Franch, X. [et al.]. Towards integrating data-driven requirements engineering into the software development process: a vision paper. A: International Working Conference on Requirements Engineering: Foundation for Software Quality. "Requirements Engineering: Foundation for Software Quality, 26th International Working Conference, REFSQ 2020: Pisa, Italy, March 24–27, 2020: proceedings". Berlín: Springer, 2020, p. 135-142. ISBN 978-3-030-44429-7.
The final authenticated version is available online at https://doi.org/10.1007/978-3-030-44429-7_10

Towards Integrating Data-Driven Requirements Engineering into the Software Development Process: A Vision Paper

Xavier Franch[1] Norbert Seyff[2] Marc Oriol[1]✉ Samuel Fricker[2]
Iris Groher[3] Michael Vierhauser[3] and Manuel Wimmer[3]

[1] Universitat Politècnica de Catalunya, Spain
{franch|moriol}@essi.upc.edu
[2] University of Applied Sciences and Arts Northwestern Switzerland FHNW, Switzerland
{norbert.seyff|samuel.fricker}@fhnw.ch
[3] Johannes Kepler University Linz & CDL-MINT, Austria
{iris.groher|michael.vierhauser|manuel.wimmer}@jku.at

Abstract. *[Context and motivation]* Modern software engineering processes have shifted from traditional upfront requirements engineering (RE) to a more continuous way of conducting RE, particularly including data-driven approaches. *[Question/problem]* However, current research on data-driven RE focuses more on leveraging certain techniques such as natural language processing or machine learning than on making the concept fit for facilitating its use in the entire software development process. *[Principal ideas/results]* In this paper, we propose a research agenda composed of six distinct research directions. These include a data-driven RE infrastructure, embracing data heterogeneity, context-aware adaptation, data analysis and decision support, privacy and confidentiality, and finally process integration. Each of these directions addresses challenges that impede the broader use of data-driven RE. *[Contribution]* For researchers, our research agenda provides topics relevant to investigate. For practitioners, overcoming the underlying challenges with the help of the proposed research will allow to adopt a data-driven RE approach and facilitate its seamless integration into modern software engineering. For users, the proposed research will enable the transparency, control, and security needed to trust software systems and software providers.

Keywords: Data-Driven Requirements Engineering · Feedback Gathering · Requirements Monitoring · Model-Driven Engineering

1 Vision

Software systems have an increasingly critical role in today's society. The efficient construction, operation, and evolution of software systems to satisfy the functionality and quality that users expect is key to success. This also includes to anticipate user expectations and provide functionality and qualities that users are unaware of and cannot communicate explicitly. Moreover, software systems that adapt to context changes need to gain users' trust. New approaches such as continuous software engineering [7, 15] have the potential to successfully keep the user in the loop, which is still a challenge in the requirements engineering discipline (RE) [6].

With this new demand, the so-called data-driven RE (DDRE) has emerged [9, 10]. DDRE proposes a paradigm shift in that RE is becoming a data-centred endeavour in support of the continuous evolution of software-intensive systems. Instead of letting a requirements engineer elicit, analyze, document, and validate requirements, (a crowd of) users generate data that leads to requirements as a result. This data can be provided by users either explicitly in the form of comments, ratings, and other kinds of feedback or implicitly through usage logs and monitoring data [7]. The realization of DDRE benefits from recent technological advancements, such as machine learning (ML) and natural language processing (NLP). In contrast to traditional RE techniques, DDRE enables the continuous elicitation of requirements directly from the (crowd of) end-users using the software. Although DDRE can only be applied to eliciting requirements from existing software, and must take into account regulatory and privacy concerns, the advent of Continuous Delivery, combined with techniques for privacy management, foster the applicability of DDRE and reduce those limitations. Furthermore, it is argued that software products' success depends on user feedback [9]. For instance, a recent survey with release engineers showed that 90% believed that users' feedback has the highest importance for evaluating success and failure of mobile apps [10].

