements on training and validation data sets* and obtained *improved evaluation results* based on new data collected from recent projects. Overall, we intend to show that modeling in a RE context can bring interesting benefits in terms of:

- scalability, e.g., a large number of handled requirements,
- heterogeneity, e.g., partners with different profiles and types of RE background,
- adaptability and extensibility, e.g., new project configurations or produced artifacts,
- traceability, e.g., from the initial requirements to the software components,
- automation, e.g., requirement documentation generation,
- consistency and quality, e.g., of the requirements and their metadata,
- as well as general usefulness and usability from the project partners' perspective.

This article is structured as follows. Section 2 introduces the current state of the art related to MBRE, particularly for large collaborative projects. Section 3 describes the MBRE solution we propose. Section 4 evaluates this solution and demonstrates its practical application in the context of five large European collaborative projects. Section 5 discusses the lessons learned and open challenges we identified during this long-term experience. Finally, Section 6 concludes the paper.

2. State of the art

The related literature is rich concerning RE approaches and corresponding technical solutions [15]. In the following, we will revisit works that are closer to our approach from different points of view.

2.1. Requirements elicitation and specification for collaborative projects

Tool-supported requirement management approaches for projects and teams have been traditionally developed using communication systems (e.g., wikis and online forums) to increase collaboration between heterogeneous participants. For instance, this is the case of a wiki-based approach for eliciting and consolidating requirements in a relatively small-sized European project [28,29]. The approach focuses on improving requirements' quality by combining online tools and physical meetings. Notably, using a wiki to collect requirements intends to reduce the communication barrier among project partners while allowing a certain level of requirements versioning and tracking. However, the authors do not mention how the requirements are used and maintained later during the project.

The authors of [30] propose a web-based tool for managing requirements specifications collaboratively. The main focus is providing access control roles, templates, and variability modeling of requirements. Even though the concepts of the approach are defined using a domain-specific language, the tool is implemented using Google Drive and its APIs. Architecture identification and requirements validation road mapping are not addressed.

2.2. Requirements modeling approaches

Modeling has often been essential in the RE process [31]. Thus, model-based principles and techniques have already been applied to address different RE activities [32]. Indeed, providing relevant abstraction, genericity, or reusability capabilities (among others) has become more prevalent in the industry over the last two decades [33]. From a RE perspective, this resulted in acknowledged contributions such as goal modeling languages [34,35] or the ReqIF requirement modeling standard [36]. However, few proposed model-based approaches partially cover RE processes. To the best of our knowledge, none of them allows systematically addressing complete RE processes in the general case.

For example, an existing solution relies on a generic modeling framework to represent and simulate requirements independently from their context [37]. Another solution is based on a customized core requirements metamodel to support various RE processes [38]. We can also mention transformation-based approaches from goals models to design models, such as the KAOS method [39]. Furthermore, [40] provides a collaborative requirement elicitation process based on the EIA-632 standard. The approach uses UML object diagrams to model the requirements, while tool support is provided as a web application SPECJ. However, these approaches only target the requirements elicitation phase. Other approaches have been deployed only in single projects [41] or primarily focused on some particular RE aspects, such as requirements visualization, for instance [42].

2.3. Model-based collaborative approaches

Closer to our context and objective, a few model-based approaches have been proposed and used in the context of collaborative research projects to handle, at least partially, the corresponding RE processes.

For instance, Nielsen et al. [43] proposed a RE process for small EU-funded projects. They offered to have each case study provider paired with an academic partner to assist the former in eliciting and specifying the requirements. This can be very challenging in the context of large projects with many partners, potentially leading to incomplete resulting requirements.

Another work introduced a visual modeling notation to help project managers proactively plan the collaboration infrastructures needed to support requirements-related activities in distributed projects [44,45]. The underlying metamodel allows one to model site locations, stakeholder roles, communication flows, critical documents, supporting tools, and repositories. However, the communication process between stakeholders is not clearly defined. Furthermore, it is not clearly explained how the business requirements analysis from multiple locations, stakeholders, domains, and products should be instrumented. The approach uses a model-based approach, is collaborative, and can be customized based on the specific needs of different projects. However, it does not target architecture identification or projects with heterogeneous partners.

In a different work, lessons learned from a large-scale research project are discussed, and various elicitation, traceability, and gap analysis techniques are identified [7]. They partially use model-based techniques for elicitation and analysis as well as for identifying the framework architecture. However, the authors did not apply this method to support traceability. Nevertheless, they also defined metrics-based validation criteria for requirements. The authors used a wiki system to manage the requirements collaboratively. They thus had to develop a dedicated tool to detect inconsistencies and perform gap analysis using a visualization toolchain based on GraphViz to better explore their requirements database with generated viewpoints such as graphs and diagrams.

A more recent work presents an empirical study using requirements modeling in large-scale agile development [46,47]. It notably introduces a set of interesting conclusions, challenges, and needs. Among them, the authors highlighted that “requirements models are useful and valuable” but creating and maintaining requirements models with high quality is challenging. They also mentioned that it was difficult to model collaboratively with existing technologies and that, many times, requirements models have to coexist with textual representations to provide better access and understanding. Therefore, their main suggestions for improvement are that 1) models should contain enough information to satisfy the needs of different stakeholders, 2) there should be means for the automated generation of artifacts from models depending on stakeholder needs, and 3) it is important, in general, to support the heterogeneity of the agile teams. These observations align with our long-term experience and the two main challenges we intend to address with our proposed solution. Furthermore, they support their approach with a homemade RE tool called T-Reqs [48]. It is a text-based Git-powered tool that allows for collaborative requirements management and offers several requirements modeling features intended to address RE challenges faced in large, collaborative, and agile projects. It also allows keeping the requirements close to the code base when hosted on the same Git repository. However, they highlight the tool’s limitations in capturing complex requirements modeling aspects, namely in supporting traceability [46].

2.4. Framework identification

Another work [49] focuses on identifying the framework architecture in a European project with heterogeneous requirements. The approach provides a set of steps to find commonalities in the requirements and to extract common requirements and the architecture components. However, the proposed approach is not model-based, tool support is not discussed, and the links between the overall architecture and the tools developed by technology providers are missing. Consequently, this makes gap analysis and road mapping more challenging to perform.

The authors of [50] propose a model-based approach for supporting requirements and architecture documentation. Their process is iterative, enabling collaboration and bridging the terminology gap between requirements engineers and system architects. However, the approach is primarily targeted at documenting design decisions, and it is not intended explicitly for large-scale collaborative projects.

2.5. Iterative requirements modeling with microtasks

The microtasks-based approach proposed in this article has been influenced by *microtask programming* [51]. In there, the software development tasks are broken down into smaller tasks that can be distributed to different development teams or individual developers to provide easy-to-follow decontextualized assignments, decrease joining barriers, and optimize the effort of allocating developers to multiple projects.

Microtask programming has been applied in the past in the context of behavioral-driven crowd-sourcing projects [52], which yielded fewer programming errors in the resulting code. However, to the best of our knowledge, the idea has not been applied to requirements engineering activities in large-scale collaborative European projects.

2.6. Summary

While reviewing the state-of-the-art and related work, we observed several challenges. The presented processes are generic and lack clarity on how to implement them effectively in real settings. One major issue is the absence of a solution for involving heterogeneous partners in the modeling work, organizing their contributions, and integrating the results. Additionally, there is a scarcity of experience-based reports that provide concrete guidelines for a process implemented, tested, and validated in several projects. Moreover, the presented solutions often heavily depend on specific tools, making them less adaptable. Furthermore, many MBRE solutions lack scalability, both in terms of tooling and organizing large teams.

3. Proposed solution

Our Model-Based Requirements Engineering (MBRE) solution comprises a comprehensive overall approach, a related process, and a dedicated modeling language, all associated with tooling support. In this section, we describe the solution while emphasizing

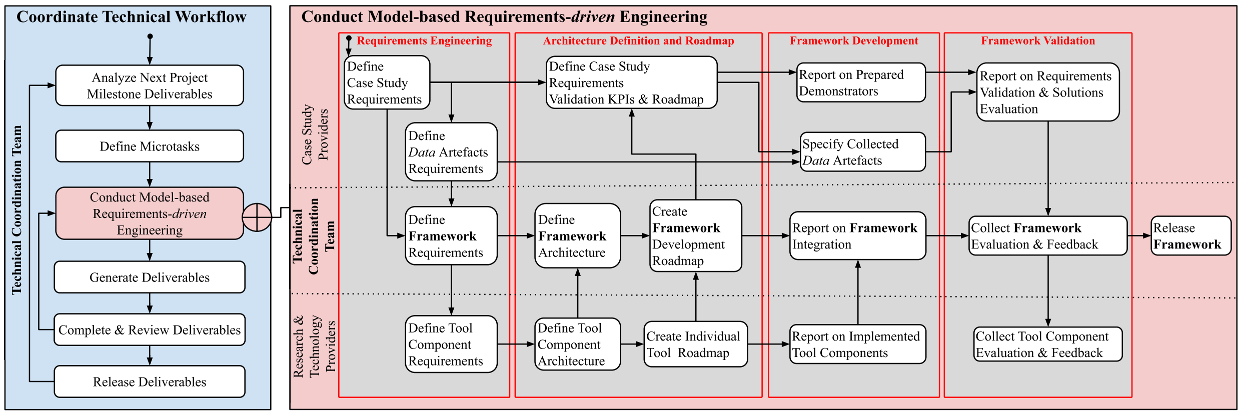

Coordinate Technical Workflow
Technical Coordination Team
Analyze Next Project Milestone Deliverables
Define Microtasks
Conduct Model-based Requirements-driven Engineering
Generate Deliverables
Complete & Review Deliverables
Release Deliverables
Conduct Model-based Requirements-driven Engineering
Requirements Engineering
Architecture Definition and Roadmap
Framework Development
Framework Validation
Case Study Providers
Technical Coordination Team
Research & Technology Providers
Define Case Study Requirements
Define Data Artefacts Requirements
Define Framework Requirements
Define Tool Component Requirements
Define Case Study Requirements Validation KPIs & Roadmap
Define Framework Architecture
Create Framework Development Roadmap
Define Tool Component Architecture
Create Individual Tool Roadmap
Report on Prepared Demonstrators
Specify Collected Data Artefacts
Report on Framework Integration
Report on Implemented Tool Components
Report on Requirements Validation & Solutions Evaluation
Collect Framework Evaluation & Feedback
Collect Tool Component Evaluation & Feedback
Release Framework

Fig. 1. Our Iterative Model-based Requirements Engineering Conceptual Approach.

the novelty compared to the initial published version [27]. This latest version integrates several improvements in terms of iterations toward the project's milestones (cf. Section 3.1). In addition, we describe the accompanying MBRE process that now relies on the use of microtasks (cf. Section 3.2). Then, we present our related MBRE language that now also captures additional important aspects such as *data requirements* (cf. Section 3.3). Finally, we present the updated implementation and related tooling supporting our overall MBRE solution (cf. Section 3.4).

3.1. An iterative model-based approach for RE

Our MBRE approach is defined to take into account the roles and activities of typical project participants in large collaborative research projects involving both academic and industrial partners. From our long-term experience in many projects of this type, both at the national level in each of our respective countries and at the international level (e.g., in Europe), we have been able to observe a common project participants: *Case Study Providers* bring a practical problem, e.g., in terms of development processes, product quality, or features, for which they are looking for innovative solutions; *Research Partners* develop new methods and prototypes and *Technology Providers* offer technological solutions that can be deployed and evaluated against the industrial case studies in collaboration with the research partners. It is then the task of a *Technical Coordination Team* to ensure a smooth collaboration between the involved parties, as well as the global achievement of the project's goals. Of course, there are different kinds of projects, and the participants' roles may vary. For example, the same entity can sometimes be both a Research Partner and a Technology Provider, or a Case Study Provider can also become a customer of a Technology Provider. However, we believe that the proposed categories of partners are adapted to cover most of the large collaborative research projects we target.

As shown in Fig. 1, our approach proposes that the technological solutions are provided in the form of a generic *framework*, which is the main artifact being developed throughout the project and produced by its participants. The framework is created following the traditional phases of the software development life cycle (SDLC), as follows:

Requirements Engineering. The framework requirements are elicited from industrial case study requirements, whereas Research and Technology Providers derive Tool Component Requirements for their individual tools to satisfy case study requirements. Furthermore, in this phase, requirements for validation data sets are identified depending on the specifics of the case studies. For example, a telecom company that manufactures routers with proprietary firmware may be interested in validating their case study by performing system failures root-cause analysis on *real-time logs* generated by the routers when put in production environments. Similarly, a research and technology provider working on critical systems verification may provide *proof obligations* data sets to be mined by other partners to help in developing ML-based proof assistants.

Architecture Definition and Roadmap. The framework aggregates *Tool Components* specific to different application domains and applicable to one or several case studies. Each tool component has a specific architecture, including the interfaces for interconnection with other tools. This notably allows different toolchains to be easily created for particular case studies. Each tool component is developed with different priorities and becomes available at different milestones from the *Tool Component Roadmap*.

Based on the individual tool component architectures and roadmaps, the technical coordination team designs the *Framework Architecture* and the *Framework Development Roadmap*. This roadmap allows the case study providers to know when different technologies will be available for evaluation and consequently to define *Case Study Requirements Validation KPIs and Roadmaps*.

Framework Development and Validation. Research and technology providers develop their individual tools as *Tool Component Implementations*. At the same time, necessary data sets are collected for validating the component implementations. When the latter become available at different project milestones, they are integrated into the framework and evaluated against the case studies. After each evaluation phase, feedback is collected and provided to relevant parties. In addition, incremental versions of the framework are released.

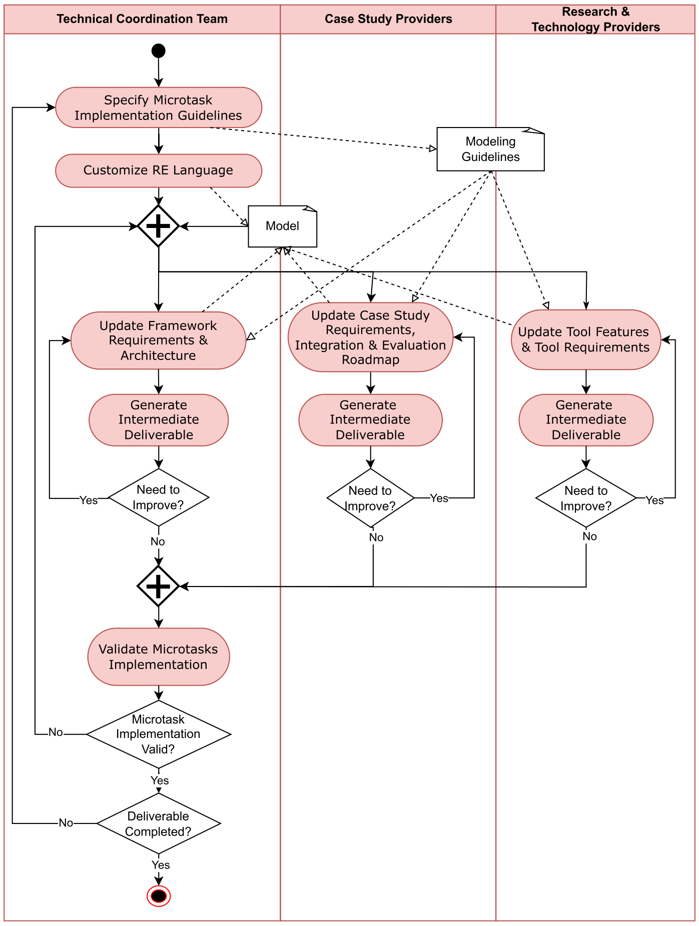

Technical Coordination Team
Case Study Providers
Research & Technology Providers
Specify Microtask Implementation Guidelines
Modeling Guidelines
Customize RE Language
Model
Update Framework Requirements & Architecture
Update Case Study Requirements, Integration & Evaluation Roadmap
Update Tool Features & Tool Requirements
Generate Intermediate Deliverable
Need to Improve?
Yes
No
Validate Microtasks Implementation
Microtask Implementation Valid?
Deliverable Completed?

