HUMAN-AI COLLABORATIVE INSTRUCTIONAL DESIGN FOR CLIL IN COMPUTER SCIENCE: A DUAL-RUBRIC EVALUATION OF ESP MATERIALS FOR SOFTWARE ENGINEERING UNDERGRADUATES

Ivan Shotlekov

Abstract. This paper reports a human-AI collaborative instructional design (HACID) approach inside a CLIL (Content and Language Integrated Learning) framework, where content mastery and language development are pursued simultaneously. A coursebook ecosystem was built and deployed in an ESP (English for Specific Purposes) course for first-year software engineering (SE) undergraduates, aligned with the STEAME (Science, Technology, Engineering, Art, Mathematics, and Entrepreneurship) approach. Two contrasting coursebooks anchor the analysis. Foundations of Software Engineering (FSE) is grounded in SWEBOK 4.0a and follows a knowledge-transmission model that builds foundational concepts and disciplinary register. Creativity for Software Engineers (CSE) is an experiential, project-based module in which learners design and pitch an AI-powered startup across nine phases adapted from authentic Y Combinator practitioner content encountered in the parallel Entrepreneurship for Software Engineers (ESE) course; together, ESE and CSE form CESE. The intended CLIL effect – language and content gains in synergy – is gauged through identical Pre- and Post-Course questionnaires built from higher-order Bloom’s items. Material quality was assessed by applying two complementary AI-generated rubrics to one representative unit per coursebook: a Claude rubric foregrounding ESP principles and AI integration, and a ChatGPT rubric weighted toward engagement, authenticity, and assessment potential. The two instruments produced complementary rather than redundant findings; each rubric mapped more cleanly onto the coursebook whose pedagogical logic it best reflected. The study suggests that HACID can generate pedagogically differentiated materials within a single rubric-validated ESP framework.

Key Words: *artificial intelligence, computing education, CLIL, English for Specific Purposes, instructional design, undergraduate education, software engineering, STEAME.*

Introduction

AI tools (here, GenAI is meant throughout the paper) are now deeply embedded in STEAME and in education. CLIL was dubbed "an innovative fusion" of language and subject education[Coyle], placing learners at the centre of the process and casting teachers as facilitators. ESP targets learners' "needs as language learners and their needs as language users" [1].

ESP was introduced as a standalone discipline at FMI Plovdiv in 2023/2024, although CLIL had been in place since 1995. The ELT tradition relies on a core textbook supplemented by teacher-designed materials. ESP courses, however, are more frequently personalized and updated, which calls for custom-made teaching materials.

CLIL, ESP for software engineering, and human-AI collaborative design rarely appear together in the literature, and that gap is what this paper investigates. Its research question is direct: what is the quality of human-AI collaborative instructional design for CLIL in computer science (CS)? The contribution offered here is a unified ESP framework that produces pedagogically differentiated materials, validated through a dual-rubric instrument.

Theoretical framework

A systematic review of CLIL in CS reports that "its application within the field of computer science remains underexplored", even as the integration of ICT into CLIL has been studied more thoroughly [2]. CLIL strengthens students' motivation and language skills [3]. For SE undergraduates, the "language" to be taught is broad: domain lexis, code comments, issue reports, pull-request discourse, API documentation, project proposals, sprint presentations, conference talks, workplace communication across a wide range of the application of AI in the tools used by CS learners. ESP caters not only for the workplace, but also for career development.

Another systematic review on teacher–AI co-design of learning tasks finds that the dominant model is "AI as an assistant or draft generator", while richer pedagogical potential emerges when AI acts as a "co-designer or reflective partner" – a mode that, alongside AI as feedback agent, "remain[s] underexplored in instructional practice" [4]. The systematic combination of GenAI and learning analytics is reported to "create an effective environment for developing and refining educational content" [5].

Another systematic review concludes that “educators are gradually adopting LLMs in their courses but that most CS curricula still need to be changed to accommodate recent advances in AI” [6]. A review of AI and ML in programming education shows that AI tools now operate across course design, classroom implementation, assessment, feedback, and performance monitoring [7]. Another author observes that “Generative AI … has the potential to transform many aspects of how computer science is taught” [8].

