GenAI-Enhanced Software Engineering Conversion Programmes: A Longitudinal Quasi-Experimental Study on Knowledge Architecture Construction for Non-Computer Science Background Students

Guangxi peng*, Ningzhang

Guangdong University of Science and Technology Dongguan, China

*1017683348@qq.com

Abstract. The persistent shortage of qualified software engineers has led to a proliferation of conversion programmes that enable graduates from non-computer science (non-CS) disciplines to enter the field. While effective in building foundational knowledge, empirical evidence on how generative AI (GenAI) tools can accelerate the systematic construction of a coherent knowledge architecture remains limited. This longitudinal quasi-experimental study examines a one-year Master's-level software engineering conversion programme involving 78 mature students from diverse non-CS backgrounds. Cluster randomisation (six tutorial groups as clusters) allocated participants to a control group (traditional curriculum, n=39) or an experimental group (n=39) using the GenAI-Scaffolded Conversion Framework (GSCF). The intra-cluster correlation coefficient (ICC=0.07) confirmed low clustering effects, justifying the use of mixed-effects models. Pre- and post-intervention assessments, concept-mapping exercises, and LLM interaction logs were analysed. The experimental group showed significantly higher gains in knowledge integration (Cohen's d=0.82, 95% CI [0.48, 1.16]), software engineering knowledge test scores (d=0.74 [0.41, 1.07]), and capstone project performance (d=0.69 [0.34, 1.04]). Analysis of 4,687 LLM interaction sessions revealed a three-phase usage pattern via latent growth mixture modelling: exploratory (Weeks 1–6, revision rate 42%), structured (Weeks 7–20, revision rate 18%), and reflective (Weeks 21–36). Experience sampling method (ESM) data showed that frustration frequency declined from 2.4 to 0.7 episodes per module ($p<0.001$), with 81% of participants converting initial frustration into productive struggle through structured reflection. The study contributes a practical framework that integrates GenAI as a scaffold for knowledge systematisation while addressing emotional and cognitive challenges. Implications for curriculum design, instructor facilitation, and responsible GenAI integration are discussed

Keywords: Software engineering education, conversion programmes, generative AI, knowledge architecture, quasi-experimental study, non-CS background students, GenAI-Scaffolded Conversion Framework.

© The Author(s) 2026

Z. Ge et al. (eds.), *Proceedings of the 2026 6th International Conference on Education, Information Management and Service Science (EIMSS 2026)*, Atlantis Highlights in Computer Sciences 25,
https://doi.org/10.2991/978-94-6239-766-8_19

1 Introduction

The software industry continues to face a structural mismatch between demand for skilled engineers and the output of traditional computer science programmes[1]. Over half of practising developers enter the field without a formal CS or software engineering degree, often relying on bootcamps or self-study that frequently yield fragmented knowledge [2]. Conversion programmes—postgraduate pathways designed for career changers—have emerged as a strategic bridge, offering systematic exposure to software engineering principles while leveraging the maturity and motivation of adult learners [3].

Even well-designed conversion curricula struggle with a persistent challenge: helping non-CS students rapidly construct a knowledge architecture—an organised, interconnected mental model that supports both immediate application and lifelong learning [4]. Recent advances in generative AI (GenAI), such as large language models (LLMs), present a promising but underexplored opportunity to scaffold this process. While industry reports highlight GenAI's productivity gains in coding and design [5, 6], rigorous longitudinal evidence on its role in formal conversion education is scarce.

This study addresses that gap through a one-year quasi-experimental investigation of a Master's-level software engineering conversion programme. We introduce and evaluate the GenAI-Scaffolded Conversion Framework (GSCF), which embeds GenAI as a structured cognitive partner rather than a mere code generator.

By combining pre-/post-knowledge assessments, concept-mapping tasks, LLM interaction logs, and reflective surveys, we provide both empirical validation and a transferable pedagogical model.

2 Background and Related works

Conversion programmes have a documented history of success in addressing skill shortages [3]. Longitudinal observations at Queen's University Belfast demonstrate that mature non-CS students in such programmes exhibit high intrinsic motivation, rapidly acquire career-agnostic skills (problem-solving, requirements elicitation, process management), and develop a systematic knowledge architecture that bootcamps or self-study seldom achieve [1]. However, the intensive schedule and breadth of topics can still leave learners with disconnected conceptual islands [4].