Fig. 2. Iterative process of our MBRE Approach: *iterative model extensions and continuous updates timely support the project's evolution and the deliverables creation process.*

3.2. A dedicated process for RE based on microtasks

The general MBRE approach in Fig. 1 is applied continuously during the implementation of the project. In each iteration, the Technical Coordination Team analyzes the project's schedule and deliverables and defines *microtasks* for the next working period. A given microtask consists of adding new model elements and may correspond to one or more steps in the MBRE approach (Fig. 2). For example, one microtask may consist of collecting data requirements and mapping them to the various use case scenarios. Another microtask could be defined to collect additional info about the proposed solutions licenses, URLs, and expected delivery dates.

Following the process described in Fig. 2 the microtasks are specified in detail via a set of *Modeling Guidelines* and communicated and explained to partners during regular project meetings. In addition, the Technical Coordination Team will perform necessary updates to the shared *Model*, by adding, for instance, new model element types and containers as placeholders for the work and initiating a template for the deliverable.

Subsequently, all three categories of partners will collaboratively update the model according to their specific guidelines. At any moment, any of the partners may choose to generate an intermediate version of the deliverable containing all the information available in the model needed for that deliverable to check the status and completeness of the information. If more updates are needed, they will be added to the model. When all partners have finalized their contribution to the model, the Technical Coordination Team will check that all requested information is present in the model. In case more information is needed, partners are asked for more contributions. Also, if the deliverable requires further updates, microtask modeling guidelines, and the model are updated to accommodate the new information, and the process is restarted.

This workflow enables an iterative application of our MBRE approach. Its key advantage is the continuous integration of content from the partners in the requirements model, which is considered the *single source of truth* for providing coherent and up-to-date content to all the project's deliverables. The incremental metamodel extension allows adaptation according to changes in the project's initial objectives and plan, as well as its dynamic progress while meeting the deliverable deadlines. Furthermore, this iterative application through smaller microtasks allowed for a smoother learning curve of the approach, model, and tool by all the involved partners. This notably includes those who were unfamiliar at first with our MBRE approach or even with modeling in general.

Table 1
Mapping our MBRE metamodel to UML and SysML.

MBRE metamodel	UML / SysML mapping - abstract syntax
Requirement	SysML::Requirements::Requirement
RequirementsContainer	UML::Package or SysML::ModelElements::Package
ArchitecturalElement	UML::Class
Component	UML::Component
Package	UML::Package
Node	UML::Node
Interface	UML::Interface

3.3. An enriched modeling language for RE

To realize the proposed conceptual approach and to represent and share appropriately the requirements during the full RE process, we developed a dedicated modeling language for RE. The reason for developing a dedicated language, rather than directly using an existing one, is that the commonly used modeling languages, e.g., general-purpose ones, are very broad in scope. Thus, users tend to have diverse ways of specifying requirements by applying a variety of design decisions that are difficult to unify. This is an issue in our context of large collaborative projects involving partners with different backgrounds and expertise.

Thus, we decided to design and build our modeling language by following a bottom-up approach: we started by analyzing which documents (i.e., deliverables) were generally needed in terms of both requirements and architecture in large collaborative projects. To this end, we studied standard representation formats such as ESA ESS [21] and standard modeling languages such as UML [17], SysML [18]. Then, inspired by them, we designed a generic modeling language that would help in supporting and simplifying the automated generation of these documents/deliverables. This language was designed and evolved based on the needs of our requirements engineering processes. However, the language also went through tailoring and customization depending on the project in which we used it. Overall, the objective of the language is to allow the project participants to elicit, specify, and then share both the requirements and the details of corresponding architectural elements involved in our approach (cf. Section 3.1).

Fig. 3 shows the current version of the metamodel of our RE modeling language. There are different types of requirements, i.e., `ToolRequirement`, `FrameworkRequirement`, `CaseStudyRequirement`, and `DataRequirement`. We generalize all types of requirements into a generic `Requirement` element. A `CaseStudyRequirement` can also be classified w.r.t. five main DevOps activities (i.e., requirements engineering, modeling, coding, testing, and monitoring) for better requirements understanding, monitoring, and validation. To be able to manage large numbers of requirements, a `RequirementsContainer` can be used for grouping them.

On the framework side, the main element is the `ArchitecturalElement` from which architectural `Components` are derived to describe the tools implemented in the project (i.e., `ToolComponents` and the `FrameworkComponents` they support). Each component will provide an `Interface` and can consist of several `SubComponents`. The `Package` is defined to organize and group `Components`, deployment `Nodes`, and `Interfaces`. This grouping can be done based, for instance, on the case study or the purpose of the architectural element (e.g., design, testing, monitoring, etc.). An `ArchitecturalElement` may `satisfy` one or more `Requirement` elements. This allows the project partners to specify which tool components may fulfill a framework, tool, or case study requirement and which data models are associated with which data requirements. A more detailed example of the use case and tool requirements is provided in Section 3.4.3.

Compared to the initial version of the language [27], the newly added concepts notably cover data- and DevOps-related aspects. Due to the crucial role of data in large collaborative projects, we now capture `DataRequirements` for each case study as well as the `DataModel`(s) considered for handling the related data (i.e., data metamodels/languages). Moreover, we also capture to which `DevOps- Phases` (i.e., the main engineering activities) the case study requirements relate. This allows for a more precise classification of the case study requirements and, thus, a finer-grained matching with corresponding technical solutions (challenge #1). Overall, such a more refined matching allows us to better characterize both existing and potential collaborations between case study and solution providers. It also facilitates a more detailed validation of the requirements by the case study providers and the provisioning of more complete feedback to the solution provider (challenge #2).

3.4. Current implementation in modelio

The MBRE solution presented in this paper may be possibly implemented by relying on different modeling tools. However, for integration purposes, we chose to implement them in the single Modelio tool [16] developed by SOFTEAM - one of the core partners of the projects mentioned in this paper. This allowed us to benefit from extensive support during the design, development, deployment, and progressive evolution of our global MBRE solution.

In particular, our concepts map to the UML and SysML metaclasses as depicted in Table 1.

To illustrate, Fig. 4 shows an example project structure in Modelio using the previous version of the metamodel. For the Requirements level, we directly benefited from Modelio Analyst features such as tabular and diagram view, traceability matrices, and automated import from Excel. For the Architecture level, we relied on the standard implementation of UML as currently available

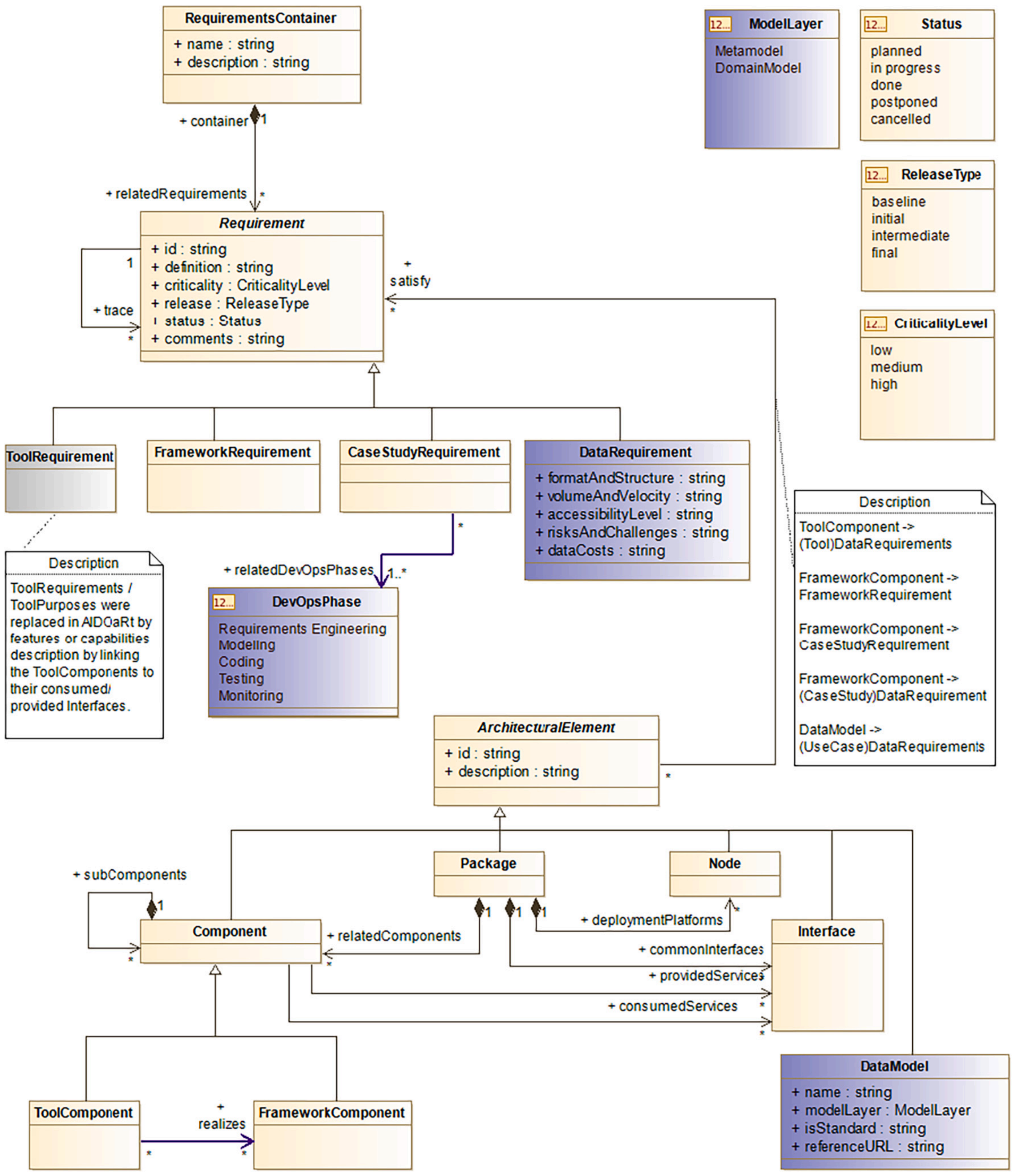
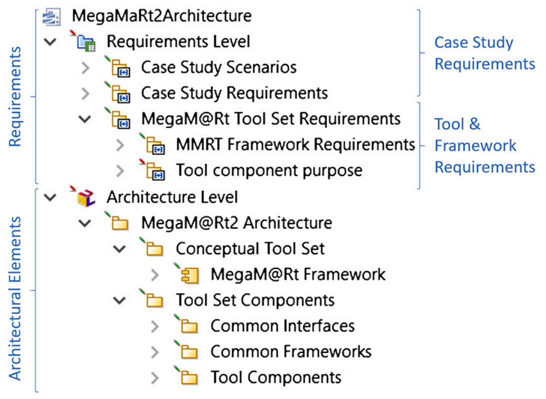


Fig. 3. Metamodel of our Extended RE Language (elements in purple depict the newly added concepts to the abstract syntax of our language). (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

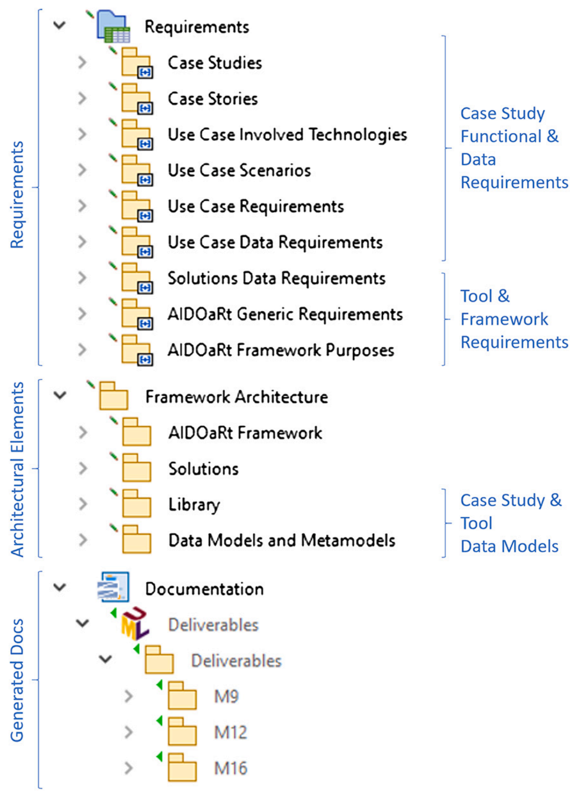
in Modelio. We refined this UML implementation to limit the use of UML to the concepts considered in our RE language. Moreover, to facilitate the initial use of the RE language, we have also built a specific template for tool components and made it available directly from the Modelio workbench. This way, the users of our RE language can benefit from clear guidelines in Modelio on what is expected from them when elaborating on the RE model for their tool components.

To complement this, Fig. 5 shows the structure of the enriched RE model in Modelio, as instantiated and tailored for the AIDOaRt project. We can observe that the model is richer in terms of requirements modeling as it now includes both `CaseStudyRequirements` and `DataRequirements`. Furthermore, the *Case Studies* container holds the high-level description of each case study, while *Case Stories* and *Use Case Scenario* containers list more detailed descriptions of the aims of each case study from an end-user point of view. These model elements were added in AIDOaRt as instances of `CaseStudyRequirement` to allow the case study providers to gradually elicit their needs. Thus, case study providers can start from a high-level textual description and end up with a structured list



MegaMaRt2Architecture
- Requirements Level
  - Case Study Scenarios
  - Case Study Requirements
  - MegaM@Rt Tool Set Requirements
    - MMRT Framework Requirements
    - Tool component purpose
- Architecture Level
  - MegaM@Rt2 Architecture
    - Conceptual Tool Set
      - MegaM@Rt Framework
    - Tool Set Components
      - Common Interfaces
      - Common Frameworks
      - Tool Components

Fig. 4. Structure of the base RE model in Modelio: Example from the MegaM@Rt2 project.