Used responsibly, AI works as a teaching assistant that augments the instructor without replacing them. It can act at any phase of design – needs analysis, content–language mapping, activity design, scaffolding, assessment, feedback – more efficiently than manual design alone, and it can bridge expertise gaps to help educators address content or pedagogic shortfalls. Iteration is cheap, so refinement becomes more responsive. The approach also exercises 21st-century competencies, since learners and educators must think critically about AI-generated outputs in modern tech environments. The human-AI partnership is synergetic as humans tend to better understand learner needs, apply learning science, create emotionally engaging experiences, while AI excels at content generation and curation, adaptive learning paths, automated assessments & feedback. [9]

The coursebook ecosystem

The HACID effort drew on authentic sources to build an ecosystem of free-access, self-study-friendly coursebooks. The materials are designed around real-life competencies, often pushing students out of their comfort zone. The coursebooks are the carriers for courses and course modules and address different needs:

A. General-purpose: 1. TotB Thinking outside the Box – cross-programme problem-solving activities; students substantiate and illustrate their answers.

B. Computer Science programmes: 2. FSE Foundations of Software Engineering (SWEBOK-based); **3. ESE** Entrepreneurship for Software Engineers (YC-based); **4. CSE** Creativity for Software Engineers (experiential, learn-by-doing build of a startup app one step at a time); **5. Technical Writing for Software Engineers** (SRS-based); **6. LSSE** Life Skills for Software Engineers (1st semester); **7. EAPSE** English for Academic Purposes for Software Engineers (1st semester); **8. SSS** Soft Skills for Software Engineers (20 soft skills as a common thread, 1st semester).

C. Mathematics programmes: 9. M4CS Mathematics for Computer Science; **10. MathoScope** Selected chapters in Mathematics (video-based); **11. Writing for Math:** Scientific Journal Article.

D. Education programmes: 12. Methodology of Teaching CS & IT in High School (advanced pedagogical approaches and technology integration); **13. Writing for Education:** Scientific Journal Article (elective).

The empirical work concentrated on two of these: FSE and CSE, the latter, together with ESE, form CESE (Creativity and Entrepreneurship for Software Engineers). Their pedagogical logics, summarised in Table 1, are deliberately contrastive within the same framework.

Table 1. FSE and CESE course modules

FSE (transmission model)	**CESE** (ESE & CSE)
Before class: Reader 6.1, Reader 6.2. **During class:** vocabulary, reader, comprehension, AI-integrated activity with sample answer. **Next class:** retrieval via peer-briefing infographics. **Resources:** two audio readers and a glossary of reader terminology.	**ESE** (transmission model) **A.** Brainstorming in pairs; **B.** vocabulary; **C.** problem solving; **D.** short video; **E.** comprehension; **F.** grammar focus; **G.** POV discussion in small groups.
FSE activator (interactive individual, peer, and small group CLIL activities)	**CSE** (YC-inspired experiential model) "StudyBuddy Startup" storyboard.
A. Term classifier; **B.** defining register: choose the exact word; **C.** register transformation: formal technical writing; **D.** concept cards (term → definition); **E.** cloze synthesis passage; **F.** spoken interaction on a lecture-related topic; **G.** discussion prompts. Teacher Materials Pack included.	**A.** Lesson overview; **B.** microlearning content background; **C.** key vocabulary; **D.** activities; **E.** deliverable template; **F.** language focus, patterns in context; **G.** self-reflection; **H.** self-evaluation rubric; **I.** extension resources. Homework: activity plus sample deliverable. Teacher Materials Pack with autograding rubric and prompt.
FSE Capstone Project:	**CESE Captone Project:**
Hackathon – AI app	Mock investor meeting; a startup pitch
Methodology highlights:	**Methodology highlights:**
Task-Based Language Teaching with authentic materials; technology-enhanced and blended learning; Project-Based Learning; Design Thinking; Inquiry-Based Learning; 5E/6E STEAME models (Engage, Explore, Explain, Elaborate, Evaluate).	Business-literacy substrate and authentic Y Combinator practitioner materials (ESE); experiential, project-based model with a design process and creativity skills as cognitive tools (CSE); language performance and professional artefact in one (CSE); language development through doing, not studying (CSE).

In both modules, the activities span the Bloom's taxonomy range from Apply to Evaluate.