While the general idea of DDRE is clear and increasingly accepted, its impact on software development and software systems is still an open question. Maalej et al. [9] have identified three directions for future research: more sophisticated ML and NLP techniques with analytic capabilities, integration of explicit and implicit feedback, and exploitation of data in release planning. While these three areas are subject of current research (e.g., [3, 12, 16]), we deem additional research directions crucial for the success of DDRE. These are the systematic development and integration of software development and runtime infrastructure required to implement DDRE, the integration of DDRE into a continuous software engineering process, and the trust of end-users in the responsible use of their data. Some research has started, e.g., with the integration of data-driven requirements management into rapid software development [5]. However, additional effort is needed considering the ongoing shift in software technologies (e.g., Cyber-Physical Systems) and software engineering (e.g., Agile/Lean and DevOps).

In this paper, we present our vision of enabling and integrating DDRE in continuous software engineering. Section 2 describes the challenges we deem important. Section 3 outlines our research roadmap alongside an introduction of an envisioned DDRE framework. Finally, Section 4 concludes the paper.

2 Research Challenges

In recent years, researchers have proposed adaptations of traditional RE with aspects of DDRE, such as user feedback and runtime monitoring [12, 16]. However, no holistic approach has been proposed for the integration of DDRE into software engineering. This goal would require flexible processes and tools that seamlessly integrate with existing environments and development processes. In the following, we describe the challenges we deem crucial for our vision coming to fruition. The identified challenges were obtained after studying the scientific state of the art and analyzing the limitations of current approaches.

Challenge 1: Seamless integration into existing development processes and software systems. Typically, the development of DDRE components (such as feedback forms or monitoring components) is done ad-hoc without considering the information needs of the different stakeholders in the development processes. Furthermore, the evolution and adaptation of these DDRE components is not always well-coordinated and aligned with the evolution of the system itself. This co-evolution process requires a flexible and configurable DDRE infrastructure incorporating different tools and interfaces to keep the system and the DDRE components in sync.

Challenge 2: Collection, processing, and integration of relevant heterogeneous information. The combination of user data from diverse sources into a consolidated source of feedback provides semantically richer information for decision-making [10, 12, 16]. The "diverse sources" comprise those mentioned above: data collected with feedback forms (e.g., ratings, text, images, or videos), logs of user interactions with the system, and quality-of-service data gathered through runtime monitoring of the system execution (e.g., response time, invalid accesses).

Challenge 3: Context-awareness and adaptability. The contexts in which users operate with the systems may change even during runtime when the system is being used. It is necessary to adapt the DDRE infrastructure to these changes. The context comprises several facets, for instance, locations, time of the day, environmental conditions, and user profiles [2].

Challenge 4: Provision of actionable feedback. Consolidated feedback needs to be analysed to inform decision-makers about the users' experience. The increasing popularity and adoption of software analytics [11], data science [4], and visualization approaches offer novel techniques to design methods supporting DDRE. Traceability to the requirements and design artefacts of the system is key, but often missing [14].

Challenge 5: Gaining users' trust. Information obtained via monitoring and collecting feedback from users is sensitive. The information may be misused for exposure, discrimination, and even identity theft. If such information concerns business affairs, it can expose a company to business intelligence and espionage. Hence, DDRE must win the users' trust in the responsible use of the collected data. It may do so by guiding developers in ethically sound use of data and help them to comply with regulations, e.g., by respecting the human users' privacy and the corporate users' business secrets.

Challenge 6: Provision of value for the entire life-cycle. So far, DDRE has mainly focused on the utilization of user data to support requirements elicitation, covering only a fraction of a system's life-cycle. We envision the same concepts being applied during system maintenance and evolution. E.g., before putting a release in operation, collected data could be leveraged to create realistic and lifelike simulations. This way, the DDRE infrastructure may be used for multiple ends and improve the return of investment in it.

Although we deem those six challenges as crucial for integrating DDRE into software engineering, it must be acknowledged that there might be additional issues that may require further research. For instance, analyzing the limitations and pitfalls of DDRE; defining types of systems or domains for which DDRE may be difficult to apply; studying the possible combination of DDRE with traditional RE techniques; identifying the risks of misuse of DDRE (e.g., biased data or inappropriate choice of data sources); or challenges in upgrading the skills and training of RE practitioners.