Requirements
Architectural Elements
Generated Docs
Case Study Functional & Data Requirements
Tool & Framework Requirements
Case Study & Tool Data Models

- Requirements
  - Case Studies
  - Case Stories
  - Use Case Involved Technologies
  - Use Case Scenarios
  - Use Case Requirements
  - Use Case Data Requirements
  - Solutions Data Requirements
  - AIDOaRt Generic Requirements
  - AIDOaRt Framework Purposes
- Framework Architecture
  - AIDOaRt Framework
  - Solutions
  - Library
  - Data Models and Metamodels
- Documentation
  - Deliverables
    - Deliverables
      - M9
      - M12
      - M16

Fig. 5. Structure of the enriched RE model in Modelio: Example from the AIDOaRt project.

of atomic and functional *Use Case Requirements* as well as *Use Case Data Requirements*. Finally, the fulfillment of all these requirements can be evaluated towards the end of the project with measurable KPIs. Note that these KPIs are also modeled, but we do not present them in this article for the sake of simplicity.

This last version of our model also adds new elements to the framework architecture. For example, the framework is defined in the *AIDOaRt Framework* package as a set of components and functional interfaces that aim to fulfill the case study requirements. These requirements are implemented by various research and industrial solutions specified in the *Solutions* package. For each solution, the list of capabilities can be specified in terms of operations, with input/output data defined in the *Library* package. Finally, the *Data Models and Metamodels* package contains the specification of the data models specified and listed in various case studies and solution data requirements (NB: when the data is a set of models, to be mined with AI/ML techniques, the data model is then a metamodel).

In addition, the model contains a *Documentation* package holding the various document generation templates grouped by project milestone (M9, M12, etc.). These templates are used to automatically generate the project's *Deliverables* based on the latest model content that is collaboratively and continuously updated by the project's participants (cf. Section 3.4.1).

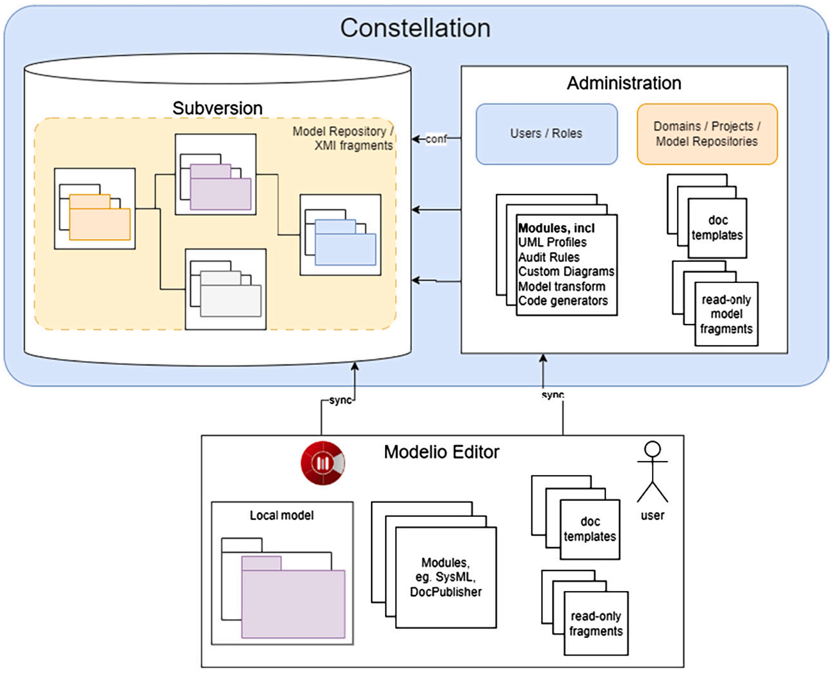

Constellation
Subversion
Model Repository / XMI fragments
conf
Administration
Users / Roles
Domains / Projects / Model Repositories
Modules, incl UML Profiles Audit Rules Custom Diagrams Model transform Code generators
doc templates
read-only model fragments
sync
sync
Modelio Editor
Local model
Modules, eg. SysML, DocPublisher
doc templates
read-only fragments
user

Fig. 6. High-level view of the Modelio and Constellation solutions for collaborative modeling support.

The model repository system named *Constellation* [53], offered by Modelio SaaS,[6] allows collaborative and concurrent model editing by all partners. It preserves the model change history throughout the whole project. This allows us to revert unwanted accidental changes, as well as to navigate back to a particular version related to a given milestone or a specific deliverable.

The Constellation employs Subversion as its model repository (Fig. 6). The model is divided into atomic fragments, facilitating concurrent modifications even at a fine granularity. The Subversion server is configured and managed by an Administration server, which handles all aspects related to modeling projects. This includes setting up Users, Roles, and their access rights based on specific Domains, Project levels, or a specific Model repository. Each project can rely on a set of modules, document generation templates, and read-only model libraries for configuration management. The User's local environment is configured through synchronization with the Administration server. In particular, our MBRE solution leverages pre-configured SysML and DocPublisher modules, along with a set of custom-made document generation templates. All these customizations are delivered to the local user's environment upon connection. We deliberately avoided creating specific UML Profiles or custom diagrams to ensure that we could rely on commonly known standard UML modeling techniques.

In addition to that, we created dedicated model containers for each partner (Fig. 7). Thus, team members of a given organization can freely edit the model elements related to their case study or solution in isolation without altering other partners' content. Model containers can be edited separately by several partners in parallel to maximize the concurrent access and update the model in a shorter time (particularly close to a tight deliverable deadline) while minimizing the conflicts. Furthermore, we enforced the approach of locking model elements before editing them. This eliminates the possibility that two members of the same partner's organization simultaneously edit the same model element.

We have deliberately minimized the common parts of the model that partners may edit. This includes the standard interfaces between the tools, such as XMI files, and deployment platforms, such as Java runtime. Those parts are typically defined only once in a specific package. Afterward, the individual model container will only reference (like "expose" or "use") those common model elements. These operations do not require any changes in the common model. In addition, the parts of the model that require multiple modifications, such as the component architecture of the framework, are done during live meetings by the technical coordination team.

Collaborative modeling may also be influenced by the adopted modeling language. Indeed, in the DataBio project, we first experimented with extending/refining the ArchiMate modeling language [54] as the high-level model representation for the architecture. However, due to the constraints imposed by the ArchiMate metamodel, collaborative modeling was quite tedious in Modelio. For example, the locking of a single element was leading to the locking of a large portion of the model, possibly impacting other users. For this reason, in the subsequent projects, we opted for extending/refining UML as it allows atomic editing of model elements.

[6] More on Modelio SaaS at https://www.modeliosoft.com/, and its Modelio open source version at https://www.modelio.org/.

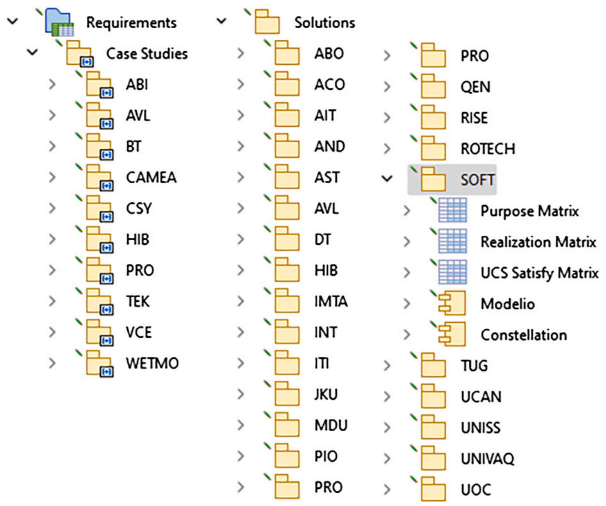


Fig. 7. The MBRE model elements are arranged by partner. Model elements of a given partner are isolated in a separate container to allow concurrent access and update while avoiding conflicts.

3.4.1. Document generation support

The Modelio Document Publisher was a major feature of Modelio to construct our MS Word document generator. Indeed, Modelio provides a simplified document template editor with a graphical interface that has pre-built functionalities for navigating the model, filtering model elements, extracting textual notes and diagrams, as well as for building sections, paragraphs, tables, and matrices. These features were particularly useful to generate important parts of our project's deliverables. This is notably the case for roadmaps, where we created tables displaying mappings between case study and tool requirements with planned delivery dates. Based on such roadmaps, the case study providers could anticipate and organize both the building of their toolchains and their validation activities. Moreover, the traceability information in the RE model was extremely important to identify the case study requirements that are not addressed by any tool component. This way, we were able to conduct a gap analysis from our RE model to better plan corrective actions.

Fig. 8 shows an excerpt from two generated deliverable documents, D3.2 [55] and D3.3 [56].[7] This example shows the cover page, meta-data, auto-generated table of contents, as well as headings, paragraphs, lists, diagrams, and tables that are generated from the MBRE model. Besides the model elements, text definitions, and UML text-based annotations, partners can annotate particular elements with rich-text documents (e.g., MS Word documents) and commit them to the model repository. These embedded *document snippets* are then seamlessly included in the deliverable at generation-time (cf. the custom system description diagrams in the bottom-right page of Fig. 8). This allows for incorporation in the deliverables of custom-rich content that is freely edited outside of Modelio without being bound by the tool capabilities nor constrained by the formal modeling languages. The automated generation of significant parts of the deliverables ensures overall coherence and consistency of their content, as well as the alignment of their format and style with the project's quality control templates and guidelines.

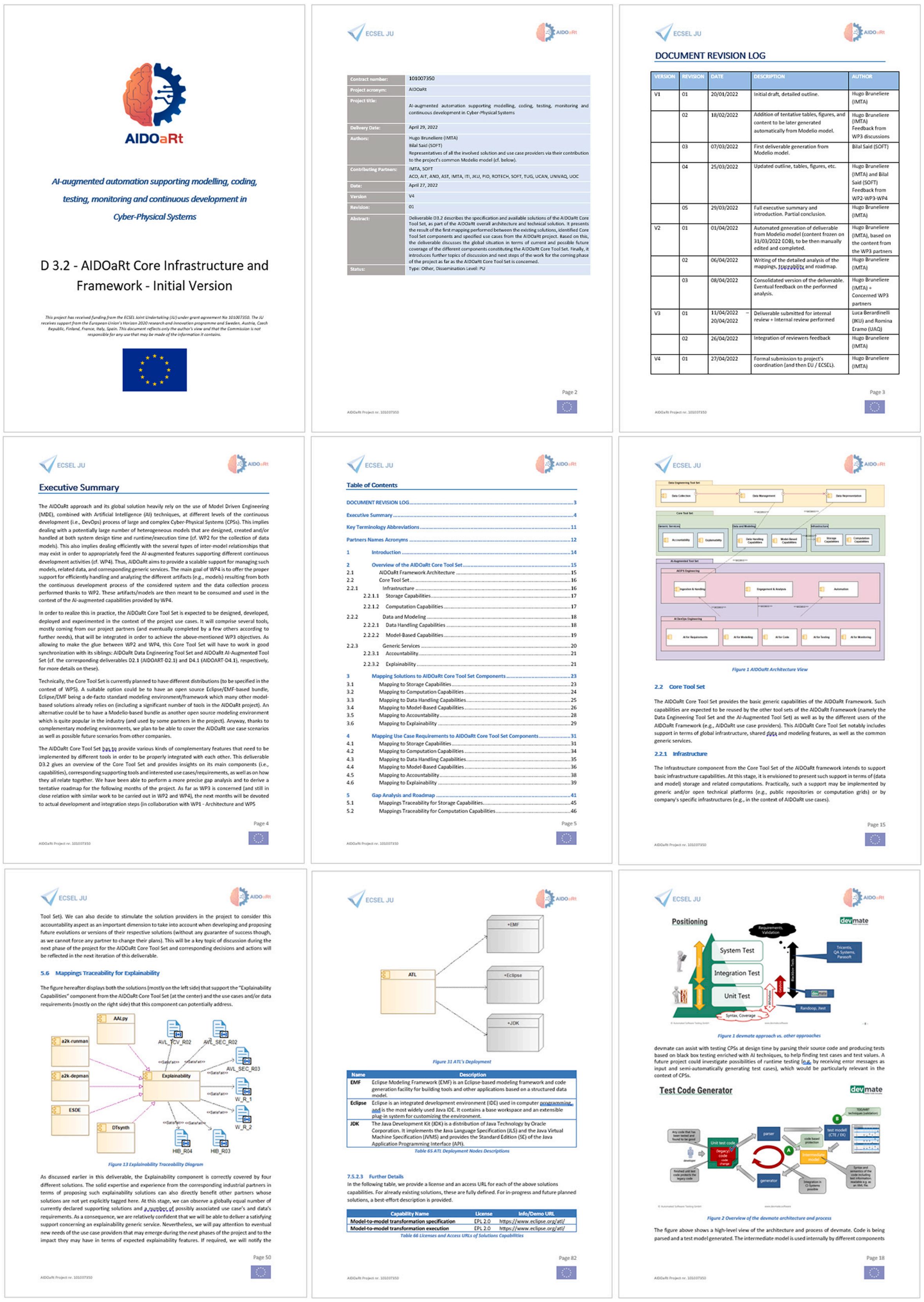
3.4.2. Traceability support

Another interesting feature is the capability to relate and trace our RE models to elements coming from other kinds of models in Modelio. Indeed, the tool integrates UML and many other metamodels, e.g., Enterprise Architecture models expressed in ArchiMate, business models in BPMN [57], system models in SysML, etc. This allows us to use elements of various metamodels together and trace model elements across several metamodels.

Fig. 9 shows the list of solutions (`ToolComponent`) that implement the *AI for Requirements Engineering* architecture component (`FramewokComponent`) from the AIDOaRt project. It also shows the list of case study requirements and data requirements that can be fulfilled (via the *Satisfy* links) by this architecture component. Such traceability links established an indirect relation between tool components and case study requirements, which allowed the AIDOaRt consortium to identify potential collaboration links between solution providers and case study providers.

However, modeling these links is not always an easy task. For instance, in AIDOaRt, before linking a solution component to an architecture component with an *implements* relationship, the solution provider must check that the specification of their solution is aligned with one of the architecture components and its functional interfaces. This requires time-consuming navigation throughout the long list of components and interfaces, reading their descriptions, and checking if they can be met by the solution. An extra action is then needed to navigate back to the solution component and to create the *implements* link element. Similarly, case study providers had to define that a given architecture component or functional interface *satisfies* some of their requirements. Several

[7] additional public deliverables in AIDOaRt are available here: https://www.aidoart.eu/aidoart/results/deliverables.



AIDOaRt

AI-augmented automation supporting modelling, coding, testing, monitoring and continuous development in Cyber-Physical Systems

D 3.2 - AIDOaRt Core Infrastructure and Framework - Initial Version

This project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 101007350. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Sweden, Austria, Czech Republic, Finland, France, Italy, Spain. This document reflects only the author's view and that the Commission is not responsible for any use that may be made of the information it contains.