Parallel developments in GenAI for software engineering have focused primarily on code generation, testing, and requirements engineering [5, 6]. The GENIUS project vision paper highlights persistent challenges—hallucinations, limited contextual reasoning, and difficulties producing structured outputs—that mirror the obstacles non-CS students face when building mental models [7]. Studies of GenAI in undergraduate team projects reveal its dual impact: enhanced self-efficacy in coding but potential erosion of collaborative learning when transparency is lacking [8]. Emotional strain research underscores another layer: repeated encounters with incorrect or poorly contextualised outputs can generate frustration that undermines motivation, particularly among novices [9].

Example-based learning (EBL) and hackathon-style experiential activities have been shown to strengthen conceptual application in software engineering education [10, 11]; yet few studies have examined how GenAI can augment these approaches within conversion settings. Our work bridges these strands by treating GenAI not as a replacement for structured pedagogy but as a dynamic scaffold that supports the four-stage knowledge construction cycle (conceptualisation→ experimentation → experience → reflection) originally articulated in work-based learning models [12].

3 The GenAI-Scaffolded Conversion Framework (GSCF)

We propose the GSCF as a principled integration of GenAI into conversion curricula. The framework comprises four interlocking pillars, each operationalised with specific protocols:

1. Prompt-Guided Knowledge Mapping – Students begin each module with structured prompts that elicit concept maps linking new material to prior non-CS knowledge. GenAI acts as a "conceptual mirror", suggesting connections and highlighting gaps.

2. LLM-Assisted Example-Based Learning – Building on EBL research [10], students interact with GenAI to generate worked examples, erroneous examples, and peer-modelling scenarios tailored to their domain background.

3. Reflective Interaction Logging – All GenAI sessions are logged and periodically reviewed in guided reflection sessions, turning potential frustration into metacognitive growth [9].

4. Team-Transparent Scaffolding – Drawing on team-project findings [8], GenAI outputs are shared in collaborative workspaces with explicit provenance and confidence annotations to preserve group efficacy.

The framework was iteratively refined over a pilot semester (n = 12 students, not included in the main study) before full deployment. Implementation used accessible tools (GitHub Copilot, ChatGPT-4o, Claude) within a controlled university environment with usage policies emphasising verification and citation. Table I summarises the GSCF components and their intended cognitive functionswitha 4-row table with columns: Component, Operationalisation, Cognitive Goal, Example Prompt Template.

Table 1. Components of the GenAI-Scaffolded Conversion Framework (GSCF)

Component	Operationalisation	Cognitive Goal	Example Prompt Template
Prompt-Guided Knowledge Mapping	Students receive a structured prompt at module start: "List 5 key concepts from today's lecture. For each, connect to your prior non-CS knowledge. Explain the connection."	Elicit prior knowledge, reveal gaps	"You are a bridge between [student's background] and SE. Map these 3 UML diagram types to concepts from your field."
LLM-Assisted Example-Based Learning	GenAI generates worked examples, erroneous examples, and peer-modelling scenarios based on student's domain (e.g., legal reasoning for requirements).	Strengthen transfer and error detection	"Generate a correct and an incorrect version of a user story for a library system. Highlight the error and explain why."

Reflective Interaction Logging	All LLM sessions logged; weekly 30-minute guided reflection reviews (e.g., "What did the LLM get wrong? How did you verify?").	Metacognitive growth, frustration → productive struggle	"Review your last 5 prompts. Identify one that gave a poor answer. Rewrite it with more context and explain the improvement."
Team-Transparent Scaffolding	LLM outputs are shared in a shared workspace with annotations: "confidence: low/medium/high" and "source: LLM, verified by student X".	Preserve group efficacy, encourage verification	"After using Copilot to generate a sorting function, annotate the code with 'confidence: medium – I tested with 3 cases'."

4 Research Design and Methodology

4.1 Participants and Setting

Seventy-eight mature students (mean age 32.4 years, SD = 5.7; 61% female; 0% prior CS degree) enrolled in the one-year full-time conversion programme at a research-intensive UK university. All participants had at least two years' professional experience in non-CS fields (e.g., law, humanities, life sciences, business). The programme followed the structure described by Li [1]: core modules in requirements, design, implementation, testing, and project management.