3 Research Roadmap

In order to tackle the challenges presented in Section 2, we identify and elaborate respective research directions and propose a conceptual model-driven DDRE infrastructure (cf. Fig. 1). The infrastructure comprises five major parts: a family of domain specific languages (DSLs), the management of data sources, code generation, DDRE support components (such as monitoring and context-based feedback mechanisms), and components for analytics and decision support.

The first two parts are dedicated to the description and management of various different design time artifact that play a critical role in a data-driven RE process. This includes, for example, the requirements for the system, various design and context models, and monitoring or adaptation rules. In order to consolidate these diverse sources, we envision a family of DSLs facilitating the declarative description of these artifacts in a structured way. The third part of the infrastructure is dedicated to making use of these components at runtime. This is achieved by employing a model-driven approach that allows generating executable components based on the descriptions found in the DSL (e.g., monitors collecting certain information about the system). The data and user feedback collected by these components then needs to be consolidated and analysed. Finally, this will provide the foundation for the last part, supporting the decision-making process for new or changing requirements of the system. Related to these parts we have derived six distinct research challenges that drive our work on the DDRE infrastructure:

Research Direction 1: Specification and Generation of DDRE infrastructures. Bridging the gap between system development and infrastructure generation requires developing them from the same underlying basis, namely the system requirements. We think that system requirements provide valuable information to identify feedback needs and guide infrastructure development and customization. For instance, a non-functional requirement such as *"The system shall complete a user's purchase order in less than 2 seconds in 95% of the time"* points out the need for: a) logging response times, b) generating infrastructure code to aggregate all purchase order response times and check the stated condition, c) generating context-based feedback forms to be shown to users for validating the requirement, and d) mining suitable forums (blogs, twitters, ticketing systems) to gain additional information about user satisfaction or dissatisfaction. We anticipate employing a model-driven development approach [1] for generating DDRE infrastructure (cf. Fig. 1 – RD 1). A family of DSLs may allow specifying a DDRE infrastructure by, e.g., refining the requirements as formulas and linking them to the design of the system that receives the user data during runtime. An important milestone in this direction would be the availability of first DSLs for DDRE.

Research Direction 2: Embracing heterogeneous sources and feedback types. In order to collect valuable information for DDRE, the large variety of different sources and diverse types of feedback need to be taken into consideration [12, 16]. This in turn requires to identify relevant sources and to understand their information structure. Again, dedicated DSLs are needed to describe different feedback types (cf. Fig. 1 – RD 2). Such descriptions allow for the structured combination of different kinds of feedback and are the basis for any form of subsequent data analysis. Bringing together heterogeneous feedback, therefore, is key to automatically uncover hidden requirements, identify problems to be solved, and improvement opportunities to be seized. Concrete

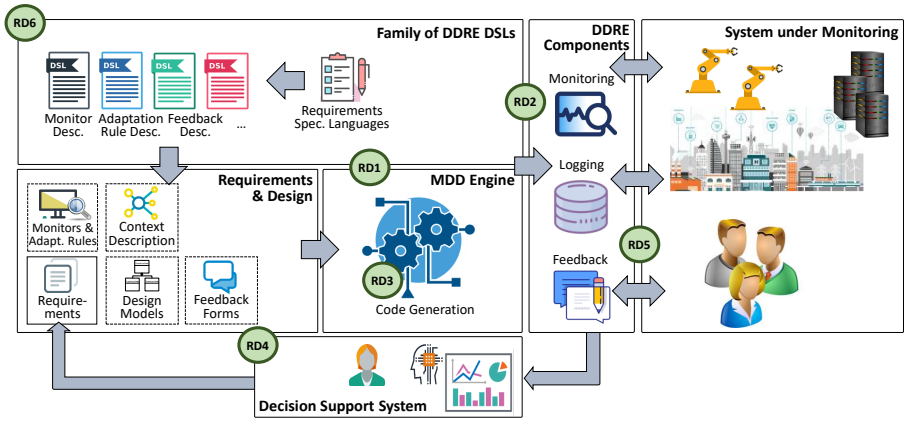

Fig. 1. Generation of the data-driven RE infrastructure: a model-driven vision.

outcomes regarding this second research direction could include advanced definitions of data sources and data sets in the form of data models. The availability of such models could be considered as important milestone in this direction.

