Reflections on an ERP Curriculum: Insights from Three Enterprise Systems Courses

Simon Hacks[1,†], Martin Henkel[1,*,†]

[1]Department of Computer and Systems Sciences, Stockholm University, Stockholm, Sweden

Abstract

Enterprise Resource Planning (ERP) systems integrate organizational processes and shared data structures, making them an essential topic in information systems education. Yet, teaching ERP differs from classical computer science education: student learning is often driven less by programming and more by system configuration, master data design, and reasoning about end-to-end business processes in a large standard system. This paper presents our reflections on three ERP courses at Stockholm University. We describe the course progression from the first course on introductory ERP use, to technical perspectives, to the third course on project-based ERP system tailoring. We relate our course design to established ERP education research and report observations from multiple course iterations, including the benefits and limitations of recurring threshold ERP concepts that impede student progress, the flipped classroom delivery model, challenges in teaching ERP-related tasks that rely on prior knowledge, and emerging issues in examinations.

Keywords

ERP education, enterprise systems, standard systems, flipped classroom, threshold concepts, curriculum design

1. Introduction

Enterprise Resource Planning (ERP) systems constitute a central class of standardized enterprise systems that support integrated, cross-functional business processes and shared data structures across an organization. Classic research highlights that such systems promise enterprise-wide integration of information flows and standardized process execution, thereby shaping how organizations operate [1, 2]. For information systems and related programs, this creates a clear educational necessity: graduates must be able to reason about enterprise integration and process orientation, and they need practical competence in order to connect conceptual knowledge to operational reality [3, 4].

Teaching ERP systems, however, differs in important ways from classical computer science (CS) education. In many CS courses, students primarily learn by constructing software artifacts through programming and algorithmic problem solving. By contrast, ERP systems are ready-made software that need to be configured to fit an enterprise. Thereby, learning ERP systems typically emphasizes understanding and instantiating enterprise concepts through configuration, master data design, and process execution within large standard systems. This shifts the pedagogical focus from implementing functionality in code to interpreting understanding end-to-end processes, and tracing the effects of configuration and data [5, 6]. ERP education research, therefore, often recommends experiential and practice-oriented learning designs to help students develop an integrated view of business operations and enterprise integration [5, 6].

This article reflects on the development and current practice of ERP teaching at Stockholm University (SU) through a three-course sequence: ERP1, ERP2, and PROAFF. The initial versions of the courses were shaped by established enterprise-systems teaching material [7]. Over time, the courses were iteratively adapted and extended. A major redesign occurred during the COVID-19 period, when the courses transitioned to a flipped classroom model. Most recently, the widespread availability of generative AI

Baltic DB&IS 2026 Conference Forum and Doctoral Consortium, 28 June - 1 July 2026, Tartu, Estonia

*Corresponding author.
†These authors contributed equally.
simon.hacks@dsv.su.se (S. Hacks); martinh@dsv.su.se (M. Henkel)
0000-0003-0478-9347 (S. Hacks); 0000-0003-3290-2597 (M. Henkel)

tools has introduced new challenges for ERP education, particularly regarding student preparation practices, authenticity of written artifacts, and the validity of certain assessment formats [8, 9].

The goal of this work is to synthesize our experiences into a structured account of how to design, deliver, and evolve ERP teaching under changing constraints. We contribute (i) an overview of the three-course sequence and its pedagogical rationale, (ii) observations about recurring learning barriers and effective teaching interventions, and (iii) practical recommendations that other lecturers can adapt when developing or revising ERP courses in higher education.

The remainder of the article is structured as follows. Section 2 discusses key challenges of teaching ERP systems and how they differ from CS-oriented teaching contexts. Section 3 describes the three-course sequence at SUs. Section 4 presents our observations from course delivery, and Section 5 presents the students' perspective. Finally, Section 6 summarizes lessons learned and outlines directions for future development of ERP education.

2. Challenges of teaching ERP systems

ERP education has matured into a distinct stream within the field of information systems education research. A recurring motivation is that ERP platforms embody enterprise-wide integration and process orientation, which are difficult to convey through isolated, function-specific IT exercises. Early contributions, therefore, focused on integrating ERP topics into university curricula. For example, Watson and Schneider [3] argue that ERP concepts can be taught without direct access to a live system, but that hands-on exposure substantially strengthens experiential learning. Hawking [10] describes practical barriers to the introduction of ERP into curricula and proposes an e-learning delivery model that combines synchronous and asynchronous elements.

