Reengineering of Information Systems toward Classical-Quantum Systems

Ricardo Pérez-Castillo[0000-0002-9271-3184]

Social Science & Information Technology Faculty, University of Castilla-La Mancha
Av. Real Fábrica de Sedas, s/n, Talavera de la Reina, 45600, Spain
ricardo.pdelcastillo@uclm.es

Abstract. In the coming years, companies will progressively need to add quantum computing to some or all of their daily operations. It is clear that all existing, classical information systems cannot be thrown away. Instead of this, it is expected to add some quantum algorithms working embedded in classical information systems. So far, there is not a systematic solution to deal with this challenge. Thus, this talk suggests a software modernization approach (model-driven reengineering) for restructuring classical systems together with existing or new quantum algorithms to provide target systems combining both computational paradigms. The method highlighted is systematic and based on existing software engineering standards (such as KDM and UML). As a result, it could be applied in industry in a compliant manner regarding the existing software evolution processes.

Keywords: Quantum computing, reengineering, software modernization.

1 Introduction

The quantum momentum is today stablished. QC is becoming more and more a mature area while an investment scalation is happening in both, public and private sectors [1, 2]. Thus, the effective quantum supremacy [3] is expected for the next few years, i.e., when quantum computers are able to solve problems that classical computer cannot in practice. Actually, that QC is a transversal and interdisciplinary opportunity for digital transformation and social impact [4], with multiple applications, for example, biomedical simulations and disease diagnosis, machine learning, optimization problems such as logistics, financial modelling and risk management, chemical modelling, cybersecurity and cryptography, among many other.

In the last years, quantum physics, mathematics, computers, algorithms and, in general, quantum computer science present certain progress. Despite QC is becoming more and more mature and mainstream, software engineering has not been considered in depth for quantum software as it is for classical software during last decades [5, 6].

In particular, in this work we point out the problem of model-driven reengineering [7]. We believe, most organization will demand the migration of their first quantum algorithms or future ones and its integration with the existing enterprise, classical information systems. Obviously, in a short-term basis, quantum computers will not be used for everything (among other things because of its initial prohibitive cost). Instead of this, it will be more common to use quantum computer to solve certain hard problems through specific calls from classical computers to remote quantum computers in the cloud. In this scenario, software modernization processes have proven to be an effective mechanism to migrate and evolve software while business knowledge is preserved [8].

In particular, this research proposes a software modernization approach (model-driven reengineering) for restructuring classical systems together with existing or new quantum algorithms to provide target systems combining both, classical and quantum information systems. The solution proposed is systematic and based on existing, well-known standards like Unified Modelling Language (UML) [9] and Knowledge Discovery Metamodel (KDM) [10].

The main implication of this contribution is the technical and economic impact derived of the possibility of reusing the knowledge embedded in legacy information systems while the effort of developing new quantum information systems is reduced. Also, since this proposal is based in international standards to represent knowledge in an agnostic manner, the independence regarding quantum programming languages is achieved. As a result, the application of this proposal is feasible in the volatile environment expected during the first stages of the QC industry.

2 Reengineering of classical systems

Despite the fact that legacy systems may be obsolete, this kind of system usually has a critical mission within the company and represents a valuable asset for companies, since legacy systems embed a lot of business logic and business rules that are not present elsewhere [11]. As a result, in spite of the upcoming quantum revolution, the companies cannot discard their legacy systems.

Reengineering has been a successful practice in the software industry. It consists of three phases: reverse engineering, restructuring and forward engineering. More than half of the traditional reengineering projects fail when dealing with specific challenges because of a lack of standardized and automated processes [12]. Firstly, standardization constitutes a problem since the reengineering process has been typically carried out in an *ad hoc* manner [8]. Thus, reengineering projects must focus their efforts on a better definition of the process. Furthermore, the code cannot be the only software asset that the standardization covers, since *"the code does not contain all the information that is needed"* [13]. The reengineering process must be formalized to ensure an integrated management of all of the knowledge involved in the process such as source code, data, business rules, and so on. Secondly, automation is also a very important problem. In order to prevent failure in large complex legacy systems, the reengineering process must be more mature and repeatable [14]. In addition, the reengineering process needs to be aided by automated tools so that companies can handle the maintenance cost [12]. Moreover, automation can be considered as a problem derived from the standardization problem, since standardization and the formalization of the process are necessary requirements to provide tools to automate the process, which can be reused for several reengineering projects.

The software modernization paradigm, and particularly ADM as defined by the OMG, can be considered as a mechanism for software evolution, i.e., it makes it possible to modernize the legacy information systems and eradicates, or at least minimizes, the software erosion problem in legacy systems. [15]. This approach is aligned with the low-code paradigm [16], the last trend in enterprises for which sophisticated platforms are employed for generating new code for their applications, Thus, there are progressively fewer use cases in which organizations must hand-code anything. PIM with the details that specify how that system uses a particular type of platform or technology.