Research Direction 3: Context-Aware Adaptation of the DDRE infrastructure. As a response to Challenge 3, we envision a certain degree of (self-)adaptability of the infrastructure to foresee, respond to, and learn about changing user contexts. This also means that this challenge includes two main aspects, context-awareness and adaptation. Analyzing usage contexts and being aware of the capabilities of DDRE components, we expect so-called context-awareness patterns to emerge that can be used to generate context-related code (cf. Fig. 1 – RD 3). These patterns will be bound to certain context dimensions and provide the necessary input for the adaptation of DDRE components in order to ensure the effective and efficient collection of data. Furthermore, the adaptation is needed to ensure that users' data is gathered accurately, efficiently, and in a non-intrusive way. For example, when a mobile device is running out of battery (a context change) the monitoring sampling rate could be reduced or monitoring could even be temporarily deactivated (adaptation of the DDRE components). This pattern may be always applied in systems deployed on a mobile device. The provision of advanced context models, the definition of context-awareness patterns and actual code generation to ensure the adaptation are important milestones for this research direction.

Research Direction 4: Advanced Data Analysis capabilities and Decision Support Systems. We foresee the extensive use of analytics tools (such as SonarQube [13]) fed with the data collected by the DDRE infrastructure. Analytics may include indicators about customer satisfaction, risk, or time-to-market. They inform decision-makers and guide the evolution of the system by triggering requirement changes (cf. Fig. 1 – RD 4). As we perceive requirements to be the source for building the DDRE infrastructure, requirement changes also have an impact on this infrastructure. This self-adaptation enabling loop means that the infrastructure can co-evolve with the monitored system even at runtime. Providing first prototypes of DDRE infrastructures which are capable of co-

evolving with the system itself, based on requirements for the system can be considered as key milestone here.

Research Direction 5: Ensuring Users' Trust in DDRE. Building and maintaining user trust requires an ethically and legally sound approach for collecting and processing data (cf. Fig. 1 – RD 5). In particular, DDRE should support privacy and business secrecy laws, such as the European General Data Protection Regulation 2016/679 (GDPR) and Trade Secrets Directive 2016/943. These will affect the DDRE technical architecture and DSL, the software lifecycle processes benefiting from DDRE, and the organisational structure of the data processors and users. Several aspects will need to be considered, such as purposeful data minimisation and safeguarding, end-user data governance with dynamic consent, and mobility of the collected data. Furthermore, any DDRE approach will need to be evaluated in terms of privacy and trust impact and in its ability to unlock data for supporting decisions in the software process. In general, a better understanding of user's trust in the context of DDRE and the documentation of this understanding, e.g., in the form of trust models can be considered important milestones.

Research Direction 6: Processes Integration of DDRE into existing software development lifecycles. DDRE needs to be smoothly integrated and exploited in rapid, even continuous software development (cf. Fig. 1 – RD 6). The support of all relevant activities in the end-to-end process should be studied, starting with the identification of user needs and ending with the addition of requirements to the product backlog. Knowing how and when data analysis is performed and by whom allows understanding the implications of DDRE in the software process. Furthermore, the DDRE infrastructure has to be validated before going into production. This is challenging considering the context-dependent nature of the infrastructure, which calls for a component able to generate contexts that are part of the infrastructure-testing process. Finally, DDRE needs to be aligned with existing paradigms and methods in software engineering and business modelling. E.g., online controlled experimentation is one such relevant method [8] that uses collected usage data, here for evaluating different implementations of a feature. The data-driven nature of such paradigms and methods calls for the exploration of possible synergies. In the previous paragraph, we have highlighted several important steps towards process integration of DDRE which represent key milestones in this regard.