Methodology: dual-rubric evaluation design

Two instruments addressed the research question. The first was a Pre/Post questionnaire developed with Claude, made up of 20 scenario-based items pitched at the higher-order levels of Bloom's taxonomy and designed to be administered identically before and after the course. A representative stem reads: "A software engineer discovers that their employer's product secretly collects user health data without consent and shares it with advertisers. According to the ACM/IEEE Code of Ethics, what should guide the engineer's decision?" – with five options culminating in an explicit "I don't know" to discourage guessing. Post-course results were not yet available at the time of writing.

The second instrument was a dual-rubric gauge: a Claude-developed rubric foregrounding ESP and AI integration, and a ChatGPT-generated rubric weighted toward engagement, authenticity, and assessment. One representative unit per coursebook was assessed, since all units follow the same pattern. A comparative dual-rubric design was preferred to a single consensus rubric for breadth, and the second rubric was generated by ChatGPT as an external evaluator because the materials had been produced by Claude (with the audio readers as exceptions).

Claude's rubric uses 16 criteria organised across six dimensions: ESP Principles & Disciplinary Language (technical vocabulary integration, genre & text-type awareness, needs-analysis alignment); AI Tool Integration & Pedagogical Design (purposeful AI use, scaffolding & instructional sequence, cognitive load management); Language Skills Development (technical reading, technical/professional writing, professional speaking & listening); Critical AI Literacy & Learner Autonomy (evaluating AI output, learner autonomy & metacognition); Assessment, Feedback & Validity (alignment of objectives–tasks–assessment, quality and timeliness of feedback, academic-integrity safeguards); and Accessibility, Inclusivity & Ethics (multimodal accessibility, cultural & linguistic inclusivity). Each criterion is scored 1–4 against a detailed descriptor. Final scores fall into four colour-coded bands – Exemplary (86–100%), Proficient (66–84%), Developing (42–64%), Beginning (25–41%) – and are accompanied by qualitative notes on Strengths, Areas for improvement, and Specific Recommendations for AI Tool Adjustment, plus an Overall Recommendation: adopt; adopt with revisions; redesign.

ChatGPT's rubric uses 10 weighted criteria: alignment with ESP and course outcomes; linguistic quality and appropriateness; authenticity and disciplinary relevance; pedagogical design and instructional value; quality of AI integration; accuracy, reliability, and transparency of AI-generated content; assessment and feedback potential; student engagement and interaction; accessibility and inclusiveness; and ethical and responsible use of AI. Its bands are stricter: Excellent / highly suitable for implementation (85–100); Good / suitable with minor revision (70–84); Adequate / requires targeted improvement (55–69); Unsatisfactory / major redesign needed (below 55). It also produces qualitative notes on Strengths and Areas for improvement.

AI assistance (Claude) was also used for the manuscript editing.

Findings

Two AI reviewers, Claude and ChatGPT, applied each of the two rubrics.

Table 2. Evaluation by the Claude rubric v3, 16 criteria

	FSE & FSE Activator		CSE	
Criteria / Reviewer	Claude	ChatGPT	Claude	ChatGPT
Overall score	37/64	42/64	53/64	52/64

The review reports are detailed and offer concrete pointers for the next iteration alongside named strengths.

FSE strengths (Claude evaluator): exemplary CLIL integration, with technical content verifiably grounded in SWEBOK v4 (Software Architecture KA, Software Engineering Models and Methods KA) interwoven with language pillars (defining register, function/purpose statements, passive-authority constructions, technical discourse) across seven well-paced sections; mature, discipline-appropriate visual design; well-managed cognitive load.

FSE areas for improvement (Claude evaluator): AI tool integration is thin; authentic SE writing genres are absent (no READMEs, user stories, ADRs, commit messages, or incident reports), and writing work stays at sentence-level transformation; no AI speech tools or pronunciation feedback; the drag-and-drop interactions in Sections A and E lack keyboard alternatives and ARIA support (WCAG 2.1 fail), and colour is the sole differentiator for some feedback states; metacognitive prompts are minimal.

Table 3. Evaluation by the ChatGPT rubric v2

	FSE & FSE Activator		**CSE**	
Criteria	**Claude**	**ChatGPT**	**Claude**	**ChatGPT**
Overall score	77.50/100	77.50/100	95.00/100	95.00/100

Some scores might have been higher had the evaluators received more context up front. AI-use essentials such as critical reading of outputs and hallucination risks were taught in the first-semester EAP course and were not repeated in writing in the unit, only orally as needed.