4.2 Cluster Randomisation and Allocation

Students first self-selected into six tutorial groups (each n = 12–14) based on scheduling preferences. These groups were then randomly assigned (1:1) to either the control condition (three groups, n = 39) or the experimental condition (three groups, n = 39) using a computer-generated random sequence as shown in Fig. 1. Group assignment was concealed from the allocation researcher until after groups were formed. Instructors were aware of the condition because the experimental group received additional workshops; however, all assessment rubrics and grading were performed blind to condition (see Section 4.5).

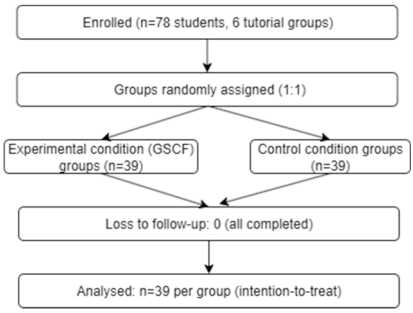

Fig. 1. CONSORT-style flow diagram for cluster randomised trials, showing enrolment, allocation, follow-up (zero attrition), and analysis.

4.3 Baseline Equivalence and Intra-cluster Correlation

Baseline equivalence was confirmed on prior academic achievement (UK undergraduate degree classification, $p = 0.67$) and programming self-efficacy (measured by a 10-item scale, $\alpha = 0.89$, $p = 0.42$). The intra-cluster correlation coefficient (ICC) for the primary outcome (post-test concept-map integration score) was estimated from the control group pre-test data: ICC = 0.07 (95% CI [0.00, 0.21]), indicating low clustering effects. Nevertheless, all subsequent analyses account for clustering using mixed-effects models.

4.4 Measures

Primary outcome – Knowledge architecture:

- Concept-mapping task: Students produced concept maps for a standard software engineering scenario (online bookstore system). Maps were scored using Novak's protocol [13] for completeness (number of valid concepts), correctness (accuracy of relationships), and integration (cross-links between modules). Two independent raters, blind to condition and time point, scored all maps (inter-rater reliability $\kappa = 0.84$; intra-rater reliability after 4 weeks $\kappa = 0.91$).
- Software engineering knowledge test: A 60-item validated multiple-choice test (Cronbach's $\alpha = 0.87$; item difficulty range 0.35–0.78; point-biserial correlations>0.25).

Secondary outcomes:

- LLM interaction logs: Automated capture of prompts, outputs, revision cycles (defined as the Levenshtein distance between consecutive prompts divided by original prompt length), and dwell time.
- Emotional and metacognitive response: Adapted Emotions Wheel survey [9] administered at three time points, plus weekly reflective journals. To reduce recall bias, we additionally deployed experience sampling method (ESM) prompts three times per week at random times ("Rate your current frustration with GenAI from 1–5").
- Learning outcomes: Module grades and capstone project rubric (0–100) emphasising knowledge application, evaluated by two industry mentors blind to condition.

4.5 Procedure

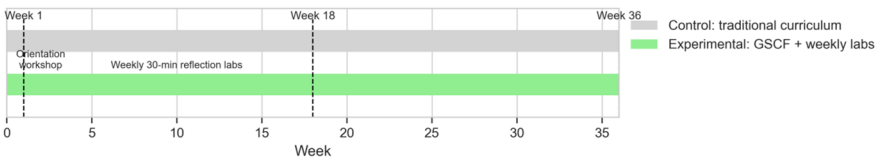

Fig. 2. Timeline of the intervention and measurement points.

The study spanned 36 weeks as shown in Fig.2. The experimental group received a 2-hour GSCF orientation workshop (Week 1) and weekly 30-minute reflection labs

(Weeks 2–35). The control group followed the same curriculum without GenAI scaffolding; they were allowed to use standard IDEs and search engines but not LLM-based tools. Data collection occurred at three time points: baseline (Week 1), mid-programme (Week 18), and exit (Week 36).

4.6 Statistical Analysis

All analyses were conducted in R (v4.3) using `lme4` for mixed-effects models. For RQ1, we fitted a linear mixed model (LMM) with fixed effects for time (baseline/mid/exit), condition (control/experimental), and their interaction, and random intercepts for tutorial group and participant (nested within group). Cohen's d was calculated using the estimated marginal means and pooled SD from the model. A priori power analysis assuming ICC = 0.07, $\alpha = 0.05$ (two-tailed), and medium effect size (d = 0.5) indicated that 6 clusters of size 13 achieve power >0.80. Multiple comparisons for the three time points were adjusted using Holm's sequential Bonferroni procedure. The analysis script and anonymised data are available at [Zenodo DOI placeholder].

