Repositories, Contributors, and Continuity: An Empirical Study of Foundational Quantum Software

Vincent Gierisch*
Technical University of Applied Sciences Regensburg
Regensburg, Germany
vincent.gierisch@othr.de

Nicole Hoess*
Technical University of Applied Sciences Regensburg
Regensburg, Germany
nicole.hoess@othr.de

Ralf Ramsauer*
Technical University of Applied Sciences Regensburg
Regensburg, Germany
ralf.ramsauer@othr.de

Wolfgang Mauerer
Technical University of Applied Sciences Regensburg
Siemens AG, Technology
Regensburg/Munich, Germany
wolfgang.mauerer@othr.de

*Authors contributed equally

***Abstract*—Driven by contributions from academia, industry, and open-source communities, the quantum software ecosystem is rapidly growing. Across this ecosystem, new concepts often emerge through software artefacts accompanying scientific publications as well as through sustained development in larger communities. However, many repositories receive development efforts only over a limited period of time, raising the question whether their concepts persist beyond individual repositories.**

In this paper, we apply established empirical software engineering techniques to analyse a set of foundational quantum software repositories. We combine cross-repository activity with contributor relationships to study the evolution of communities. Our analysis provides empirical evidence of contributor migration patterns and indications of cross-project knowledge transfer. We observe multiple development paths: projects may evolve into sustained communities, contributors may integrate concepts into established ecosystems, or activity may continue through new and follow-up software artefacts. Our observations provide an initial empirical perspective on how concepts and influence persist across repository boundaries in quantum software ecosystems.

***Index Terms*—Quantum Software Engineering, Mining Software Repositories, Software Ecosystems, Contributor Analysis**

I. Introduction

Quantum computing has led to the emergence of a rapidly growing software ecosystem that comprises applications, high-level programming abstractions, compiler stacks, intermediate representations, as well as software for pulse execution and control. Research-driven developments play a central role in this ecosystem. Frequently, scientific publications introduce new concepts through accompanying software artefacts that serve as standalone proof of concept implementations. At the same time, larger community-driven efforts contribute to sustained *long-term maintained* software. However, many repositories receive development effort only over a limited period of time. This raises the question if concepts of *short-lived* projects are transferred into long-term maintained development efforts.

In the field of software engineering (SE), empirical analysis of software ecosystems has long been established as a means to study repository evolution [1], project activity [2], community dynamics [3], and long-term sustainability [4]. Within SE, Mining Software Repositories (MSR) [5] techniques provide perspectives that extend beyond functional or performance-oriented evaluation. Such perspectives are also beginning to gain traction in quantum computing [6], with recent work [7]–[11] studying repository evolution and maintenance, software patterns and quality characteristics in quantum projects.

While these developments establish MSR as an emerging perspective for quantum software, existing analyses primarily characterise isolated repositories and their observable development artefacts. Less attention has been paid to relationships *across* repositories and to whether the influence of projects on the ecosystem persists beyond periods of active development.

In this work, we combine repository activity with contributor relationships across repositories to analyse whether and how development activity and influence extend beyond individual projects. Therefore, we study a subset of foundational repositories[1] from quantum computing compiler projects spanning research-driven and community-driven development.

Our analysis provides empirical evidence of contributor migration patterns and indications of cross-project transfer and continuation of concepts. We observe multiple development paths: repositories may evolve into sustained communities, contributors may carry development efforts into established ecosystems, or development may continue through new and follow-up software artefacts. This represents an encouraging signal for quantum software researchers: although cross-project knowledge transfer can still be improved, the first pathways for research to make an impact already exist. Overall, we make the following contributions:

- We construct a dataset of foundational quantum software repositories and analyse their development dynamics using established empirical software engineering techniques.
- We provide a network representation of the quantum ecosystem to quantify contributor relationships across repositories and identify migration patterns between research-driven and sustained community ecosystems.
- We show indications that repository inactivity does not necessarily imply disappearing influence and discuss cross-project continuation beyond the original repository.

[1]Related publications: arXiv:2010.03935, arXiv:2101.11365, 10.1109/DAC56929.2023.10247886, 10.21105/joss.07478, arXiv:2604.08674, 10.21105/joss.06720, arXiv:1811.04968, arXiv:2101.11030, 10.1145/3183895.3183901, arXiv:2405.08810, arXiv:2408.06469, arXiv:2510.11420, arXiv:2109.02409, arXiv:2005.13283, arXiv:2404.14299, arXiv:2404.12603.