Summarizing existing literature on ERP teaching in higher education, the following challenges are often highlighted:

Pedagogical challenges

1. **Teaching integration rather than transactions.** Students can learn keystrokes, but developing understanding of cross-functional integration and end-to-end process logic is harder; typical exercises risk remaining modular and fragmented [11, 6].
2. **High cognitive load and "overwhelming systems".** ERP systems expose learners to dense interfaces, extensive terminology, and many configuration options. Without careful scaffolding, students may focus on procedural survival rather than conceptual learning [5].
3. **Balancing conceptual depth with hands-on realism.** Courses must continuously negotiate the trade-off between conceptual ERP principles (process orientation, data integrity, integration mechanisms) and practical competence (executing and explaining tasks in the system) [5, 3].

Infrastructure challenges

4. **Infrastructure and platform constraints.** Access to stable ERP environments, suitable datasets, user management, and resettable sandboxes is non-trivial and can shape what kinds of pedagogy are feasible [10].
5. **Keeping content current.** ERP ecosystems evolve (cloud delivery, APIs, analytics, vendor updates). Courses must be continuously revised to remain aligned with contemporary ERP practice while preserving conceptual continuity [4].
6. **Instructor expertise and preparation effort.** ERP teaching requires substantial instructor and support-staff capability. Barriers include limited prior faculty experience, the time required for curriculum development, and the complexity of maintaining the environment [10].

Course design challenges

7. **Course design for heterogeneous cohorts.** ERP courses often attract students from different majors and with different expectations (business vs. technical orientation). Designing learning progressions that serve this diversity is a persistent issue [4, 5].
8. **Assessment of applied configuration work.** Evaluating ERP competence is not equivalent to

grading programming assignments; assessment must capture process understanding, correctness of configuration, and the ability to demonstrate and explain outcomes in a running system [5].

9. **Managing teamwork and authentic project work.** Many ERP teaching designs rely on group projects to mirror enterprise practice, but group dynamics, role division, and fairness in assessment require explicit management and transparent evaluation criteria [12].

Overall, published research suggests that ERP education is most effective when it combines structured conceptual framing with repeated, scaffolded practice in authentic or simulated ERP environments [5, 6]. Simulation games and flipped designs are frequently reported as effective because they make integration visible [13, 11].

3. Teaching ERP systems at Stockholm University

At SU, ERP systems are taught through a coherent sequence of three courses: ERP1, ERP2, and PROAFF. Together, these courses form the core of a bachelor-level program in ERP systems, offering a structured progression from introductory system use to technical understanding and, ultimately, project-based solution implementation. While they are designed as a connected learning pathway within the program, each course, except PROAFF, can also be taken as a standalone elective, resulting in a mix of student backgrounds and a broader range of expectations regarding prior knowledge.

3.1. Overarching concepts

The three courses (ERP1, ERP2, and PROAFF) are designed as a coherent sequence, but they share a set of overarching concepts that unify them. A central design choice across the courses is the use of a flipped classroom format [14]. Core content [7] is delivered through pre-recorded lectures that students are expected to watch before scheduled sessions. The allocated lecture time is then used for active work on tasks and peer discussion. This structure shifts the emphasis from passive reception to application and problem-solving.

In addition to lectures, the courses include a substantial lab component conducted in Microsoft Dynamics 365 Business Central[1], a mid-sized ERP system with about 400 screens. The lab provides continuity and a shared experiential basis across the cohort: students work with the same platform and must complete tasks that reflect business-relevant operations of varying difficulty. Typical tasks include executing economic or logistical activities in Business Central (e.g., purchasing, inventory movements, or related postings). The labs are conducted in groups of four to five students, reflecting the collaborative nature of enterprise systems work in organizations, where tasks often span functional roles and require coordination. During the project period, regular supervision meetings are organized. These meetings serve as structured support points where groups can raise technical issues, discuss process interpretations, and obtain guidance when they encounter dead ends.