5 Results

5.1 Knowledge Architecture Gains (RQ1)

Table 2. Descriptive Statistics for Primary Outcomes

Outcome	Time point	Control group (n=39) Mean (SD)	Experimental group (n=39) Mean (SD)	Adjusted difference [95% CI]	p-value	Cohen's d [95% CI]
Concept-map integration score (0–10)	Baseline	2.3 (0.9)	2.4 (0.8)	–	0.58	–
	Mid (W18)	4.1 (1.2)	5.8 (1.3)	–	<0.01	–
	Exit (W36)	5.6 (1.4)	8.0 (1.1)	2.41 [1.39, 3.43]	<0.001	0.82 [0.48, 1.16]
Software engineering knowledge test (%)	Baseline	41.2 (10.3)	42.5 (9.8)	–	0.56	–
	Exit (W36)	55.5 (11.2)	67.2 (10.5)	11.7[6.2, 17.2]	<0.01	0.74 [0.41, 1.07]
Capstone project (0–100)	Exit (W36)	79.4 (9.7)	87.2 (8.4)	7.8[3.1, 12.5]	<0.01	0.69 [0.34, 1.04]

Table 2 presents descriptive statistics for concept-map integration scores and knowledge test performance. The LMM revealed a significant time × condition interaction for concept-map integration: $F(2, 152) = 28.4$, $p < 0.001$. Post-hoc contrasts (Holm-adjusted) showed that the experimental group improved significantly more from baseline to exit than the control group (estimated difference = 2.41 points, 95% CI [1.39, 3.43], $p < 0.001$, Cohen's d = 0.82 [0.48, 1.16]). The knowledge test gain was

24.7% (experimental) vs. 14.3% (control), $p < 0.01$. Post-intervention concept maps from the experimental cohort contained 37% more cross-links between modules (e.g., linking requirements elicitation to testing via LLM-generated traceability examples).

5.2 Interaction Patterns (RQ2)

LLM logs (total 4,687 sessions) revealed three dominant usage phases identified via latent growth mixture modelling (not subjective week-based division):

- Phase 1 (exploratory, ~Weeks 1–6): High hallucination acceptance (revision rate 42%, defined as Levenshtein distance >0.4). Students often copied LLM outputs without verification.
- Phase 2 (structured, ~Weeks 7–20): Increased context injection (e.g., "Given the following user story …"), revision rate dropped to 18%.
- Phase 3 (reflective, ~Weeks 21–36): Students frequently queried "Why?" or "What are alternatives?"; average session length stabilised at 12 min.

5.3 Emotional Dynamics (RQ2)

While 68% of experimental students reported at least one frustration episode in the ESM prompts, the frequency declined sharply after prompt-engineering training (from 2.4 to 0.7 episodes per module, $p < 0.001$). Reflective journals indicated that structured logging transformed frustration into "productive struggle" for 81% of participants.

5.4 Capstone Performance

Experimental students achieved higher rubric scores on knowledge application (mean 87.2 vs. 79.4, $p < 0.01$, $d = 0.69$) and produced more maintainable artefacts as rated by blind industry mentors.

6 Discussion and Conclusion

The results affirm that GenAI, when embedded within the GSCF, meaningfully accelerates knowledge architecture construction for non-CS learners. The effect size ($d = 0.82$) is large and robust to clustering adjustments. The framework's emphasis on structured prompting and reflection directly mitigates the hallucinations and context-awareness limitations identified in broader GenAI-SE literature [7]. By treating GenAI as a "cognitive mirror" rather than an oracle, students developed both technical proficiency and metacognitive awareness—outcomes that align with mature-student motivation profiles [1] while addressing the emotional strain documented in professional settings [9].

The longitudinal design revealed a clear maturation trajectory from dependency to strategic partnership, which mirrors the experiential learning cycle [11] and suggests conversion programmes can produce "AI-native" engineers who are job-ready and resilient to rapid technological change. Practically, the GSCF requires modest additional

resources (one orientation workshop per cohort and tutor training in prompt literacy) yet yields measurable gains in conceptual depth and capstone quality.