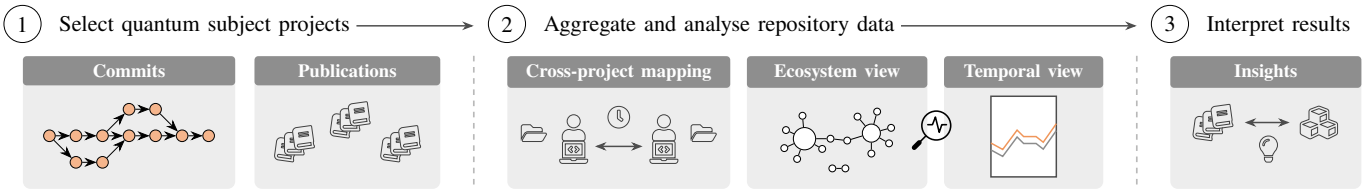

Fig. 1. Overview of the analysis workflow. We first relate quantum software repositories to associated publications and derive repository-level observations from the development history. We then map contributor identities across repositories and construct complementary ecosystem and temporal views to capture relations and evolution beyond individual projects. This allows for analysing activity, contributor dynamics, and the continuation of concepts in the ecosystem.

- We provide a self-contained reproduction package [13], [14] with all data, scripts and the execution environment.[2]

II. Methodology

Our investigation of cross-project influence in the quantum software ecosystem builds on the premise of *developer-mediated knowledge transfer*. A developer who gains expertise within one project may later carry that technical know-how into others [15], [16], whether by joining an established community, or starting a new initiative. Therefore, we hypothesise that the same developer will at some point be an active contributor in multiple projects. To find such knowledge flows, we follow the multi-stage MSR pipeline illustrated in Fig. 1.

a) Subject project selection: We start with a structured project selection based on expert knowledge to identify a subset of software projects related to quantum compilation from academic publications and official repositories. Projects were included if they provide a publicly accessible implementation of at least one quantum-specific compilation function, while considering diverse properties shown in Table I, columns 2–4.

[2]https://github.com/lfd/qset26-reproduction, 10.5281/zenodo.21628842

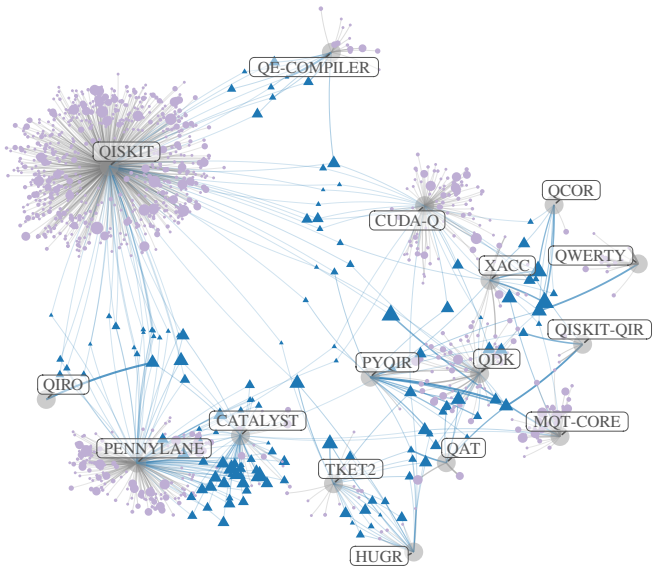

Fig. 2. Overview of the quantum software ecosystem with projects (grey nodes) and contributors (blue nodes and edges for developers contributing to multiple projects; lilac nodes and edges for developers contributing to a single project). Developer nodes are scaled by their total commits across the entire ecosystem. Edge widths indicate the share of commits (effort) a developer contributed to the specific project during their active months.

TABLE I
Subject project statistics and ecosystem metrics.

Project	Commits	LoC[k]	Team Size	Active Development Time	Exclusive Dev. Ratio	Cross-Project Dev. Ratio	Core Devs.	Peripheral Devs.	Median Projects/Dev.	Linked Projects
CATALYST	2,306	156	62	2023–2026	0.21	0.79	13	49	2.00	5
CUDA-Q	2,777	310	105	2023–2026	0.81	0.19	10	95	1.00	12
HUGR	1,719	108	23	2023–2026	0.22	0.78	5	18	2.00	5
MQT-CORE	3,476	136	45	2019–2026	0.89	0.11	4	41	1.00	5
PENNYLANE	6,674	537	223	2018–2026	0.70	0.30	25	198	1.00	5
PYQIR	295	19	22	2021–2026	0.18	0.82	3	19	2.00	11
QAT	194	51	8	2021–2023	0.25	0.75	2	6	2.50	6
QCOR	1,217	90	8	2018–2022	0.12	0.88	2	6	2.00	6
QDK	2,179	624	64	2023–2026	0.80	0.20	10	54	1.00	8
QE-COMPILER	244	28	22	2022–2024	0.59	0.41	8	14	1.00	2
QIRO	76	7	1	2020	0.00	1.00	1	0	4.00	3
QISKIT	9,341	363	661	2017–2026	0.93	0.07	33	628	1.00	11
QISKIT-QIR	78	3	6	2022–2025	0.50	0.50	3	3	1.50	4
QWERTY	398	49	13	2024–2026	0.92	0.08	1	12	1.00	4
TKET2	911	123	30	2022–2026	0.40	0.60	4	26	2.00	5
XACC	2,140	1,580	24	2016–2023	0.58	0.42	2	22	1.00	8