ECSEL JU AIDOaRt

Contract number:	101007350
Project acronym:	AIDOaRt
Project title:	AI-augmented automation supporting modelling, coding, testing, monitoring and continuous development in Cyber-Physical Systems
Delivery Date:	April 29, 2022
Authors:	Hugo Bruneliere (IMTA) Bilal Said (SOFT) Representatives of all the involved solution and use case providers via their contribution to the project's common Modelio model (cf. below).
Contributing Partners:	IMTA, SOFT ACO, AIT, AND, AST, IMTA, ITI, JKU, PIO, ROTECH, SOFT, TUG, UCAN, UNIVAQ, UOC
Date:	April 27, 2022
Version	V4
Revision:	01
Abstract:	Deliverable D3.2 describes the specification and available solutions of the AIDOaRt Core Tool Set, as part of the AIDOaRt overall architecture and technical solution. It presents the result of the first mapping performed between the existing solutions, identified Core Tool Set components and specified use cases from the AIDOaRt project. Based on this, the deliverable discusses the global situation in terms of current and possible future coverage of the different components constituting the AIDOaRt Core Tool Set. Finally, it introduces further topics of discussion and next steps of the work for the coming phase of the project as far as the AIDOaRt Core Tool Set is concerned.
Status:	Type: Other, Dissemination Level: PU

Page 2

AIDOaRt Project nr. 101007350

ECSEL JU AIDOaRt

DOCUMENT REVISION LOG

VERSION	REVISION	DATE	DESCRIPTION	AUTHOR
V1	01	20/01/2022	Initial draft, detailed outline.	Hugo Bruneliere (IMTA)
	02	18/02/2022	Addition of tentative tables, figures, and content to be later generated automatically from Modelio model.	Hugo Bruneliere (IMTA) Feedback from WP3 discussions
	03	07/03/2022	First deliverable generation from Modelio model.	Bilal Said (SOFT)
	04	25/03/2022	Updated outline, tables, figures, etc.	Hugo Bruneliere (IMTA) and Bilal Said (SOFT) Feedback from WP2-WP3-WP4
	05	29/03/2022	Full executive summary and introduction. Partial conclusion.	Hugo Bruneliere (IMTA)
V2	01	01/04/2022	Automated generation of deliverable from Modelio model (content frozen on 31/03/2022 EOB), to be then manually edited and completed.	Hugo Bruneliere (IMTA), based on the content from the WP3 partners
	02	06/04/2022	Writing of the detailed analysis of the mappings, traceability and roadmap.	Hugo Bruneliere (IMTA)
	03	08/04/2022	Consolidated version of the deliverable. Eventual feedback on the performed analysis.	Hugo Bruneliere (IMTA) + Concerned WP3 partners
V3	01	11/04/2022 – 20/04/2022	Deliverable submitted for internal review + Internal review performed	Luca Berardinelli (JKU) and Romina Eramo (UAQ)
	02	26/04/2022	Integration of reviewers feedback	Hugo Bruneliere (IMTA)
V4	01	27/04/2022	Formal submission to project's coordination (and then EU / ECSEL).	Hugo Bruneliere (IMTA)

Page 3

AIDOaRt Project nr. 101007350

ECSEL JU AIDOaRt

Executive Summary

The AIDOaRt approach and its global solution heavily rely on the use of Model Driven Engineering (MDE), combined with Artificial Intelligence (AI) techniques, at different levels of the continuous development (i.e., DevOps) process of large and complex Cyber-Physical Systems (CPSs). This implies dealing with a potentially large number of heterogeneous models that are designed, created and/or handled at both system design time and runtime/execution time (cf. WP2 for the collection of data models). This also implies dealing efficiently with the several types of inter-model relationships that may exist in order to appropriately feed the AI-augmented features supporting different continuous development activities (cf. WP4). Thus, AIDOaRt aims to provide a scalable support for managing such models, related data, and corresponding generic services. The main goal of WP4 is to offer the proper support for efficiently handling and analyzing the different artifacts (e.g., models) resulting from both the continuous development process of the considered system and the data collection process performed thanks to WP2. These artifacts/models are then meant to be consumed and used in the context of the AI-augmented capabilities provided by WP4.

In order to realize this in practice, the AIDOaRt Core Tool Set is expected to be designed, developed, deployed and experimented in the context of the project use cases. It will comprise several tools, mostly coming from our project partners (and eventually completed by a few others according to further needs), that will be integrated in order to achieve the above-mentioned WP3 objectives. As allowing to make the glue between WP2 and WP4, this Core Tool Set will have to work in good synchronization with its siblings: AIDOaRt Data Engineering Tool Set and AIDOaRt AI-Augmented Tool Set (cf. the corresponding deliverables D2.1 (AIDOART-D2.1) and D4.1 (AIDOART-D4.1), respectively, for more details on these).

Technically, the Core Tool Set is currently planned to have different distributions (to be specified in the context of WP5). A suitable option could be to have an open source Eclipse/EMF-based bundle, Eclipse/EMF being a de-facto standard modeling environment/framework which many other model-based solutions already relies on (including a significant number of tools in the AIDOaRt project). An alternative could be to have a Modelio-based bundle as another open source modeling environment which is quite popular in the industry (and used by some partners in the project). Anyway, thanks to complementary modeling environments, we plan to be able to cover the AIDOaRt use case scenarios as well as possible future scenarios from other companies.

The AIDOaRt Core Tool Set has to provide various kinds of complementary features that need to be implemented by different tools in order to be properly integrated with each other. This deliverable D3.2 gives an overview of the Core Tool Set and provides insights on its main components (i.e., capabilities), corresponding supporting tools and interested use cases/requirements, as well as on how they all relate together. We have been able to perform a more precise gap analysis and to derive a tentative roadmap for the following months of the project. As far as WP3 is concerned (and still in close relation with similar work to be carried out in WP2 and WP4), the next months will be devoted to actual development and integration steps (in collaboration with WP1 - Architecture and WP5

Page 4

AIDOaRt Project nr. 101007350

ECSEL JU AIDOaRt

Table of Contents

Page 5

AIDOaRt Project nr. 101007350

ECSEL JU AIDOaRt

Figure 1 AIDOaRt Architecture View

2.2 Core Tool Set

The AIDOaRt Core Tool Set provides the basic generic capabilities of the AIDOaRt Framework. Such capabilities are expected to be reused by the other tool sets of the AIDOaRt Framework (namely the Data Engineering Tool Set and the AI-Augmented Tool Set) as well as by the different users of the AIDOaRt Framework (e.g., AIDOaRt use case providers). This AIDOaRt Core Tool Set notably includes support in terms of global infrastructure, shared data and modeling features, as well as the common generic services.

2.2.1 Infrastructure

The Infrastructure component from the Core Tool Set of the AIDOaRt framework intends to support basic infrastructure capabilities. At this stage, it is envisioned to present such support in terms of (data and model) storage and related computations. Practically, such a support may be implemented by generic and/or open technical platforms (e.g., public repositories or computation grids) or by company's specific infrastructures (e.g., in the context of AIDOaRt use cases).

Page 15

AIDOaRt Project nr. 101007350

ECSEL JU AIDOaRt

Tool Set). We can also decide to stimulate the solution providers in the project to consider this accountability aspect as an important dimension to take into account when developing and proposing future evolutions or versions of their respective solutions (without any guarantee of success though, as we cannot force any partner to change their plans). This will be a key topic of discussion during the next phase of the project for the AIDOaRt Core Tool Set and corresponding decisions and actions will be reflected in the next iteration of this deliverable.

5.6 Mappings Traceability for Explainability

The figure hereafter displays both the solutions (mostly on the left side) that support the "Explainability Capabilities" component from the AIDOaRt Core Tool Set (at the center) and the use cases and/or data requirements (mostly on the right side) that this component can potentially address.

Figure 13 Explainability Traceability Diagram

As discussed earlier in this deliverable, the Explainability component is correctly covered by four different solutions. The solid expertise and experience from the corresponding industrial partners in terms of proposing such explainability solutions can also directly benefit other partners whose solutions are not yet explicitly tagged here. At this stage, we can observe a globally equal number of currently declared supporting solutions and a number of possibly associated use case's and data's requirements. As a consequence, we are relatively confident that we will be able to deliver a satisfying support concerning an explainability generic service. Nevertheless, we will pay attention to eventual new needs of the use case providers that may emerge during the next phases of the project and to the impact they may have in terms of expected explainability features. If required, we will notify the

Page 50

AIDOaRt Project nr. 101007350

ECSEL JU AIDOaRt

Figure 31 ATL's Deployment

Name	Description
EMF	Eclipse Modeling Framework (EMF) is an Eclipse-based modeling framework and code generation facility for building tools and other applications based on a structured data model.
Eclipse	Eclipse is an integrated development environment (IDE) used in computer programming, and is the most widely used Java IDE. It contains a base workspace and an extensible plug-in system for customizing the environment.
JDK	The Java Development Kit (JDK) is a distribution of Java Technology by Oracle Corporation. It implements the Java Language Specification (JLS) and the Java Virtual Machine Specification (JVMS) and provides the Standard Edition (SE) of the Java Application Programming Interface (API).

Table 65 ATL Deployment Nodes Descriptions

7.5.2.3 Further Details

In the following table, we provide a license and an access URL for each of the above solutions capabilities. For already existing solutions, these are fully defined. For in-progress and future planned solutions, a best-effort description is provided.

Capability Name	License	Info/Demo URL
Model-to-model transformation specification	EPL 2.0	https://www.eclipse.org/atl/
Model-to-model transformation execution	EPL 2.0	https://www.eclipse.org/atl/

Table 66 Licenses and Access URLs of Solutions Capabilities

Page 82

AIDOaRt Project nr. 101007350

ECSEL JU AIDOaRt

Figure 1 devmate approach vs. other approaches

devmate can assist with testing CPSs at design time by parsing their source code and producing tests based on black box testing enriched with AI techniques, to help finding test cases and test values. A future project could investigate possibilities of runtime testing (e.g. by receiving error messages as input and semi-automatically generating test cases), which would be particularly relevant in the context of CPSs.

Test Code Generator
devmate
Unit test code
(legacy) code
code change
developer
parser
generator
test modell (CTE / EK)
Intermediate model
code based protection
TDD/MBT techniques (validation)
Any code that has been tested and found to be good
finished unit test code protects the legacy code
Integration in CI-Systems possible
Syntax and semantics of the code including test information, readable e.g. as an XML file.

Figure 2 Overview of the devmate architecture and process

The figure above shows a high-level view of the architecture and process of devmate. Code is being parsed and a test model generated. The intermediate model is used internally by different components

Page 18

AIDOaRt Project nr. 101007350

Fig. 8. Selected pages from AIDOaRt deliverables that have been automatically initiated from the MBRE model using Modelio.

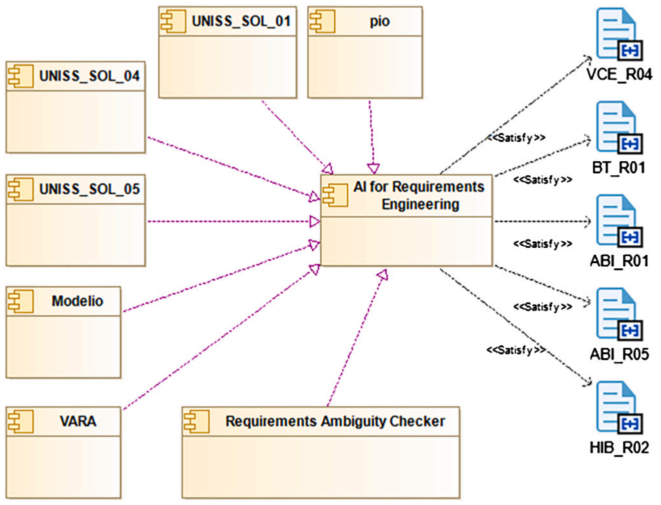


Fig. 9. Traceability diagram in Modelio: Example from AIDOaRt showing the mapping between an architecture component, related solutions, and case study requirements.

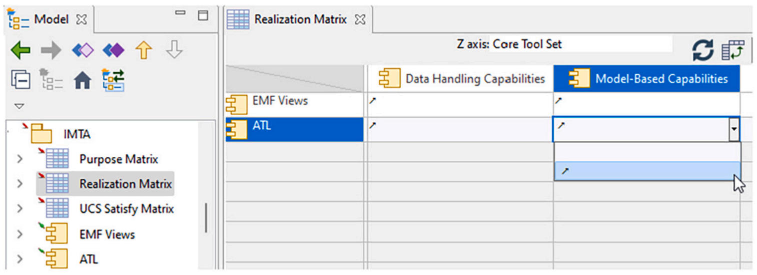


Fig. 10. Matrices in Modelio are used to define relations between the MBRE model elements. The above is a matrix that allows the creation of "implements" relations between solution components and architecture components.

similar actions were needed to define other relationships, e.g., refinement relations between requirements, dependencies between case study scenarios and corresponding functional and data requirements, and associations from data models to data requirements.

As shown in Fig. 10, and to facilitate the creation of these links, we developed and added custom matrices that filter the right type of model elements on rows and columns. This allows us to easily create relations between them by selecting the right value in the corresponding cells.

3.4.3. *Architecture and requirement validation roadmap*

Having explicit traceability between case study requirements and tool/solution requirements allows the project management to have a clear view of which case study requirements will be satisfied during the project and which require additional development. In addition, by having specified for each tool requirement at which milestone will be available for evaluation via the `ReleaseType` property (see Fig. 3), the case study providers and the project management are able to create a roadmap of when the architecture components become available and when case requirements can be validated.

For example, let us consider two case study requirements and two corresponding tool requirements for each of them.

- UCReq1 - *Automated Anomaly Detection* - automatically detect anomalies in cloud-based IoT infrastructures during the execution based on AI.
  - *ToolReq1 - AI/ML Anomaly Detector*. This tool provides log and trace processing, and it uses AI/ML techniques to detect anomalies and flaws in them and provides diagnosis and prediction of potential issues. *ReleaseType:* intermediate
  - *ToolReq3 - AI Log Traces Analyzer* This AI toolset analyzes system logs using text analytics, statistics, and NLP techniques to generate indications of current and prospective system health statuses, and current and future potential issues. *ReleaseType:* initial
- UCReq2 - *AI/ML-based Test Generation* - use automated reasoning and ML techniques for the generation of optimal test suites.
  - *ToolReq2 - Automatic Test Generator*. This tool is meant for automatic test generation and prioritization for software-intensive cyber-physical systems using AI/ML techniques. *ReleaseType:* final
  - *ToolReq4 - Model-based Test Case Generator*. This suite provides automated test case and test code generation using model- and AI-based approaches for test automation and test data determination. *ReleaseType:* intermediate

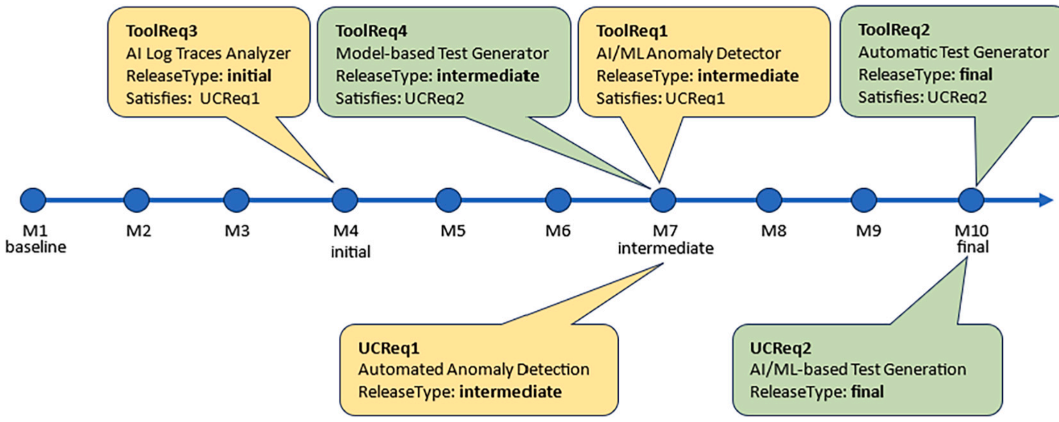


Fig. 11. Generic example of creating the roadmap for use case requirements validation.