ADM facilitates the reverse engineering stage by means of Knowledge Discovery Metamodel (KDM) [7], since this standard makes it possible to represent all software artefacts involved in a certain legacy system in an integrated and standardized way. The KDM standard is used to represent all the involved software artefacts (i.e., source code, databases, user interactions, etc.), and KDM achieves this in an integrated and technological-independent manner. Thus, it is possible to have a common KDM repository that is gradually completed with knowledge discovered through the analysis of different artefacts in the legacy systems. KDM can be compared with the UML standard (ISO/IEC 19505) [17]: While UML is used to generate new code in a top-down manner, a process involving KDM starts from the existing code and builds a higher level model in a bottom-up manner [18].

3 Reengineering of and toward Quantum Systems

Apart from differences between quantum and classical software, new software systems will probably integrate classic and quantum computation, since all kind of problems are not suitable to be addressed from a pure quantum point of view. Instead of this, future software will include some pieces of code in classical programming languages that perform calls to quantum algorithm that are executed in quantum computers. Software modernization and reengineering practices must be brought into the domain of QC. Thus, reengineering has to be revisited to deal with the problems associated with the expected QC migrations and the next coexistence of classical and quantum software.

We propose a software modernization based on existing standards such as UML and KDM. Regarding KDM, if reverse engineering of classical systems (plus quantum programs, if any) is carried out and the extracted knowledge is holistically represented in a KDM repository, then reengineering and migration towards quantum environments is improved. This means that the previous knowledge and business rules is preserved, and the impact of the integration of quantum programs is limited. Concerning UML, the standard must be extended (through the standard mechanisms) for representing and integrating quantum programs. As a result, KDM models can be automatically transformed into UML representations, and/or engineers can manually model quantum aspects for new, target systems.

4 Implications for Researchers and Practitioners

Today, QC is at an important inflection point. High-level algorithms for Quantum computers have shown considerable promise in the last years, and recent advances in QC device fabrication is increasing its utility. Nevertheless, a gap still exists between the hardware size and reliability requirements of QC algorithms and the physical machines foreseen within the next ten years. To bridge this gap, Quantum computers require appropriate software to translate and optimize applications (tool flows) and abstraction layers [5]. The future quantum developer will not be expected to have such an in-depth expertise, just as modern-day programmers have for the most part a limited knowledge of hardware issues [19]. We believe this proposal will contribute in this regard.

If quantum software is already a key concern, the software engineering field for QC will become even more critical in the near future. Concerns like quality assurance, project management, testing, continuous integration and delivery that have been considered during decades for designing, implementing and delivering classical software, must be handled for building quantum software today. Among these concerns, software modernization will gain certain relevance since new hybrid systems will considers problems like migrating software, integrating quantum algorithm into classical systems, preserving knowledge on reengineering, and so forth.

As we stated in our proposal, the usage of well-known standards in the area of software engineering can help to bring those best practices and methods to the new QSE field. In our proposal we exposed how KDM and UML can help in software modernization process by abstracting knowledge and contributing to systematic model-driven reengineering processes. However, the usage of other standards to other areas of software engineering could be beneficial. Of course, new standards will be released in the context of QC and QSE. Nevertheless, current standards that have been successfully applied for classical software systems during years, can be used to get some lessons learned and may still provide interesting contributions.

Finally, it is clear that investment and expectations on QC are growing year by year. Fig. 1 provides the expending on quantum-technology by countries, which shows that QC is global.

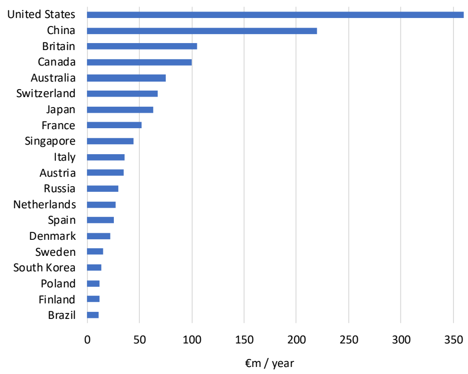

Fig. 1. Annual spending on non-classified quantum-technology research, €m [20].

Those numbers present the investment up to 2017, however current data probably exceeds in a greater extend numbers presented in Fig. 1. Outside of these numbers, United States launched in 2018 a national quantum initiative which authorizes $1.275 billion over five years for research. UE also started in 2018 to build a quantum computer with up to 100 qubits and high-precision operations through the *OpenSuperQ* project with a budget of €10.33 Mio.

A recent report by Gartner [21] states that by 2023, 20% of organizations will be budgeting for quantum computing projects. About profitability, according to [1], gains will grow first to companies in sectors with complex simulation and optimization requirements. It will be a gradual progress for the next few years: we anticipate value for end users to reach a relatively modest $2 to $5 billion by 2024. But these values will then increase ranging between 5 or 10 times as the technology and its commercial viability mature (with the advent of enhanced error correction and modular architectures, this last aligned with our proposal).

5 Conclusions

This paper has pointed out the quantum computer science momentum and claims the need for researching and developing the quantum software engineering field. In our vision, quantum physics, mathematics, computers, algorithms and quantum computer science present certain progress. However, in our opinion, quantum technologies and programming have not yet been addressed with techniques, good practices and development methodologies of software engineering to meet quantum programs' needs. In order to reduce this gap, this paper proposes a software modernization process, i.e., a model-driven reengineering process, to cope with the migration of quantum algorithms together with classical, legacy systems; as well as to address the integration of new quantum software during modernization of classical, legacy systems while knowledge is preserved.