b) Data extraction and preprocessing: To measure *project activity* as the frequency of commits contributed by developers over the project's lifetime, we extract the development history of each project from the version-control system (VCS) `git` using the MSR tool `perceval` [17]. We filter commits on each project's main branch to exclude development efforts that have not (yet) been integrated. In addition, we filter bots based on manually identified name patterns and exclude merge commits, as both primarily capture automated or integration events rather than direct development activity.

c) Cross-project mapping: To analyse developer-induced knowledge flows at an ecosystem level, we must ensure that the same developer can be identified in all projects, even when using different e-mail addresses or nicknames. For identity matching, we consolidate all projects' contributors by merging authors and committers that share one of the following: full name *and* e-mail address, e-mail address, GitHub username or exact name. To ensure correctness, we manually sanity-check the matches.

d) Ecosystem network construction: Social networks of open-source software (OSS) communities are an established means to analyse collaboration [1], [3], [18] and knowledge transfer: If a developer of one community is connected to developers of another community, he may contribute knowledge which the other community absorbs [16]. Following this premise, we construct an ecosystem network by linking the merged developer identities to all projects they contributed to

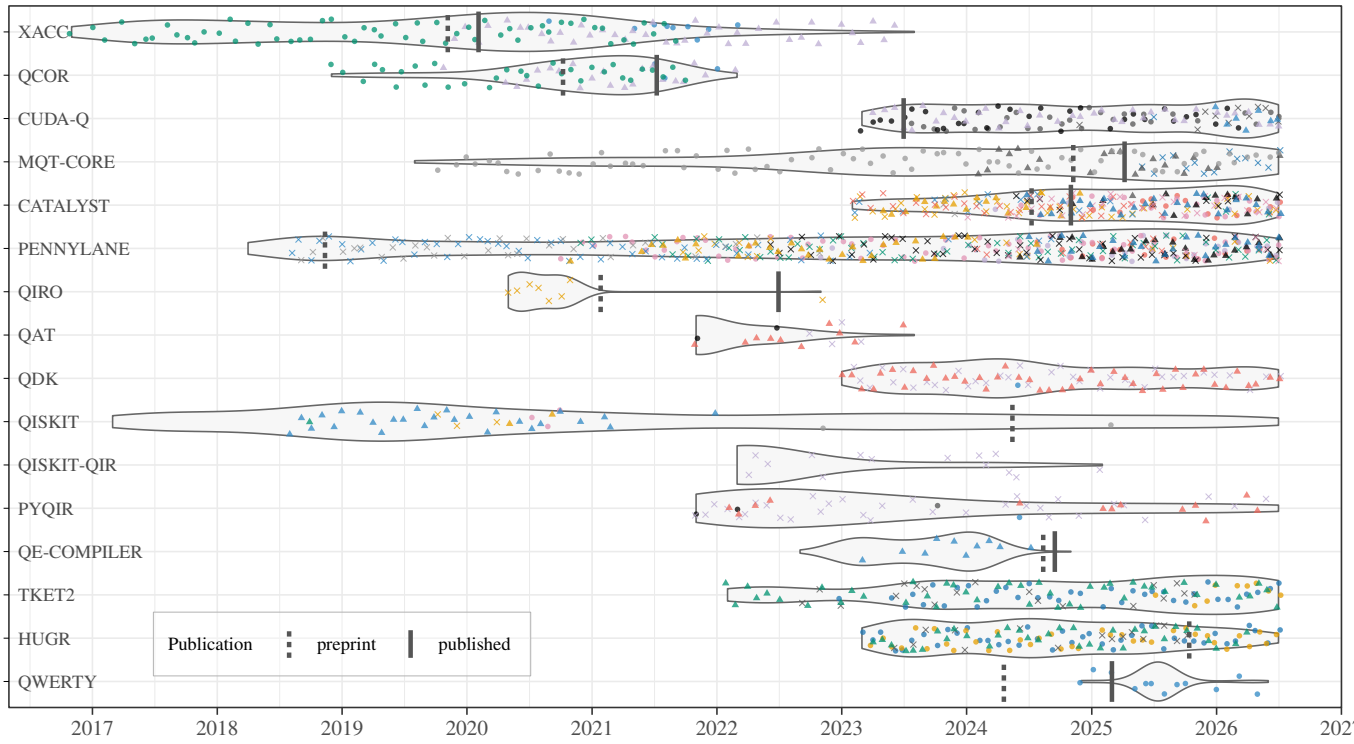

Fig. 3. Contribution activity over time. Violins show the overall commit distribution per project. Coloured points mark the top 20% (24 of 118 cross-project) contributors by total commits. Each of the colour–shape combinations denotes a single developer, allowing the same contributor to be traced across projects.

and calculating ecosystem metrics capturing developer roles (exclusive vs. cross-project developers, core developers responsible for at least 80% of commits vs. peripheral developers) and structural properties based on links.

e) Temporal analysis: Based on the static network overview, we analyse the dynamics of cross-project contributions in terms of their temporal order and direction. We count commits per developer, project and day and group results on a monthly basis to visualise cross-project contributor activity over time. This allows for identifying turnover events, in which developers end their activity in one project and possibly join another one. To understand the relationship between turnover, knowledge flows and scientific publication activities, we further annotate our data with project-specific preprint and publication dates collected from digital libraries and repositories.

III. Results / Evaluation

a) Ecosystem structure: Fig. 2 shows the ecosystem network with all developers and the projects they contribute to, while Table I presents complementary quantitative metrics. First, in most repositories, the majority of contributors remain active exclusively within their own project. Despite these largely differing developer bases, no project forms an entirely isolated community but has at least some cross-project contributors, representing candidate connections for knowledge transfer.