The assessment aligns with the dual focus on conceptual understanding and applied competence. For ERP1 and ERP2, the lecture content is assessed through a written exam that combines multiple question formats: (i) short answer essay questions, (ii) multiple choice questions, and (iii) two longer essay questions. This mixed format allows the assessment to cover both breadth and depth. For PROAFF, examinations are conducted through an individual literature assignment and a project report.

The lab component is assessed through a practical demonstration organized as a paired-group presentation. Two groups are assessed in the same session. Each group presents its implemented solution in the system and is asked to perform selected tasks live (e.g., executing a delivery in their configured environment). This assessment format is designed to evaluate not only whether the group has completed the project deliverables, but also whether they can operate the system, explain what they have configured, and respond to situated requests that resemble real operational needs.

[1] https://www.microsoft.com/en-us/dynamics-365/products/business-central

3.2. ERP1 - Introduction

ERP1 is structured as an introductory course that combines lectures with several tightly coupled learning activities. The lecture series provides the conceptual framework of enterprise systems across core business domains (e.g., supply chain, finance, CRM) and introduces managerial and societal aspects (e.g., change management and ethics). The accompanying activities are designed to enable students to operationalize these ideas in an ERP environment.

In addition to the lecture sessions and their exercises, ERP1 includes four specific components: (1) an individual, mandatory knowledge test on basic concepts delivered via the learning platform; (2) a lab in which students use a pre-configured ERP environment to execute an end-to-end operational flow; (3) a lab where students create a new company and perform initial setup activities; and (4) a seminar on current research in ERP/enterprise systems.

3.3. ERP2 - Technical insights

ERP2 builds on the operational familiarity developed in ERP1 and shifts the emphasis towards the technical and design-oriented aspects of enterprise systems. The course frames design primarily as configuring and adapting an ERP solution to organizational needs, while architecture is introduced as the structure "under the surface", including software architecture layers, integration mechanisms, data synchronization, and APIs. The course is organized around nine lectures.

While the overarching setup for the course sequence emphasizes that scheduled teaching time is devoted to student work and discussion, ERP2 operationalizes this principle through three structured theme assignments tied to specific lecture topics. These are smaller, hands-on tasks intended to connect lecture concepts to concrete modeling or technical artifacts, and they are carried out partly during the scheduled sessions, where students can obtain immediate support. The assignments cover (A) information modeling (using UML class diagrams), (B) database use (via SQL), and (C) API use (via pseudo-code).

As in the overarching course design, ERP2 includes a group project carried out in Microsoft Dynamics 365 Business Central. However, ERP2 explicitly positions the project as a more comprehensive and in-depth task than in ERP1. The project requires groups to establish a complete company setup and configure processes that span multiple functional concerns. Similar to ERP1, ERP2 separates assessment into (i) a theory component and (ii) an applied project component. The theory part consists of a written examination, while the applied part consists of the Business Central project.

3.4. PROAFF - ERP projects

PROAFF is organized as a project-based course, where project work serves as the primary vehicle for learning and assessment. In contrast to ERP1 and ERP2, which combine lecture sequences with a substantial project component, PROAFF is structured around a group project, complemented by an individual literature task that directly supports it.

The course consists of two components: a group-based project and an individual assignment that involves literature. The project involves several activities, including selecting a solution approach, development, implementation, and evaluation. Importantly, the project is expected to be executed according to project definitions and plans developed within the project work itself, which explicitly emphasizes planning and structured execution as learning objectives. Instructional support is provided through a combination of lectures, supervision sessions, and seminars. Additionally, the course includes recurring status meetings. A final seminar is included, with mandatory attendance.

The course is assessed through a project-based approach. The expected learning outcomes emphasize (i) analyzing, comparing, and proposing solutions based on available analysis and design methods, (ii) applying established methods and techniques in a structured manner, and (iii) establishing and maintaining an effective customer relationship through communication during the project.