Where *initial*, *intermediate*, and *final* are project milestones when versions of the framework architecture are released.

By analyzing the requirement mappings and their *ReleaseType*, one can conclude that *UCReq1* will be fully validated at milestone *intermediate*, whereas requirement *UCReq2* will be fully validated at milestone *final*, as depicted in Fig. 11.

4. Evaluation

To evaluate our MBRE solution, we collected various kinds of data associated with the five large collaborative projects we considered. We describe quantitative and qualitative data resulting from the projects' execution in Section 4.1. We also present general statistics on using our approach in different projects in Sections 4.2 and 4.3. A survey realized in these projects is then presented in Section 4.4. Finally, in Section 4.5, the collected data is used to discuss the relevance of our model-based solution according to the main targeted properties.

4.1. General data on projects

The first concerned project is named **DataBio**, standing for Data-Driven Bioeconomy (Horizon 2020). It lasted 3 years from 2017 to 2019, and involved 48 partners from 17 countries, for a total budget of €15M. 27 case studies in the agriculture, forestry, and fishery areas were developed during this project.

The second project is named **REVaMP2**, standing for Round-trip Engineering and Variability Management Platform and Process (EUREKA ITEA3). It lasted 3 and a half years, from 2016 to 2019, and involved 27 partners from 5 countries, for a total budget of €22M. 7 case studies in the cyber-physical systems, electronic systems, and tourism domains were analyzed during the REVaMP2 project.

The third project is named **MegaM@Rt2**, standing for MegaModeling at RunTime - A scalable model-based framework for continuous development and runtime validation of complex systems (ECSEL). It lasted 3 years, from 2017 to 2020, and involved 27 partners from 6 countries, for a total budget of €16.7M. 9 industrial use cases in the aeronautics, warehousing, automotive, construction, transportation, and telecommunications domains were evaluated.

The fourth project is **VeriDevOps** [25] dealing with automated protection and prevention to meet security requirements in DevOps (H2020). It started in 2020 and is scheduled to end in 2024. The project involves 7 partners from 4 countries for a total budget of €3.9M. The project develops 2 industrial case studies in the industrial automation domain. Compared to the other discussed projects, VeriDevOps is relatively small, though it shares most of the needs in terms of requirements engineering and technical coordination with the other projects. In our analysis, we provide some of the data in a size-normalized way to demonstrate the commonalities among the projects.

The fifth and last project is **AIDOaRt**, a 3 years long H2020-ECSEL European project involving 31 partners, grouped in clusters from 7 different countries, focusing on AI-augmented automation supporting modeling, coding, testing, monitoring, and continuous development in Cyber-Physical Systems (CPS). From 2021 to 2024, the project involves in its consortium 80 full-time equivalents with academic and industrial researchers, project managers, and other staff, for a total budget of €24M. 22 solution providers propose around 50 state-of-the-art research and industrial solutions to tackle 15 case studies proposed by 10 key industrial actors in the fields of automotive, construction, communication, and logistics.

4.2. Analysis of facts and figures from the RE model repositories

Along with the previously mentioned global figures, we provide in Table 2 additional data about the scale of the RE activity, as registered during these five projects, and we illustrate the evolution of this activity in Fig. 12. We collected this data directly from the logs of the model repository used in each project. The repositories are managed by Modelio Constellation using the SVN version

Table 2
Key figures related to the DataBio, REVaMP2, MegaM@Rt2, VeriDevOps, and AIDOaRt projects in terms of RE activities.

Number of	DataBio	REVaMP2	MegaM@Rt2	VeriDevOps[*]	AIDOaRt[*]
Partners	48	27	27	7	31
Countries	17	5	6	4	7
Case studies	27	7	9	2	15
Project months	36	36	36	36	36
Registered users	55	43	56	15	100
Contributors	31 (56%)	24 (56%)	27 (48%)	7 (47%)	60 (60%)
Commits	958	534	1322	328	2548
Requirements:	=181	=535	=428	=124	=455
(Case Study r.	77	190	106	39	363
+ Framework r.	104	56	91	NA[$]	17
+ Tool r.)	NA[#]	289	231	85	75
Model elements[†]	=5406	=3307	=4744	=2087	=9512
(Requirements	535	1091	2351	1211	3507
+ Architecture)	4871	2216	2393	876	6005
Pages generated	61	109	125	120	~160 x 11[‡]

[*] Data from VeriDevOps and AIDOaRt were collected at months M28 and M24, respectively, corresponding to the month of this paper submission.
[#] In DataBio, there was no clear separation of framework and tool requirements.
[$] In VeriDevOps, due to the size of the projects, it has been decided to directly map the case study requirements to the tools.
[†] Includes all applied elements, e.g., attributes, relations, and diagrams.
[‡] In AIDOaRt, we generated 11 deliverables, with ~80 to ~300 pages each, and ~160 pages on average.

control system. Each log trace corresponds to a user's commit to the model and consists of a timestamp, a user ID (i.e., a member of a partner organization in the project), and an (auto-generated yet user-editable) commit message that indicates the modified model elements being committed (added, removed, edited). At the time of writing, data captured in VeriDevOps and AIDOaRt concern the period up to months M28 and M24, respectively, over the whole project duration, which is 36 months. Thus, this data may further evolve in the upcoming months until the end of these two projects.

Overall, the data show that the number of individuals involved in the RE process, whether just registered users with a "read-only" profile or active contributors with a read-and-write profile, was important. Interestingly, the ratio of active contributors is globally similar in all projects (~ 50 to 60%), including the relatively smaller project VeriDevOps.

Regarding RE activities, as illustrated in our case by the number of commits on the requirements model, we observed a disparity between the five projects even though it stays globally important in all of them (cf. Fig. 12a). This can be explained by the slightly different sizes of the five projects, e.g., the number of partners or case studies. This can also be partly explained by the nature of the single commits: a given user may frequently do small single commits while another may commit a large number of updates as a single commit. This effect may have been amplified with microtasking in recent projects.

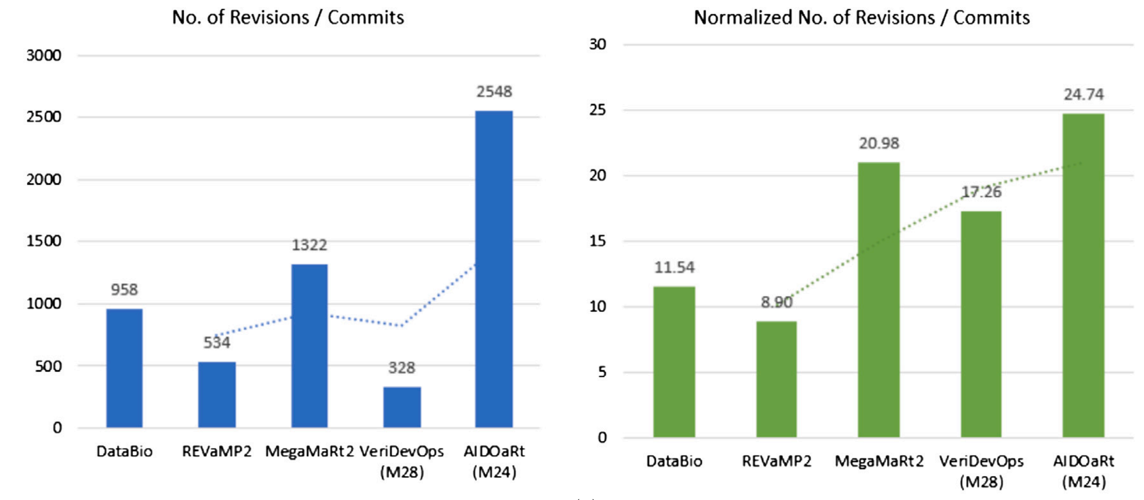
Regarding the requirements models, the data highlight the globally high number of handled requirements and related elements in the context of the five projects (cf. Fig. 12b). We can observe differences between the projects, but, as stated before, this can be explained by the specificity of each particular project, e.g., the number of partners and the case studies to be covered. These differences are also directly reflected and visible in the number/size of the various project documents or deliverables generated from the requirements models in the five projects.

We also want to note that the initial use of ArchiMate in the DataBio project, as previously mentioned in Section 3.4, does not have a significant influence since the number of concepts considered for modeling the architecture is exactly the same as in UML. The slightly bigger numbers can be explained by the greater size of the DataBio project in terms of involved partners and tools. Generally, the number of commits, model elements, and active users are higher in projects with larger consortia and tend to increase over time. This reflects an increasing involvement of the partners and a larger adoption of the MBRE solution (cf. Fig. 12).

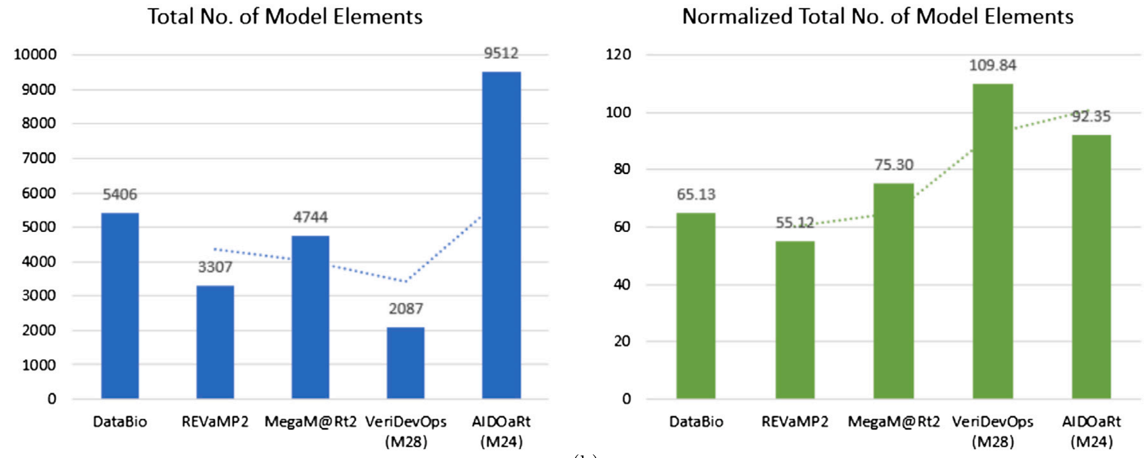
Finally, we can observe that there is a significant difference between the number of requirements level elements and architecture level elements in the five projects. There are several possible explanations: 1) Architecture specification requires many elements, e.g., interfaces, components, relations, to satisfy a set of requirements; 2) the UML metamodel usually requires the creation of several elements for a given goal, e.g., association ends are created when an association is added (and these are also counted).

4.3. Analysis of the chronological activity in the RE model repositories

In this section, we provide a more in-depth analysis of the chronology of the repository logs. The goal is to study the evolution of the actual modeling activity monitored during the execution of the projects and to identify common patterns occurring across the various projects.



(a)



(b)

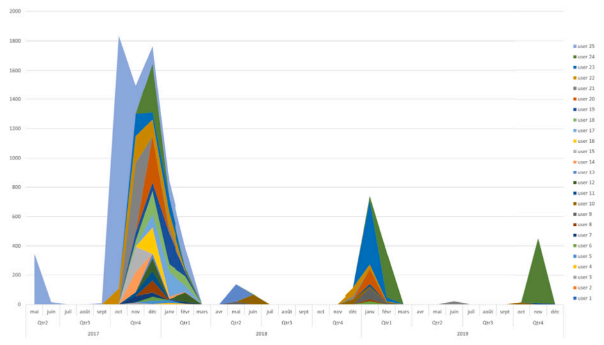
Fig. 12. Figure (a) shows the evolution of the total number of commits or revisions made by the project members to their model repositories, and Figure (b) shows the total number of created model elements. The left charts show the raw numbers, whereas the right charts show these numbers normalized w.r.t. the size of the projects (i.e., the number of partners, case studies, and solutions). The dotted curves illustrate the moving average trends.

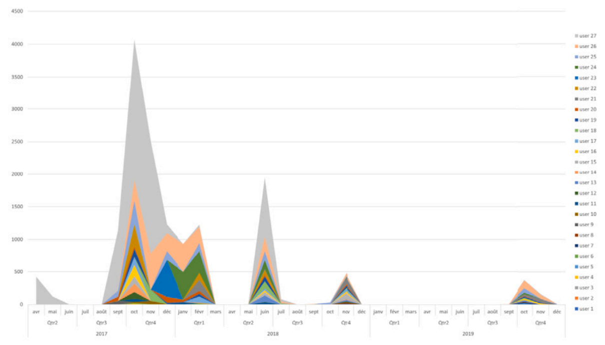
To this end, we analyzed the timestamps of the log traces and aggregated the number of edited model elements in every commit by each one of the contributors over the same periods. We present this data for the REVaMP2, MegaM@Rt2, and AIDOaRt projects (cf. Fig. 13a, 13b, and 13c, respectively). We visualize the data in stacked area charts, where the colored area under a certain curve denotes the number of model elements edited by a given user (the names of the contributors were anonymized into “user 1”, “user 2”, etc., for privacy reasons).