Established community-driven projects with an industry backbone, such as CUDA-Q, PENNYLANE, and QISKIT, benefit from their large developer base: more core developers share 80% of commits, and relative efforts of developers spread more equally. Several connections emerge across projects initiated by different companies, such as Xanadu's CATALYST, IBM's QISKIT and Microsoft's QDK, possibly indicating joint integration efforts. Mediated by individual developers, community-driven projects are also connected to academic projects, such as QIRO, QE-COMPILER, QCOR and XACC, suggesting that the larger communities support research activities or absorb know-how.

Several short-lived projects, such as HUGR, QAT and QIRO, show activity over only one or two years and were driven by one or a few developers, yet show high ratios of cross-project contributors, suggesting their developers' experience transferred elsewhere. Connections from these to active projects, as observed for MQT-CORE, PYQIR and QWERTY, could indicate parallel developments, collaborations on emerging concepts, follow-up and related research, or integration tasks. Relatedly, top contributors, by both ecosystem commits and cross-project links, often carry much of the development effort in research projects, while also contributing to industry-initiated ones.

b) Temporal analysis: The contribution activity in Fig. 3 indicates that repository activity alone does not fully capture the continued influence of research-driven quantum software artefacts. Fig. 3 provides an indication that know-how in the analysed ecosystem often moves across repository boundaries. Contributors active in two or more projects connect earlier, smaller repositories with later, larger projects that are often driven by enterprises. A comparable overlap is observed between XACC and QCOR, which share several contributors. This

relationship is plausible, as both projects originated from the same research centre [12], [19]. Several other prominent contributor overlaps can likewise be explained by shared institutional origins or affiliations. This applies to CATALYST/PENNYLANE, QISKIT/QISKIT-QIR/QE-COMPILER, HUGR/TKET2, and QAT/PYQIR. Given the shared institutional origins, knowledge transfer is likely.

Of particular interest is the contributor overlap among XACC, QCOR, and CUDA-Q. The temporal order of contributions suggests that developers previously active in XACC and QCOR subsequently contributed to CUDA-Q. This development trajectory indicates that expertise acquired in the earlier projects may have informed subsequent work on CUDA-Q. A further example of this pattern is the transition from QIRO to CATALYST, in which activity in the earlier research prototype is followed by contributions to CATALYST and PENNYLANE by the same developer. A manual verification of the commits provides additional support for the interpretation that know-how related to intermediate representations was transferred.[3]

IV. Discussion and Conclusion

Our study reveals initial patterns of knowledge transfer and sustainability in the quantum ecosystem: evolution into long-term maintenance, novel concepts move into such sustained communities or follow-up activity in new projects. Especially the second pattern is encouraging for researchers. Although the ecosystem is dominated by major vendors, long-term research creates impact by concept integration into established frameworks, boosted by publishing code and reproduction packages alongside papers. Mauerer et al. [13], [14] provide actionable guidelines.