The solution proposed is systematic and based on existing software engineering standards such as KDM and UML. As a result, it could be applied in industry in a systematic way and in a compliant manner regarding the existing software evolution processes. It is probably the first time that the software modernization process is specifically created and/or adapted for quantum technologies.

This proposal will allow companies to reuse the knowledge embedded in legacy information systems while new quantum-based projects are delivered. Thanks to the usage of KDM and UML, this proposal is independent on quantum programming languages, which makes its application feasible in the volatile technological environment expected during the quantum computing revolution.

Acknowledgements

This study has been funded by the BIZDEVOPS-Global (RTI2018-098309-B-C31) project, Ministerio de Economía, Industria y Competitividad (MINECO) & Fondo Europeo de Desarrollo Regional (FEDER).

References

1. Langione, M., C. Tillemann-Dick, A. Kumar, and V. Taneja, *Where Will Quantum Computers Create Value—and When?* Boston Consulting Group. https://www.bcg.com/ en-es/publications/2019/quantum-computers-create-value-when.aspx, 2019: p. 19.
2. Mohseni, M., P. Read, H. Neven, S. Boixo, V. Denchev, R. Babbush, A. Fowler, V. Smelyanskiy, and J. Martinis, *Commercialize quantum technologies in five years.* Nature News, 2017. 543(7644): p. 171.
3. Harrow, A.W. and A. Montanaro, *Quantum computational supremacy.* Nature, 2017. 549: p. 203.
4. López, M.A. and S.M.D. Silva, *Tecnologías Cuánticas. Una oportunidad transversal e interdisciplinar para la transformación digital y el impacto social*. 2019, Banco Interamericano de Desarrollo (BID). p. 97.
5. Chong, F.T., D. Franklin, and M. Martonosi, *Programming languages and compiler design for realistic quantum hardware.* Nature, 2017. 549(7671): p. 180.
6. Clark, J. and S. Stepney, *Proposed "Grand Challenge for Computing Research" Quantum Software Engineering* in *3rd meeting, Quantum Computing SIG*. 2002, University of York.
7. OMG, *Architecture-Driven Modernization (ADM): Knowledge Discovery Meta-Model (KDM), v1.4. https://www.omg.org/spec/KDM/1.4/PDF*. 2016, OMG. p. 372.
8. Pérez-Castillo, R., I.G.-R. de Guzmán, and M. Piattini, *Business process archeology using MARBLE.* Information and Software Technology, 2011. 53(10): p. 1023-1044.
9. ISO/IEC, *ISO/IEC 19505-1:2012. Information technology — Object Management Group Unified Modeling Language (OMG UML) — Part 1: Infrastructure. https://www.iso.org/standard/32624.html*. 2012, ISO/IEC. p. 220.
10. ISO/IEC, *ISO/IEC 19506:2012. Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM). https://www.iso.org/standard/32625.html*. 2012, ISO/IEC. p. 331.
11. Sommerville, I., *Software Engineering*. 9 th Edition ed. 2010: Pearson. 792.
12. Sneed, H.M., *Estimating the Costs of a Reengineering Project*. Proceedings of the 12th Working Conference on Reverse Engineering. 2005: IEEE Computer Society. 111 - 119.
13. Müller, H.A., J.H. Jahnke, D.B. Smith, M.-A. Storey, S.R. Tilley, and K. Wong, *Reverse engineering: a roadmap*, in *Proceedings of the Conference on The Future of Software Engineering*. 2000, ACM: Limerick, Ireland.
14. Canfora, G. and M.D. Penta, *New Frontiers of Reverse Engineering*, in *2007 Future of Software Engineering*. 2007, IEEE Computer Society.
15. OMG, *Why do we need standards for the modernization of existing systems?* 2003, OMG ADM Task Force.
16. Araujo, C. *Is it the end of coding in the enterprise?* ITSM & THE DIGITAL ENTERPRISE 2017 23/09/2019]; Available from: https://www.cio.com/article/3242253/ is-it-the-end-of-coding-in-the-enterprise.html.
17. OMG. *UML (Unified Modeling Language). Superstructure specification. Version 2.2. http://www.omg.org/spec/UML/2.2/Superstructure/PDF/.* 2009 02-02-2009 24-06-2009]; Available from: http://www.omg.org/spec/UML/2.2/Superstructure/PDF/.
18. Moyer, B. *Software Archeology. Modernizing Old Systems*. Embedded Technology Journal, 2009. 1, 1-4.
19. Yanofsky, N.S. and M.A. Mannucci, *Quantum computing for computer scientists*. 2008: Cambridge University Press.

20. Palmer, J., *Quantum technology is beginning to come into its own.* The Economist https://www.economist.com/news/essays/21717782-quantum-technology-beginning-come-its-own, 2017.
21. Panetta, K., *The CIO's Guide to Quantum Computing*. 2019, Gartner.