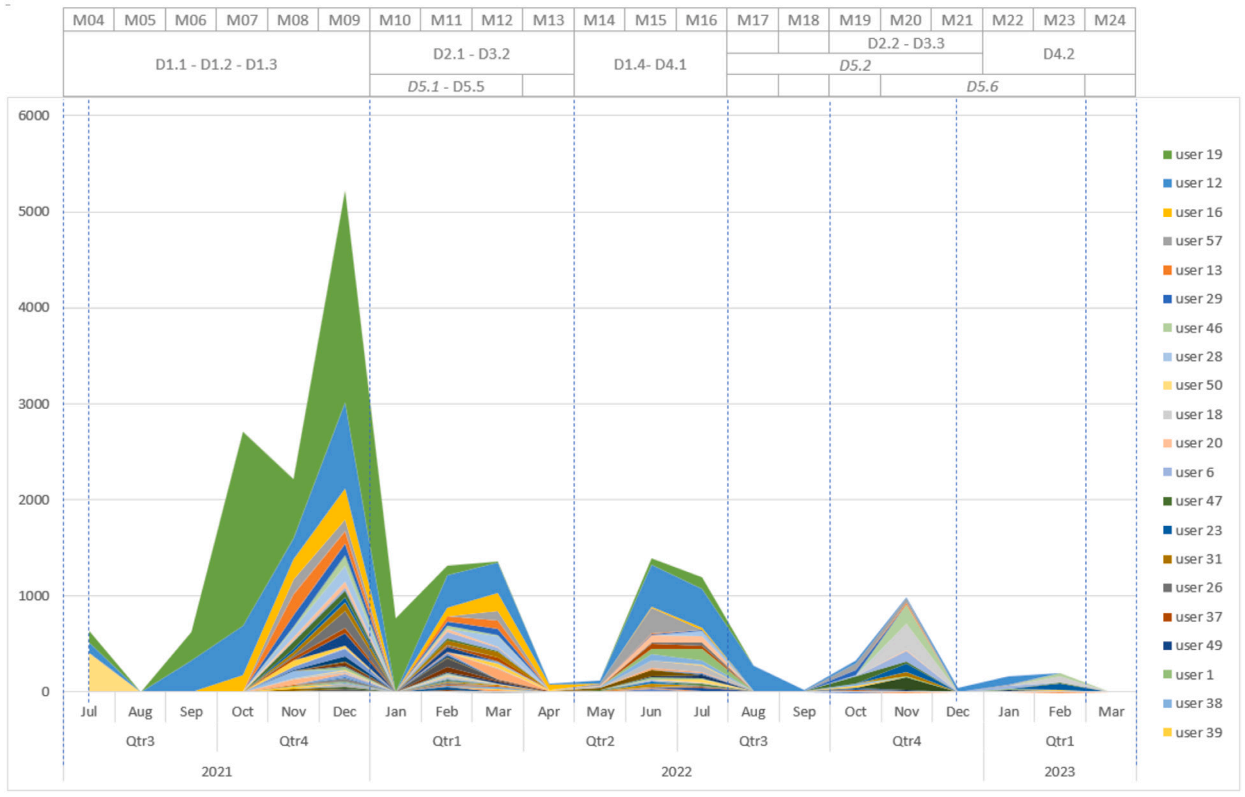
The model repository administrators and maintainers (e.g., user 19 and user 12 from AIDOaRt in Fig. 13c) contributed the most, i.e., they are associated with the largest areas in the chart. These users are typically SOFTEAM R&D team members who closely work on Modelio and Constellation. They instantiate and configure the RE model and create the model templates, document generation, and model verification scripts. They also perform initial bulk imports from external data sheets and often complete partners’ submissions with some corrections or enhancements while doing recurrent reviews and model quality control routines. Some of the large areas correspond to active users from the core teams, work package leaders, or partners maintaining a large number of model elements, e.g., participating with a large number of case studies or solutions in the project.

The chronological evolution of the users’ activity in the three projects shows a common pattern:

- Very low activity takes place at the beginning of each project, typically by the model administrators, to initialize and prepare the model repository. Then, for almost 6 months after the project’s kick-off, partners informally present and discuss their case studies and solutions, and data is typically collected in semi-structured sheets and documents. Meanwhile, the core teams plan the first microtasks, and model administrators familiarize the partners with the MBRE solution. The partners are generally not asked to directly contribute to the RE model during this period.


(a) REVaMP2


(b) MegaM@Rt2



(c) AIDOaRt (up to month M24)

Fig. 13. (a), (b), and (c) show the evolution of the number of commits over the whole project duration for REVaMP2, MegaM@Rt2, and AIDOaRt (on M24, i.e., up to month 24 over 36), respectively. Peaks correspond to deliverable submission deadlines. Areas are colored w.r.t. the different users of the model repository. The surface of the colored areas reflects the extent of involvement of the users in editing and updating the MBRE model.

- High activity is identified towards the end of the first year in each project. It corresponds to the initial phase of requirements elicitation and collection of solution descriptions directly submitted by the partners into the RE model.
- Intermittent and sporadic phases of relatively lower activities are witnessed afterward. They correspond to the phases of updating the model to refine the requirements and solutions definitions, specify generic requirements or a common framework architecture, and declare development roadmaps, potential mappings, and integration between requirements and solutions.
- Interleaving inactivity phases correspond to vacations and holidays, to the beginning of a new milestone where new deliverables are planned but no content is collected from the partners yet, or to the periods where the R&D activities are taking place but not being reported into the RE model yet.
- A final wave of activities systematically occurs towards the end of the project. It typically allows reporting about case studies implementation/results, KPIs measures, and project objectives achievement. These are typically reported in the last deliverables of each project.

A closer look at the AIDOaRt project (Fig. 13c) shows a strong correlation between the users' activity in the RE model and the projects' deliverable timelines. Notable exceptions are those in *italics*, i.e., *D5.1*, *D5.2*, and *D5.6* which were not generated from the

RE model, and to a lower extent D4.2 whose content was mostly provided at the same time as D2.2 and D3.3 (thus relying on the same set of microtasks). The peaks of activity take place close to the deliverable submission deadlines.

In general, several observations can be made from the previous analysis:

- Given the slow start at the beginning of each project, it is complicated to start collecting partner contributions in the RE model before several months of work. Thus, promising deliverables relying on contributions to the RE model at this early stage would require amendments to reschedule these deliverables' submission dates and probably several subsequent ones, which might delay the whole project.
- Having most of the partners' contributions close to the deliverables deadlines highlights the importance of having an approach that allows for timely, continuous, and coherent integration of last-minute contributions and corrections in the deliverables. In this sense, our collaborative model-based solution, coupled with (semi-) automated document generation, is relevant for real-time integration of partners' content in the deliverables while guaranteeing global coherence.
- The ability to do such an analysis also shows that our solution and the use of a collaborative model repository should be considered an important project management tool. Especially in projects with large consortia, this allows to plan and follow up the implementation of project milestones more easily. Moreover, it can also be used as a monitoring tool to measure partners' regular and active involvement throughout the project's lifetime.

4.4. Survey for projects' participants

To evaluate our MBRE solution according to more data, we also ran two consecutive online surveys among the partners of the AIDOaRt, MegaM@Rt2, REVAMP, and DataBio projects. We wanted to validate whether our approach is relevant and helpful in practice in the context of large collaborative research projects.

4.4.1. Survey design

We ran the first survey in 2021 with DataBio, REVAMP, and MegaM@Rt2 projects participants [27]. The second survey was conducted with AIDOaRt participants in the middle of that project. Note that we plan to run the second survey once again at the end of the AIDOaRt project to collect additional data.

In the first survey, we started with three quantitative assessment questions:

- *Q1: In your opinion, did you find this graphical model-based approach useful in different activities of Requirements Engineering? Followed by the list of main RE activities.*
- *Q2: In your opinion, do you see the modeling approach as an improvement compared to other non-modeling (e.g., text-only or table-based) regarding the following aspects? Followed by the list of characteristics that the requirements have to follow, such as correctness and traceability [58].*
- *Q3: In your opinion, did you find the following Modelio tool features useful in different Requirements Engineering activities? We listed all presumably key features of Modelio that could be considered helpful.*

In addition to these quantitative assessment questions, we proposed three qualitative assessment questions:

- *Q4: In your opinion, what was the most challenging aspect of the Modelio-based approach?*
- *Q5: In your opinion, what was the most useful aspect of the Modelio-based approach?*
- *Q6: In your opinion, which additional Modelio tool features would have been useful for RE in the project?*

In the second survey for the AIDOaRt participants, we decided to replace the "model-based" with "Modelio-based", since we believe this was easier to understand for the participants. In addition, we added a specific section to poll their perception of the benefits of the microtask-based iterative process for RE. We intended to validate our hypothesis that microtasks facilitate (1) comprehension of the full RE approach; (2) uniformity of the RE model that is produced collectively; (3) addressing requirements changeability and uncertainty in a large-scale research project. These questions were structured as follows:

- *Q7: In your experience, microtasks greatly facilitate learning of the proposed MBRE approach?*
- *Q8: In your experience, microtasks result in a coherent and uniform model?*
- *Q9: In your experience, microtasks enable great flexibility in addressing evolving requirements?*

For questions Q1-Q3 and Q7-Q9, we utilized a modified Likert scale [59]. Respondents were asked to indicate their level of agreement with the statements in these questions. They were presented with five options in the following order: *strongly disagree*, *disagree*, *agree*, *strongly agree*, and *no opinion*. For questions Q4-Q6, we provided a text box to allow respondents to express their thoughts and reflections openly.

In all cases, the potential respondents were project partners who held an account in the shared model repository and had access to the modeling. To mitigate subjectivity bias, we deliberately excluded ourselves from the survey, despite being the primary beneficiaries of the solution, as we are responsible for architecture definition in the concerned projects.

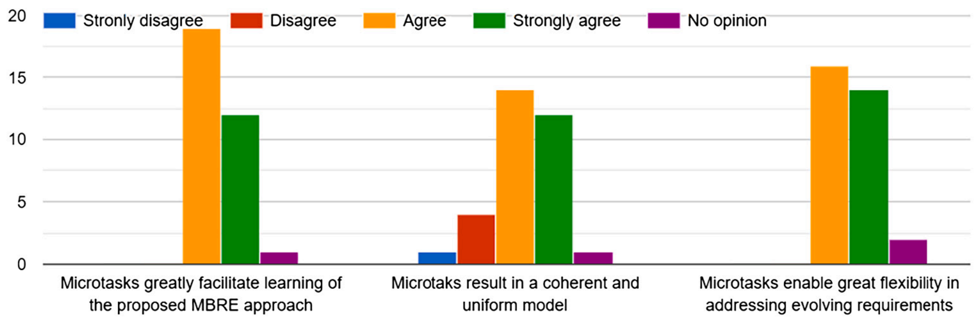


Fig. 14. AIDOaRt survey's respondent opinions about the iterative application of the MBRE approach through the use of microtasks.

4.4.2. *Survey results*

In what follows, we summarize the main findings of the surveys. The complete results are available online [60,61]. For the first survey, we had in total 154 individual contact persons: 55 for DataBio, 43 for REVAMP, and 56 for MegaM@Rt2. We received 15 complete answers, including one from a person not involved in RE activities. For the second survey, we contacted 83 persons from the AIDOaRt project and received 32 replies. Thus, the total number of respondents to the surveys is 47.

We explain this level of participation by several factors. For the first survey: (1) Few of these contact persons were active contributors to the RE process in these projects. Most of them were "readers" or contributed very few elements; (2) It has been at least 1 year since the projects terminated and at least 2 years since the end of the corresponding RE work. For the second survey, the RE work was still ongoing at that time. It is also possible that the non-respondents do not have a definitive opinion on the MBRE solution. Thus, we expect to get a second wave of feedback when we re-run the survey towards the end of the project. We would also like to note that several respondents are part of several projects. Thus, the number of possibly available unique participants in our survey would be lower than the total number of participants in all of the surveyed projects.

While this amount of data cannot be considered statistically representative, we believe that the received feedback still provides interesting and relevant complementary insights. In the next paragraphs, we outline the combined data of both surveys for questions 1-to-6 to present a more coherent analysis.

The majority of the respondents considered that our model-based RE approach was useful for different RE activities. On average, 79.43% would agree that the proposed graphical model-based approach is useful for RE (Q1), 68.80% would agree that the modeling approach is better for RE (Q2), and 60.70% would agree that the implementation of the approach in Modelio was useful for RE (Q3). In Q1, 91.49% agree that model-based approaches are useful in System modeling. In contrast, 61.70% also find that model-based approaches are useful in Requirements inception or requirements elicitation. In Q2, the highest agreement (80.85%) concerns the advantages of a model-based approach regarding Traceability: it can be linked to system requirements, designs, code, and tests. The lowest agreement (59.57%) concerns the benefits of model-based versus non-model-based approaches for dealing with Verifiability (it allows for correct implementation to be determined by testing, inspection, analysis, or demonstration). In Q3, a high number of respondents (95.74%) would find Traceability visualization in a diagram view to be useful. The lowest agreement (57.45%) concerns the usefulness of the approach for road-mapping (e.g., setting up expected delivery dates and completion stages for designed components). This latest result may be due to the very nature of research projects, where some partners are unable to provide exact expected delivery dates for their proposed solutions, prototypes and tools.

Moreover, additional questions on the appropriateness of the approach resulted in the following assessments:

- 85.11% would find the approach appropriate for the given size and scope of the project;
- 87.23% would find the tool support useful for guiding the RE process and enforcing project conventions;
- 78.72% would find the approach easy to learn;
- 76.60% would find the approach easy to apply and 61.70% would use it again in the future.

Concerning our hypothesis about the microtask-based process for MBRE (Fig. 14), the respondents generally agree (100%) that this approach facilitates learning (Q7) and that it provides flexibility to address changing requirements (Q9). In addition, 88.24% agree that the approach helps produce a coherent and uniform model (Q8). This is a promising indication that such microtasking can be recommended in model-based and RE contexts.

4.4.3. *Comparative analysis*

We analyzed the differences in survey results for both the MegaM@Rt (2021) and AIDOaRt (2022) projects, and we summarize them in Fig. 15. We noticed that the respondents from AIDOaRt were less senior with 70.6% having less than 3 years of experience with RE. Half of the participants were academic partners, and 88.2% of respondents identified themselves as technology providers in the project context. 64.7% were used to allocate less than 10% of their time to maintaining RE documents. Based on the comparison analysis (Fig. 15), we can be satisfied that the AIDOaRt respondents approved the use of our MBRE solution on a very similar scale as the more senior respondents of the MegaM@Rt survey. Once again, we believe that microtasking in collaborative modeling was beneficial to overcome the learning barrier and saved time on maintaining the RE documents.

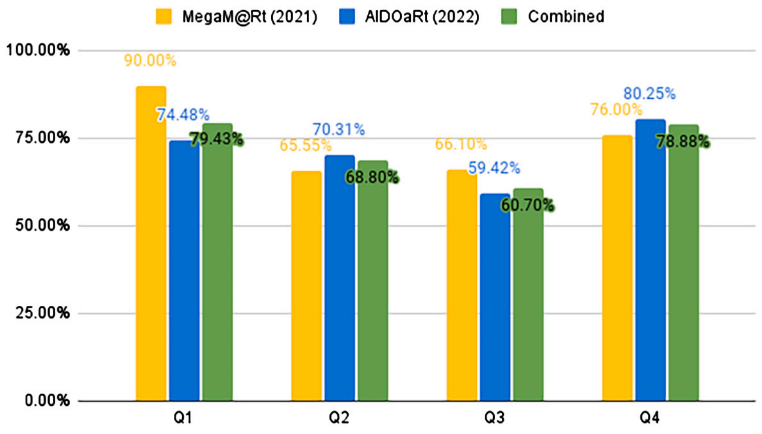


Fig. 15. Percentage of the participants who agree with the positive statements in response to the four main questions of the survey.