Yet many efforts remain within separate communities, which leaves possibilities to improve knowledge transfer in the quantum ecosystem, as established ecosystems such as Linux demonstrate. Examining development processes, design abstractions and patterns in depth could identify concepts for cross-project transfer and standardisation, reducing duplicated efforts and accelerating development.

Generalisability of our exploratory study is incomplete: we detect knowledge flows from commit activity without other channels, and rely on MSR techniques with known threats [20]. Additional data and techniques would likely reveal further examples rather than undermine the reported results.

We suggest a follow-up study including developer interviews to recover motivations that technical data cannot fully explain. We hope that this study provides a starting point for discussions of how knowledge transfer in the quantum software ecosystem currently works, and how it should work in the future.

Acknowledgements We thank Lukas Landgraf for his assistance with visualisation. We acknowledge partial support by the European Regional Development Fund (ERDF) and by the Free State of Bavaria as part of the project AIM-SMEs (Grant No. 2506-014-3.2), co-funded by the European Union; also by the High-Tech Agenda of the Free State of Bavaria, and the German Research Foundation (DFG), grant MA 9739/1-1.

[3]See CATALYST commit `a93629f`, QCOR commit `28a61e4`, and CUDA-Q commit `da88cb5`.

References

[1] M. Joblin, S. Apel, and W. Mauerer, Evolutionary trends of developer coordination: a network approach, *EMSE*, 2017.

[2] A. Ait, J. L. C. Izquierdo, and J. Cabot, An empirical study on the survival rate of GitHub projects, *MSR*, 2022.

[3] M. Joblin et al., From Developer Networks to Verified Communities: A Fine-Grained Approach, *ICSE*, 2015.

[4] R. Ramsauer, D. Lohmann, and W. Mauerer, The List is the Process: Reliable Pre-Integration Tracking of Commits on Mailing Lists, *ICSE*, 2019.

[5] H. Hemmati et al., The MSR cookbook: Mining a decade of research, *MSR*, IEEE, 2013.

[6] J. M. Murillo et al., Quantum Software Engineering: Roadmap and Challenges Ahead, *TOSEM*, 2025.

[7] K. Upadhyay et al., Analyzing the Evolution and Maintenance of Quantum Software Repositories, *QSW*, 2025.

[8] N. Ramalho et al., Mining Quantum Software Patterns in Open-Source Projects, *arXiv:2601.06281*, 2026.

[9] M. M. Yousuf and S. A. Sofi, Characterizing Bugs and Quality Attributes in Quantum Software: A Large-Scale Empirical Study, *arXiv:2512.24656*, 2025.

[10] M. De Stefano et al., Quantum software engineering issues and challenges: Insights from practitioners, *Quantum Software*. Springer, 2024.

[11] A. A. Khan et al., Mining Q&A platforms for empirical evidence on quantum software programming, *arXiv:2503.05240*, 2025.

[12] A. McCaskey et al., Extending C++ for Heterogeneous Quantum-Classical Computing, *TQC*, 2021.

[13] W. Mauerer, S. Klessinger, and S. Scherzinger, Beyond the badge: Reproducibility engineering as a lifetime skill, *SEENG*, 2022.

[14] W. Mauerer and S. Scherzinger, 1-2-3 Reproducibility for Quantum Software Experiments, *SANER*, 2022.

[15] W. Ma et al., How do developers fix cross-project correlated bugs? a case study on the GitHub scientific python ecosystem, *ICSE*, 2017.

[16] S. K. Behfar, E. Turkina, and T. Burger-Helmchen, Knowledge management in OSS communities: Relationship between dense and sparse network structures, *IJIM*, 2018.

[17] S. Dueñas et al., Perceval: Software Project Data at Your Will, *ICSE-Companion*, 2018.

[18] M. Joblin et al., Classifying Developers into Core and Peripheral: An Empirical Study on Count and Network Metrics, *ICSE*, 2017.

[19] A. J. McCaskey et al., XACC: A System-Level Software Infrastructure for Heterogeneous Quantum-Classical Computing, *arXiv:1911.02452*, 2019.

[20] N. Hoess et al., Does the Tool Matter? Exploring Some Causes of Threats to Validity in Mining Software Repositories, *SANER*, 2025.