Discussion

Two observations stand out: (1) the ChatGPT rubric tends to award higher scores than Claude; (2) the two reviewers’ scores were very close on the Claude rubric and identical on the ChatGPT rubric. The wider 9% gap on FSE & FSE Activator concentrates in four of five criteria within just two dimensions: AI Tool Integration & Pedagogical Design, and Critical AI Literacy & Learner Autonomy.

The two rubrics agree closely on the revisions needed: more learner-facing AI; SE-genre writing artefacts; richer feedback; WCAG 2.1 keyboard and ARIA compliance; culturally diversified framing; an explicit AI-use and integrity policy (even though this had been taught the previous semester); a metacognitive close; learner-visible objectives; and a listening activity. Together with other concrete recommendations, these findings supplied a clear agenda for the next edition.

For a first version of these materials, the reviews are encouraging, especially given that LLMs continue to improve and so widen the scope for higher-quality human-AI collaborative design. The lower FSE scores trace partly to legacy course design; the stronger CSE outcome owes much to working with the Claude Opus models, where the collaboration has been notably productive. In future editions, the FSE pedagogical design can be enriched with goals that are more visible to students, self-reflection, etc.

Conclusion

A high-quality outcome from human-AI collaborative instructional design depends on teachers who command both the subject matter and the pedagogy: they must steer the prompt, iterate on the output, and verify what the model produces. As the aphorism has it, “If you don’t know where you’re going, any road will take you there.” The argument here is therefore not “use AI to make English CS lessons more productive”, but rather: use AI inside an educator-controlled, evidence-informed, ESP-grounded design cycle that knits together computer-science concepts, English-for-SE genres, cognitive scaffolding, and AI literacy. Student feedback throughout the course, particularly the open responses in the Course Evaluation Questionnaire, remains a core part of that cycle.

Early in the experiment, several technical hiccups consumed excessive time, effort, tokens, and the course designer’s energy. As LLMs mature, that overhead shrinks and more of the work can go into the quality of the materials and the enrichment of the learning experience.

References

[1] T. Hutchinson, A. Waters, *English for Specific Purposes: A Learning-Centered Approach*, Cambridge University Press, 2010

[2] G. Kussaiynkyzy, I. Dringó-Horváth, ICT-Supported CLIL in Computer Science: Enhancing Learning Through Digital Tools and Resources, *J. ICT Educ.*, 2025, no. 12 (2), 1–21, DOI: https://doi.org/10.37134/jictie.vol12.2.1.2025

[3] D. Lasagabaster, Y. Ruiz de Zarobe, CLIL in Spain: Implementation, Results, and Teacher Training, in: Y. Ruiz de Zarobe, J. M. Sierra, F. Gallardo del Puerto (Eds.), *Content and Language Integrated Learning: Evidence from Research in Europe*, Multilingual Matters, 2010

[4] Z. Wang, L. Mingzhuo, A. Islam, Reimagining Teacher-AI Co-Design in Learning Task Design: Trends and Perspectives, *Humanit. Soc. Sci. Commun.*

[5] D. Mah, M. Egloffstein, A Co-Creative Approach for AI-Enhanced Instructional Design Combining Generative Artificial Intelligence and Learning Analytics, *J. Appl. Instr. Des.*, https://madoc.bib.uni-mannheim.de/70703/1/22719.pdf

[6] N. Raihan, M. Siddiq, J. Santos, M. Zampieri, Large Language Models in Computer Science Education: A Systematic Literature Review, Oct. 2024, DOI: https://doi.org/10.48550/arXiv.2410.16349

[7] P. Manorat, S. Tuarob, S. Pongpaichet, Artificial Intelligence in Computer Programming Education: A Systematic Literature Review, *Comput. Educ. Artif. Intell.*, 2025, vol. 8, no. 100403, DOI: https://doi.org/10.1016/j.caeai.2025.100403

[8] D. Franklin, P. Denny, D. Gonzalez-Maldonado, M. Tran, *Generative AI in Computer Science Education: Challenges and Opportunities*, 2025, DOI: https://doi.org/10.1017/9781009581738

[9] L. Cobing, AI vs. Human Instructional Designers: Can They Coexist? What Does the Future of Instructional Design Look Like?, www.mindsmith.ai

Ivan Shotlekov
Paisii Hilendarski University of Plovdiv,
Faculty of Mathematics and Informatics,
236 Bulgaria Blvd., 4027 Plovdiv, Bulgaria
Corresponding author: shotlekov@uni-plovdiv.bg