Table 3
Positive feedback in the surveys focuses on the importance of (1) collaborative teamwork on the model repository, (2) the continuous guarantee of the model consistency, (3) the usefulness of document generation, and (4) the satisfaction with the agile MBRE approach with the iterative definition and application of microtasks.

The visualization and **document generation** aspects
Getting rid of plain text (or MS Word) descriptions.
Being able to **detect inconsistencies easily**.
Consistency consistency consistency
The **teamwork** module aspects!
[. . .] collecting data, subversion system, and **revision control**
The **centralized 'Ground truth'** and the **document generation**.
Central management of all requirements and **links**
[. . .] provides a project-wide basis for **deeper analysis and research conclusions**.
The **automatic generation of the documentation**, **link** between the different software component
[. . .] on the other hand, through the MBRE and the use of Modelio it was possible to **bring many industrial realities closer to model-based design and requirement management**.
From my experience the **microtasks were great**
Microtasks saved me a lot of time, [. . .] the support given (especially by Modelio's team) in the documentation/instruction/video recording of each microtask made it very simple for me to do the tasks.
[. . .] microtasks remind me of test-driven development (TDD) – the project is good if you have TDD [. . .] because the tasks are small and simple. [. . .] AIDOaRt gained very much from these **microtasks because they made it clear** [. . .] what to do, and how you could review that it had been done.
We have achieved significant advantages in requirements management and architecture when applied to European projects. I believe it is very useful for both project management and system (software) development.
Supports in **resource and timeline planning for RE** engineering
The **achievable consistency of the different documents** needed is absolutely great. It **eases the writing of docs** as well. Thanks!
Visualization, uniform representation, **traceability for large scale / complex systems**
It provides basic support for **requirements traceability and automated documentation generation**.

4.4.4. *Qualitative feedback*

Finally, we also received open qualitative feedback from the surveys, as summarized in Tables 3 and 4. Respondents mention their overall appreciation in terms of the unification of the RE process provided by our solution, its relevance in large European research projects (via our collaborative model repository), the usefulness of microtasks, the improved consistency with traceability support, and the fast and clean documentation generation support. We would like to note that our MBRE solution has already been presented

Table 4
Excerpt of feedback from the survey respondents on the limitations of the adoption of the MBRE approach, its iterative application through the use of microtasks, and the use of the modeling tool, Modelio. Most remarks concern the tool and its technical details rather than the approach per se.

I feel that **[the microtasks] could often be made clearer via tool examples** instead of descriptions, and indeed often this was the case via demos from Softeam or highlighting partner input [. . .]
Doing queries in the model as a more casual user. I sometimes wanted to find some information that could be queried, but I found it hard to solve intuitively.
Navigation in the tool: Where is the link between object A and object B? Should I add a document, or a note, or make a comment in some other way?
The initial learning curve is steep for partners not coming from a strong modeling background. Also, **collaboration is difficult on some occasions** (e.g., nearing a deadline, everybody is locking the same items so it gets difficult and time-consuming to find an available slot to make changes).
Keeping the overview of cross-relations once the model is evolving
Learning to add information
[. . .] the **initial install** and the more or less **hidden changes to the doc generation** ...
Installing the application on a **non-Windows-based OS** + Getting an overview **where specific components could be found**
[We faced several problems with] **MS Word in Modelio** under MacOS and Linux. It is still much better than sparsely editing offline documents with a large number of collaborators!
I think that adding a feature for **generating documents from LaTeX** snippets would be great. [...] As an academic who is used to work with LaTeX, my view is certainly biased.
Difficulty in uploading **Word documents from Linux**.
For a new user like me **it took effort to learn how to use it**. I had some personal help to solve some tasks, otherwise it could take a bit long.
From the point of view of the end user, **the use of the tool is cumbersome**; within the European projects in which it was used, the maximum effort was on the Modelio experts. **The usability of the tool and some facilities can be improved**.
I had a few **issues with obtaining locks** when working on some microtasks, but otherwise no special challenges.

to the independent external experts mandated by the European Commission during project review meetings of the MegaM@Rt2 and AIDOaRt projects. In both cases, our solution received very positive feedback.

Nevertheless, the survey respondents also mentioned some limitations. In the first survey, they mentioned the difficulty of convincing stakeholders to apply the approach at first and difficulties in terms of model synchronization during collaborative editing. In the second survey, they identified the difficulty of navigating and querying the model with a large number of model elements. While extensively using the tool, they also complained about some of its technical details (e.g., limited support on some OS platforms, issues in the teamwork module, and the difficulty of finding specific components).

Furthermore, the respondents proposed several potential improvements according to their past and present practices. Notably, they suggested supporting in the future a better connection with other tools (modeling tools but not only), more advanced traceability between our RE language and other languages (e.g., for software design and development), and providing more structured templates for the requirements description. They also requested to better illustrate new microtasks with additional examples and tool use guidelines, and to show model changes' effects on the generated documentation timely.

4.5. Overall assessment of our MBRE solution

Based on all the collected data, we can provide an assessment of the relevance of our MBRE solution according to various important properties:

- **Scalability** - The consolidated data extracted from the requirements model repository in the five projects clearly shows that we have been able to support a significant number of users (including regularly active ones) as well as to handle a relatively large number of requirements and related model elements. Moreover, the success of the five projects in terms of deliveries (i.e., documents, tools, demonstrators) and the survey results also demonstrate the general scalability of our MBRE solution, at least for projects going up to the size of the five mentioned ones. However, as some survey respondents notice, the larger the model becomes, the more difficult it is to find and work with its elements. To overcome this, we suggested using containers to efficiently group model elements in different ways according to the needs. On the contrary, documentation generation was not affected by the large size of the models and proved its usefulness. Indeed, physical limitations, such as the operating memory size of client computers, may hinder the handling of large models. To address these constraints for very large models, we propose adopting the fragment technique. This approach involves having individual contributors work on smaller pieces of the model, which are then seamlessly integrated into the joint model in a read-only manner.
- **Heterogeneity** - The varied characteristics of the five projects (e.g., various partners providing different kinds of tools and technologies, various case studies covering several application domains) show that we have handled a certain level of heterogeneity. Based on our experience in these five projects, and the participants' feedback we collected via the surveys, we can also argue

that our MBRE solution can be applied similarly in any large collaborating project whose main purpose is to produce integrated software solutions.

- **Adaptability and Extensibility** - We have also shown that our MBRE solution can be extended with new elements (e.g., in the RE language or in the microtasks) that satisfy the needs of given projects. This is particularly useful to adapt to the partners' different levels of experience and their various practices. In particular, we have added extensive support for various types of Use Case requirements in the AIDOaRt project. Moreover, we applied this type of customization in the VeriDevOps project where, due to its smaller size, we skipped the use of microtasks. Overall, the adaptability and extensibility of our approach are subject to the limitations of UML. While UML is an excellent tool for high-level architecture and software design, it has limited applicability in other areas, such as embedded systems. However, specialized extensions like SysML and the MARTE UML profile address these limitations and offer more suitable solutions for certain domains.
- **Traceability** - From the surveys' results, we can state that traceability and the capability to guide and enforce full MBRE processes have been the most acknowledged features of our MBRE solution (85.11% agree on the usefulness of traceability in Q2). It should be noted that the requirements and architecture elements are traced using manually modeled dependency relations. Furthermore, there is room for potential improvements and extensions in this area, such as the implementation of tooling support for traceability from the model down to the source code. Nevertheless, the feedback received up to now already highlights an acceptable support for traceability.
- **Automation** - The quantitative data collected show that we have automatically generated documents of significant sizes from the corresponding RE models. These were quite complete requirements, architecture, or road-map documents whose content could then be reused directly to produce the official project deliverables. The survey confirms participants' appreciation towards document generation since 91.49% agree in Q3 on the usefulness of this feature. The document generation heavily relies on manually created templates, which necessitates expert support. Though not total, the achieved level of automation was already quite appreciated in the projects.
- **Consistency and Quality** - Having a single central model allows us to periodically inspect the quality and consistency of the data using automated scripts. This has been used at different steps of our overall RE process in order to check various kinds of properties regarding the case study requirements, architecture components, tool components, etc. However, improvements are still possible regarding the use of Model Verification and Validation techniques.
- **Usefulness and Usability** - The fact that our MBRE solution has been successfully used in practice in the context of five different collaborative projects already shows a certain level of usability. The qualitative data collected from the surveys also confirm that users globally found our solution useful, as 79.43% replied positively in Q1. Furthermore, they appreciated in particular the fact that the solution was relatively easy to both learn and apply in their respective contexts.

5. Discussion

In the following, we summarize our experience from applying our MBRE approach into a set of lessons learned (Section 5.1), we discuss threats to validity we have identified concerning the current evaluation of our MBRE solution (Section 5.2), and we introduce some open challenges concerning both our model-based approach and its application in other contexts in the future (Section 5.3).

5.1. Lessons learned

From our experience of designing and then applying our MBRE solution in the context of five different large collaborative European projects, we have been able to extract some general lessons learned. We hope that they will be useful to the modeling community as well as more globally to the whole Software Engineering community:

- **Model as a common language** - Having a common modeling language and process for the entire project consortium allowed us to address the heterogeneity of the partners by "forcing" them to use the same steps, concepts, and notations. In addition, the tool support for the approach allowed the partners to work collaboratively on the same model by updating and retrieving the relevant information at different phases of the project. Finally, having traceability links between the elements of the same model allowed us to easily analyze the model to generate road maps and perform gap analysis.
- **Project Planning** - As discussed in Section 4.3, we have observed that using a model-based approach in our RE context can impact how the project's milestones and deliverables are planned. Notably, we suggest considering the time needed to plan the contents of the deliverables and define the required microtasks at the beginning of each milestone. This is crucial to be completed before all partners can collaborate on the RE model.
- **Project Management** - The feedback received shows that our model-based solution is beneficial from the project management perspective: One of the most valuable user features is the automation capability. Notably, the possibility to perform more easily gap analysis, obtain a corresponding roadmap, and generate long documents directly from the requirements model was highly appreciated. In addition, the funding agencies highly appreciated and recommended the model-based solution since the technical coordination effort was made visible, and the projects were more manageable.
- **Framework Architecture** - The proposed model-based solution was naturally relevant for defining the framework architecture. Notably, the combination of well-known diagrams originating from UML and SysML with tabular views was perceived as a good way to facilitate the use of our solution, e.g., for partners already having experience in modeling activities. In addition,

the support for collaborative work and the integration with a model repository were considered among the most appreciated features by our partners.

- **Language and Model Complexity** - With the growth of the project complexity and related needs, more and more concepts had to be modeled. For example, *data requirements*, *data models*, and *solutions* were added to the model in the context of AIDOaRt. Thus, it is important to adopt a flexible approach that can easily incorporate new concepts. To this extent, microtasking helps in bringing all the team on the same page. Nevertheless, it is still necessary to rely on modeling experts for the coordination effort.
- **Learning Curve** - Some participants in our projects had a somehow limited experience in modeling, both conceptually and technically. This resulted in difficulties for them to catch up with some of the concepts in our RE language. It also appears that initially provided guidelines on the approach/language and Modelio tooling were not sufficiently detailed. This was quickly fixed by the Modelio team via online hands-on sessions, for example. The participants were then more easily able to go on with their modeling activities in the context of their respective projects. In addition, we recommend the microtasking approach to overcome the learning barriers and enhance the uniformity of the modeling effort.
- **MBRE and Collaboration** - Our experience has demonstrated that MBRE and microtasking greatly facilitate collaboration. The joint model repository encompasses various types of case study requirements, framework architecture, and high-level architecture of proposed solutions. Accessible to every project member, the repository enables seamless navigation from requirements to solutions and vice versa. This approach provides the case study partners with clear indicators of which technology providers address their use cases. Simultaneously, the technology providers identify potential partners for a joint solution to address a case study. Microtasks serve as synchronization points where all partners contribute and become aware of each other's contributions during the synchronization meetings. These essential synchronizations are conducted at regular intervals.

5.2. Threats to validity

The main threat to validity concerns the amount of survey data we have been able to gather from project participants. Indeed, our MBRE solution has been deployed in “only” five different projects from which we have extracted mostly quantitative data. However, the fact that these were large projects that ran over 7 years in total already provides a certain level of confidence about the quality and relevance of the collected data. Moreover, we complemented this quantitative data with extra data (quantitative but also qualitative this time) collected from two surveys largely distributed among the participants of the five projects. Obviously, it would have been more significant to get more answers to the survey. However, we believe the collected feedback coming from 47 different participants is already interesting to improve our global appreciation of the proposed solution. Another threat stands in the fact that the evaluation of the proposed solution included several dimensions of our work, i.e., the modeling language, the model-based process, and the supporting tooling. To this end, we paid attention to gathering different types of data on these dimensions and specifying separate groups of complementary questions in the survey.

5.3. Challenges

As a result of all our work and experience, we highlight some big challenges we believe to be worth investigating as far as MBRE or, more generally, modeling is concerned:

- **From Requirements to Source Code** - Our RE models contain information concerning mostly the needs and architectural decisions at the project level. While this is already useful for coordinating the common global effort toward the realization of the target solution, it remains relatively far from being fully model-driven. Indeed, we have not yet integrated the use of software models, and related capabilities, such as the generation of implementation and verification artifacts from the model(s), are not supported at this stage. Thus, efforts still have to be made to better fill this gap between the system's architecture and development levels in our MBRE solution and more globally in many others.
- **User Training and Support** - In our RE language, we deliberately decided to extend UML and restrict its usage to a limited number of concepts. We also provided tooling support, user guidelines, and online workshops to get the various projects' partners more acquainted with modeling in the Modelio environment. Nevertheless, some partners appeared to still need regular support concerning both the tooling and the solution. Thus, we strongly believe that the usability and learning curve of model-based solutions are the key elements to consider and improve accordingly to allow for their large dissemination in different contexts. In the AIDOaRt project, this was partially achieved by integrating the use of microtasks.
- **Collaborative and Online Model Editing** - One of the most reported issues concerned restrictions in the collaborative editing capabilities provided by our model-based solution. The Modelio model repository we use currently relies on a Subversion-based “lock - edit - commit - release” operation mode. However, modern users generally tend to prefer online editing collaboration modes, e.g., via their favorite web browser. Even if the situation has improved during the last few years, we have seen limited support for that so far in existing modeling tools or even in popular IDEs. Extended support for online collaboration at the model level is probably a path to be explored more deeply in the future to improve current model-based solutions.
- **Automation and Production-Readiness** - There are still open challenges related to the support for automation in model-based processes and, more generally, in Software Engineering processes. For instance, we have been considering the possibility of building some more automated support for document generation or even code generation to be integrated into our overall RE process. However, it is always a matter of cost/benefit balance since development resources can be limited in collaborative R&D

projects. Thus, our model-based solution is still not in the full production stage. Indeed, it requires some level of customization for each new project and will require higher investments and efforts to eventually become an actual product in the future.