4 Conclusion

Recent research in DDRE contributes important pieces of the puzzle. To be valuable and useful in real-world applications, these pieces need to be arranged so that they fit together seamlessly and automation possibilities need to be explored. This is particularly true for current user-driven approaches that are already delivering value to software producers but also face challenges which we have outlined in this paper. Although we expect DDRE to have a major impact on RE in the near future, RE as we know it will still be necessary when it comes to the development of entirely new systems where experiences and data cannot be sufficiently leveraged to "generate" requirements. Furthermore, DDRE does not limit creative developers but is intended to offer means that empower creativity. We are convinced that this novel RE paradigm will increase the

effectiveness of RE, improve software quality, and eventually will help to increase the trust of users in software applications.

Acknowledgements. This work has been supported by: the Spanish project GENESIS (TIN2016-79269-R), the Christian Doppler Forschungsgesellschaft, the Austrian Federal Ministry for Digital and Economic Affairs, the National Foundation for Research, Technology and Development, and the Austrian Science Fund (FWF) under the grant numbers J3998-N31, P28519-N31, and P30525-N31.

References

1. Brambilla, M., Cabot, J., Wimmer, M.: Model-driven software engineering in practice. Morgan & Claypool Publishers, 2nd edn. (2017)
2. Cabrera, O., Franch, X., Marco, J.: 3LConOnt: a three-level ontology for context modelling in context-aware computing. Software & Systems Modeling **18**(2), 1345–1378 (2019)
3. Dąbrowski, J., Letier, E., Perini, A., Susi, A.: Finding and analyzing app reviews related to specific features: A research preview. In: Proc. of REFSQ. pp. 183–189. Springer (2019)
4. Ebert, C., Heidrich, J., Martinez-Fernandez, S., Trendowicz, A.: Data science: Technologies for better software. IEEE Software **36**(6), 66–72 (2019)
5. Guzmán, L., Oriol, M., Rodríguez, P., Franch, X., Jedlitschka, A., Oivo, M.: How can quality awareness support rapid software development?–a research preview. In: Proc. of REFSQ. pp. 167–173. Springer (2017)
6. Jarke, M., Loucopoulos, P., Lyytinen, K., Mylopoulos, J., Robinson, W.: The brave new world of design requirements. Information Systems **36**(7), 992–1008 (2011)
7. Johanssen, J.O., Kleebaum, A., Bruegge, B., Paech, B.: How do practitioners capture and utilize user feedback during continuous software engineering? In: Proc. of RE (2019)
8. Lindgren, E., Münch, J.: Raising the odds of success: the current state of experimentation in product development. Information and Software Technology **77**, 80–91 (2016)
9. Maalej, W., Nayebi, M., Johann, T., Ruhe, G.: Toward data-driven requirements engineering. IEEE Software **33**(1), 48–54 (2015)
10. Maalej, W., Nayebi, M., Ruhe, G.: Data-driven requirements engineering: an update. In: Proc. of ICSE/SEIP. pp. 289–290. IEEE (2019)
11. Martínez-Fernández, S., Vollmer, A.M., Jedlitschka, A., Franch, X., López, L., Ram, P., Rodríguez, P., Aaramaa, S., Bagnato, A., Choraś, M., Partanen, J.: Continuously assessing and improving software quality with software analytics tools: A case study. IEEE access **7**, 68219–68239 (2019)
12. Oriol, M., Stade, M., Fotrousi, F., Nadal, S., Varga, J., Seyff, N., Abello, A., Franch, X., Marco, J., Schmidt, O.: FAME: supporting continuous requirements elicitation by combining user feedback and monitoring. In: Proc. of RE. pp. 217–227. IEEE (2018)
13. SonarQube: `https://www.sonarqube.org` (accessed 24-01-2020)
14. Vierhauser, M., Cleland-Huang, J., Burge, J., Grünbacher, P.: The interplay of design and runtime traceability for non-functional requirements. In: Proc. of the 10th Int'l Workshop on Software and Systems Traceability. pp. 3–10. IEEE (2019)
15. Villela, K., Groen, E.C., Doerr, J.: Ubiquitous requirements engineering: A paradigm shift that affects everyone. IEEE Software **36**(2), 8–12 (2019)
16. Wüest, D., Fotrousi, F., Fricker, S.: Combining monitoring and autonomous feedback requests to elicit actionable knowledge of system use. In: Proc. of REFSQ. pp. 209–225. Springer (2019)