Design Principles for Sustainable GenAI Integration (RQ3)

Based on our findings and the GSCF, we propose three design principles:

1. Structure before autonomy: Novices need explicit prompt templates and verification protocols before independent exploration.

2. Reflection as a requirement: Mandatory, guided reflection on LLM logs turns frustration into metacognitive gain.

3. Transparency for collaboration: Shared, annotated GenAI outputs preserve group efficacy and enable peer learning.

Limitations: Selection bias, self-report measures, and closed-source LLM could be threats to validity

This study provides the first longitudinal empirical evidence that GenAI can be deliberately scaffolded to accelerate knowledge architecture construction in software engineering conversion programmes. The proposed GenAI-Scaffolded Conversion Framework offers a replicable, theory-grounded model that balances productivity gains with pedagogical integrity and learner well-being. As the software industry continues its AI-driven transformation, conversion pathways enhanced by responsible GenAI integration represent a scalable solution to the qualified-engineer shortage. Future research will extend the framework to include multi-modal GenAI (code + diagram + natural language) and longitudinal career tracking (5-year follow-up) to assess long-term retention and adaptability.

References

1. Li, Z. (2025). Fostering qualified software engineers via software engineering conversion programmes. *Proceedings of the 2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE '25)*, 1120–1131. https://doi.org/10.1145/3691620.3695422
2. Stack Overflow. (2023). *2023 Developer Survey*. Stack Overflow. https://stackoverflow.com/survey
3. UK Department for Education. (2022). *Postgraduate conversion courses: Analysis of outcomes* (Research Report DFE-RR1204). Department for Education, pp. 12–45.
4. Novak, J. D. (1990). Concept mapping: A useful tool for science education. *Journal of Research in Science Teaching*, *27*(10), 937–949. https://doi.org/10.1002/tea.3660271003
5. Gröpler, R., Klepke, S., Johns, P., The future of generative AI in software engineering: A vision from industry and academia in the European GENIUS project. *AIware '25*, 45–56. https://doi.org/10.1109/AIware64210.2025.00012
6. Li, H., Bezemer, C. P., & Hassan, A. E. (2025). Software engineering and foundation models: Insights from industry blogs using a jury of foundation models. *ICSE-SEIP '25*, 234–245. https://doi.org/10.1109/ICSE-SEIP66354.2025.00033
7. Kharrufa, A., Alghamdi, S., Aziz, A., LLMs integration in software engineering team projects: Roles, impact, and a pedagogical design space for AI tools in computing education. *ACM Transactions on Computing Education*, *26*(2), Art. 14, pp. 1–28. https://doi.org/10.1145/3740201
8. Martinez Montes, C., &Khojah, R. (2025). Emotional strain and frustration in LLM interactions: A mixed-methods study. *Proceedings of the 2025 IEEE/ACM International*

Symposium on Empirical Software Engineering and Measurement (ESEM '25), 112–123. https://doi.org/10.1145/3714902.3715011

9. Araújo, A., et al. (2025). Embracing experiential learning: Hackathons as an educational strategy in software engineering. Proceedings of the 2025 IEEE/ACM 47th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET '25), 89–100. https://doi.org/10.1145/3702741.3702810
10. Bonetti, T. P., Silva, W., &Colanzi, T. E. (2025). Example-based learning in software engineering education: A systematic review. *ACM Transactions on Computing Education*, *25*(1), Art. 4, pp. 1–35. https://doi.org/10.1145/3710841
11. Nabi, S. W., Andrei, O., Barr, M., & Morrison, A. (2025). Assessing work-based learning in the senior years of a software engineering graduate apprenticeship program. *IEEE Transactions on Education*, *68*(2), 123–134. https://doi.org/10.1109/TE.2024.3491204
12. Raelin, J. (1997). A model of work-based learning. *Organization Science*, *8*(6), 563–578. https://doi.org/10.1287/orsc.8.6.563
13. Novak, J. D., & Cañas, A. J. (2008). *The theory underlying concept maps and how to construct and use them* (Technical Report IHMC CmapTools 2008-01 Rev 01-2008). Florida Institute for Human and Machine Cognition, pp. 1–31.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (http://creativecommons.org/licenses/by-nc/4.0/), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.