6. Conclusion

In this paper, we reported on our practical experience of designing, developing, and deploying a Model-based Requirements Engineering (MBRE) solution over 7 years in the context of 5 different large European collaborative projects providing complex software solutions. Our overall MBRE solution provides an iterative model-based approach accompanied by an RE modeling language and the corresponding tooling support. To address the two main challenges introduced in Section 1, our solution aims at supporting several complementary activities: 1) requirements can be modeled appropriately in different layers (case study, framework, tool), 2) modeled requirements can be better interconnected and traced during the RE process, 3) modeled requirements can be used to (semi-)automatically perform gap analysis and propose a roadmap, 4) modeled requirements are also used for designing, implementing, and validating the architecture of the framework provided by the project, 5) different deliverables of the project are automatically generated from the RE model and 6) modeling activities are done incrementally using microtasks with clearly specified guidelines.

Based on this global experience and the collected data, we demonstrated that our model-based solution can bring interesting benefits in terms of scalability, heterogeneity, adaptability/extensibility, traceability, automation, consistency/quality, and general usefulness or usability. We also discussed the limitations we faced, as well as open challenges to improve our solution in terms of modeling support (among others). In the next steps of our work, we will continue to collect even more data from the solution usage within the still-running projects VeriDevOps and AIDOaRt. We will also re-apply and extend the solution within new collaborative projects to start. In particular, we already plan to take into account the modeling of projects' KPIs, high-level objectives, and evaluation results from the case studies. Furthermore, we also want to consider the technical integration of our solution with existing project management tools.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work has received funding from the ECSEL Joint Undertaking under grant agreements No. 101007350 (AIDOaRt project) and No. 737494 (MegaM@Rt2 project), from ITEA3 under grant agreement No. 15010 (REVaMP2 project), and from H2020 under grant agreements No. 732064 (DataBio project) and No. 957212 (VeriDevOps project).

References

[1] R. Van Noorden, D. Butler, Science in Europe: by the numbers, Nature 569 (May 2019).
[2] European Commission-DG CONNECT, Digital economy and society index (DESI) 2020 - the EU ICT sector and its R&D performance, online at https://ec.europa.eu/newsroom/dae/document.cfm?doc_id=72352. (Accessed 14 March 2023), 2023.
[3] European Commission, EU research programmes, https://ec.europa.eu/info/funding-tenders/funding-opportunities/funding-programmes/overview-funding-programmes_en. (Accessed 14 March 2023), 2023.
[4] Accelopment, Lessons learnt from horizon 2020 for its final 2 years, https://accelopment.com/blog/lessons-learnt-from-horizon-2020-for-its-final-2-years/. (Accessed 7 April 2021), Feb. 2019.
[5] Ecsel joint undertaking work plan 2020, https://www.ecsel.eu/calls-2020-wp2020, 2023.
[6] ECSEL-JU, Productive4.0 project, https://www.ecsel.eu/projects/productive40. (Accessed 14 March 2023), 2023.
[7] S. Gürses, M. Seguran, N. Zannone, Requirements engineering within a large-scale security-oriented research project: lessons learned, Requir. Eng. 18 (1) (2013) 43–66, https://doi.org/10.1007/s00766-011-0139-7.
[8] D. Nepelski, G. Piroli, Organizational diversity and innovation potential of EU-funded research projects, J. Technol. Transf. 43 (3) (2018) 615–639.
[9] IEEE/ISO/IEC, International standard - systems and software engineering – life cycle processes – requirements engineering, Tech. Rep. 29148-2018, IEEE Standards Association, 2018.
[10] J. Dick, E. Hull, K. Jackson, Requirements Engineering, Springer, Berlin, Germany, 2017.
[11] K. Pohl, Requirements Engineering - Fundamentals, Principles, and Techniques, Springer-Verlag Berlin Heidelberg, 2010.
[12] A. Van Lamsweerde, Requirements Engineering: From System Goals to UML Models to Software, vol. 10, John Wiley & Sons, Hoboken, NJ, U.S.A., 2009.
[13] G. Kotonya, I. Sommerville, Requirements Engineering: Processes and Techniques, John Wiley & Sons, Hoboken, NJ, U.S.A., 1998.
[14] B. Nuseibeh, S. Easterbrook, Requirements engineering: a roadmap, in: ICSE 2000, ACM, New York, NY, USA, 2000, pp. 35–46.
[15] B.H.C. Cheng, J.M. Atlee, Research directions in requirements engineering, in: 2007 Future of Software Engineering, FOSE '07, IEEE Computer Society, USA, 2007, pp. 285–303.
[16] Modelio, A collaborative business or software modeling platform, https://www.modeliosoft.com/en/, 2023.
[17] O.M.G. Unified, Modeling language, https://www.uml.org/, 2023.
[18] O.M.G. Systems Modeling Language (SysML), https://www.omgsysml.org/, 2023.
[19] J. Hutchinson, J. Whittle, M. Rouncefield, Model-driven engineering practices in industry: social, organizational and managerial factors that lead to success or failure, in: Special Issue on Success Stories in Model Driven Engineering, Sci. Comput. Program. 89 (2014) 144–161, https://doi.org/10.1016/j.scico.2013.03.017, https://www.sciencedirect.com/science/article/pii/S0167642313000786.

[20] D. Leffingwell, Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise, Addison-Wesley Professional, 2010.

[21] European Cooperation for Space Standardization, Space engineering software, Tech. Rep. ECSS-E-ST-40C, ECSS Secretariat ESA-ESTEC Requirements & Standards Division, Noordwijk, the Netherlands, Mar. 2009.

[22] A. Sadovykh, A. Bagnato, A.J. Berre, S. Walderhaug, Archimate as a specification language for big data applications - databio example, in: J.-M. Bruel, M. Mazzara, B. Meyer (Eds.), Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment, Springer International Publishing, Cham, 2020, pp. 191–199.

[23] A. Sadovykh, T. Ziadi, A. Bagnato, T. Berger, J.-P. Steghöfer, J. Robin, R. Mazo, E. Gallego, Revamp2 project: towards round-trip engineering of software product lines - approach, intermediate results and challenges, in: M. Mazzara, J.-M. Bruel, B. Meyer, A. Petrenko (Eds.), Software Technology: Methods and Tools, Springer International Publishing, Cham, 2019, pp. 406–417.

[24] W. Afzal, H. Bruneliere, D. Di Ruscio, A. Sadovykh, S. Mazzini, E. Cariou, D. Truscan, J. Cabot, A. Gómez, J. Gorroñogoitia, L. Pomante, P. Smrz, The MegaM@Rt2 ECSEL project: MegaModelling at runtime – scalable model-based framework for continuous development and runtime validation of complex systems, Microprocess. Microsyst. 61 (2018) 86–95.

[25] A. Sadovykh, G. Widforss, D. Truscan, E.P. Enoiu, W. Mallouli, R. Iglesias, A. Bagnto, O. Hendel, Veridevops: automated protection and prevention to meet security requirements in devops, in: 2021 Design, Automation and Test in Europe Conference and Exhibition (DATE), 2021, pp. 1330–1333.

[26] H. Bruneliere, V. Muttillo, R. Eramo, L. Berardinelli, A. Gómez, A. Bagnato, A. Sadovykh, A. Cicchetti, Aidoart: ai-augmented automation for devops, a model-based framework for continuous development in cyber-physical systems, Microprocess. Microsyst. (2022) 104672.

[27] A. Sadovykh, D. Truscan, H. Bruneliere, Applying model-based requirements engineering in three large European collaborative projects: an experience report, in: 2021 IEEE 29th International Requirements Engineering Conference (RE), 2021, pp. 367–377.

[28] G. De Angelis, A. Ferrari, S. Gnesi, A. Polini, Collaborative requirements elicitation in a European research project, in: Proceedings of the 31st Annual ACM Symposium on Applied Computing, SAC '16, Association for Computing Machinery, New York, NY, USA, 2016, pp. 1282–1289.

[29] G. De Angelis, A. Ferrari, S. Gnesi, A. Polini, Requirements elicitation and refinement in collaborative research projects, J. Softw. Evol. Process 30 (12) (2018) e1990, https://doi.org/10.1002/smr.1990, https://onlinelibrary.wiley.com/doi/abs/10.1002/smr.1990.

[30] P. Cecilio Lopes, A. Rodrigues da Silva, A collaborative platform for better managing technical documentation: an analysis from a requirements engineering perspective, in: 2018 11th International Conference on the Quality of Information and Communications Technology (QUATIC), 2018, pp. 160–163.

[31] A. van Lamsweerde, Requirements engineering in the year 00: a research perspective, in: Proceedings of the 2000 International Conference on Software Engineering. ICSE 2000 the New Millennium, 2000, pp. 5–19.

[32] S. Assar, Model driven requirements engineering: mapping the field and beyond, in: 2014 IEEE 4th International Model-Driven Requirements Engineering Workshop (MoDRE), 2014, pp. 1–6.

[33] M. Brambilla, J. Cabot, M. Wimmer, Model-driven software engineering in practice, Synth. Lect. Software Eng. 1 (1) (2012) 1–182.

[34] E.S. Yu, Towards modelling and reasoning support for early-phase requirements engineering, in: ISRE 1997, IEEE Computer Society, Washington, DC, U.S.A., 1997, pp. 226–235.

[35] A. Van Lamsweerde, Goal-oriented requirements engineering: a guided tour, in: ISRE 2001, IEEE Computer Society, Washington, DC, U.S.A., 2001, pp. 249–262.

[36] Object Management Group (OMG), Requirements interchange format (ReqIF) (2020), https://www.omg.org/spec/ReqIF.

[37] B. Baudry, C. Nebut, Y.L. Traon, Model-driven engineering for requirements analysis, in: 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), 2007, p. 459.

[38] A. Goknil, I. Kurtev, K. van den Berg, A metamodeling approach for reasoning about requirements, in: I. Schieferdecker, A. Hartman (Eds.), Model Driven Architecture – Foundations and Applications, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 310–325.

[39] E. Letier, J. Kramer, J. Magee, S. Uchitel, Deriving event-based transition systems from goal-oriented requirements models, Autom. Softw. Eng. 15 (2) (2008) 175–206.

[40] J. Konaté, A.E.K. Sahraoui, G.L. Kolfschoten, Collaborative requirements elicitation: a process-centred approach, Group Decis. Negot. 23 (4) (2014) 847–877, https://doi.org/10.1007/s10726-013-9350-x.

[41] S. Karg, A. Raschke, M. Tichy, G. Liebel, Model-driven software engineering in the openetcs project: project experiences and lessons learned, in: Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems, 2016, pp. 238–248.

[42] H. Solheim, F. Lillehagen, S. Petersen, H. Jorgensen, M. Anastasiou, Model-driven visual requirements engineering, in: 13th IEEE International Conference on Requirements Engineering (RE'05), 2005, pp. 421–425.

[43] T.D. Nielsen, S. Hovda, A. Fernández, H. Langseth, A. Madsen, A. Masegosa, A. Salmerón, Requirement engineering for a small project with pre-specified scope, in: NIK, 0 2014.

[44] P. Laurent, J. Cleland-Huang, Requirements-gathering collaborative networks in distributed software projects, in: 2009 Collaboration and Intercultural Issues on Requirements: Communication, Understanding and Softskills, 2009, pp. 26–30.

[45] P. Laurent, A. Steele, J. Cleland-Huang, P. Mäeder, Evaluating the effectiveness of a collaborative requirements engineering modeling notation for planning globally distributed projects, unpublished, https://api.semanticscholar.org/CorpusID:14619704, 2013.

[46] G. Liebel, E. Knauss, Aspects of modelling requirements in very-large agile systems engineering, J. Syst. Softw. 199 (2023) 111628, https://doi.org/10.1016/j.jss.2023.111628, https://www.sciencedirect.com/science/article/pii/S0164121223000237.

[47] R. Kasauli, E. Knauss, J. Horkoff, G. Liebel, F.G. de Oliveira Neto, Requirements engineering challenges and practices in large-scale agile system development, J. Syst. Softw. 172 (2021) 110851, https://api.semanticscholar.org/CorpusID:228876194.

[48] E. Knauss, G. Liebel, J. Horkoff, R. Wohlrab, R. Kasauli, F. Lange, P. Gildert, T-reqs: tool support for managing requirements in large-scale agile system development, in: 2018 IEEE 26th International Requirements Engineering Conference (RE), 2018, pp. 502–503, https://api.semanticscholar.org/CorpusID:13687446.

[49] A. Koukias, G. May, V. Vasyutynskyy, D. Nadoveza, J.C. McCarthy, M. Taisch, D. Kiritsis, Approach on analysis of heterogeneous requirements in software engineering, in: 11th IFAC Workshop on Intelligent Manufacturing Systems, IFAC Proc. Vol. 47 (7) (2013) 372–377, https://doi.org/10.3182/20130522-3-BR-4036.00088, https://www.sciencedirect.com/science/article/pii/S1474667015357037.

[50] T.-M. Hesse, B. Paech, Supporting the collaborative development of requirements and architecture documentation, in: 2013 3rd International Workshop on the Twin Peaks of Requirements and Architecture (TwinPeaks), 2013, pp. 22–26.

[51] T.D. LaToza, A.D. Lecce, F. Ricci, W. Towne, A. van der Hoek, Microtask programming, IEEE Trans. Softw. Eng. 45 (11) (2019) 1106–1124, https://doi.org/10.1109/TSE.2018.2823327.

[52] S. Saito, Y. Iimura, E. Aghayi, T.D. LaToza, Can microtask programming work in industry?, arXiv:2009.05207, 2020.

[53] P. Desfray, Model repositories at the enterprises and systems scale: the modelio constellation solution, in: 2015 International Conference on Information Systems Security and Privacy (ICISSP), 2015, pp. IS–17–IS–17.

[54] The ArchiMate Enterprise Architecture Modeling Language, https://www.opengroup.org/archimate-forum/archimate-overview, 2023.

[55] Hugo Bruneliere, Bilal Said, AIDOaRt consortium, D 3.2 - AIDOaRt core infrastructure and framework - initial version, Deliverable Ref. Ares(2022)3313183 - 29/04/2022, H2020/KDT AIDOaRt project, https://sites.mdu.se/aidoart/results/deliverables, Apr. 2022.

[56] Hugo Bruneliere, Bilal Said, AIDOaRt consortium, D 3.3 - AIDOaRt core infrastructure and framework - intermediate version, Deliverable, H2020/KDT AIDOaRt project, https://sites.mdu.se/aidoart/results/deliverables, Nov. 2022.

[57] OMG, Business process model and notation (BPMN), https://www.omg.org/bpmn/, 2023.

[58] K. Wiegers, J. Beatty, Software Requirements, Pearson Education, 2013.
[59] R. Likert, A technique for the measurement of attitudes, Arch. Psychol. (1932).
[60] A. Sadovykh, H. Bruneliere, D. Truscan, Dataset - survey results - applying model-based requirements engineering in three large European collaborative projects, https://doi.org/10.5281/zenodo.5011357, Jun. 2021.
[61] A. Sadovykh, B. Said, D. Truscan, H. Bruneliere, Dataset - survey results - applying model-based requirements engineering in AIDOaRt collaborative project, https://doi.org/10.5281/zenodo.8324734, Mar. 2023.