4. Observations

4.1. Threshold concepts

Threshold concept theory offers a lens for understanding why students can progress smoothly in certain aspects of a course while becoming "stuck" at specific conceptual barriers. Threshold concepts are typically described as conceptual "portals" that, once understood, transform how learners interpret the subject; they can be troublesome (counter-intuitive or alien), and associated with students oscillating between old and emerging understandings [15, 16]. From a curriculum perspective, this lens is especially valuable for complex domains where novices can complete tasks through procedural imitation without developing a robust conceptual model. ERP education is a typical case: students can execute transactions by following instructions, but still lack an integrated understanding of cross-functional processes, and the relationship between configuration choices and organizational outcomes. Identifying threshold concepts, therefore, helps to explain recurring error patterns, design teaching interventions that support students, and focus instructional time on concepts that enable further learning [16]. The threshold concept has been criticized for lacking preciseness and empirical validation [17]. However, here we use it to present what in our experience the student get "stuck on" in a general sense, our use of the concept is thus not tied to a specific measurement of difficulty levels.

To make these threshold concepts actionable for course development, we describe each concept using a PICO-inspired design pattern format. Here, PICO serves as a compact structure to specify the recurring learning barrier as a pedagogical (P)roblem in our context, describe the teaching (I)ntervention we use to help students cross the conceptual threshold, state the baseline or implicit (C)omparison (often a transaction-focused, step-following mode of work), and define intended (O)utcomes that indicate conceptual progress. This representation supports cumulative refinement of teaching practices and provides a common schema for discussing threshold-related interventions across courses:

Stovepipe roles. (P:) Students often approach ERP tasks with a siloed mental model. They treat role-specific work (e.g., purchasing, inventory, accounting) as independent activities and focus on local task completion, even when end-to-end outcomes are incorrect. This is troublesome because ERP systems operationalize enterprise integration; actions in one functional area propagate into other roles and into financial and operational reporting. The problem is exacerbated by the fact that many ERP systems have a role-based interface: switching to another user role provides a limited, stovepipe view of the system. (I:) To address this, we design end-to-end process tasks that require students to execute a chain of activities spanning multiple roles. (C:) The implicit baseline is transaction-focused work within a single functional "stovepipe" where success is defined as completing a click path rather than reasoning about downstream consequences. (O:) Students are considered to have crossed this threshold when they demonstrate a process-oriented approach and can predict and explain the downstream effects across roles. This aligns with ERP education findings that simulation-based and process-oriented teaching can strengthen integrative understanding by making cross-functional dependencies visible and consequential [6].

Hidden processes. (P:) A recurring difficulty is that students observe a screen-level action (e.g., posting an invoice) but do not comprehend the underlying background processes executed by the ERP (posting logic, document flow, reservations, costing, and ledger entries). They can reproduce procedures but cannot justify outcomes or interpret system feedback. The student has a problem building a mental model of how the activities form a coherent whole - a process. Similar to stovepipes, ERP systems often do not provide a graphical overview of the process state, making it difficult to determine the next step. This resembles the threshold concept notion of ’alien’ or ’ritual’ knowledge, where the underlying game is initially imperceptible [15, 16]. (I:) We therefore make traceability a routine activity: students inspect resulting entries, follow document chains, and compare expected versus actual consequences. At the start of the first course (ERP1) we also provide a process model that depict the activities to be carried out in the ERP system, thereby providing a "map" for the student to use. In the following course (ERP2), we demand that the student reflect on the overall task and draw their own map - a process model (C:) The baseline is click-path learning, where learners are not required to predict outcomes and do not systematically inspect traces beyond confirmation messages. (O:) Crossing this threshold is visible when students can explain the causal mechanism behind outcomes using system traces and can troubleshoot by identifying which hidden process failed (e.g., missing setup vs. missing master data), rather than treating errors as opaque. This is consistent with evidence that hands-on interaction combined with clear instructional structure improves students' perceived learning and satisfaction in ERP courses [18].

Reality vs system. (P:) Students may conflate business reality with its representation in the ERP system, or they may treat master data and configuration as administrative overhead rather than as representational choices that shape organizational action. Typical failure modes include unrealistic assumptions about what the system "should know" and difficulties explaining why intended operations do not work in the configured environment. For example, a common concept is to perform a "put-away" of an item. In reality, this means physically placing an item on its designated shelf in a warehouse, making it available for picking. However, in reality, not doing this may surprise the student when an item is not available for picking until they register a Put-away. (I:) We address this by requiring students to justify how master data structures and configuration choices correspond to policies, roles, and process designs in the imaginary company case. We use scenarios in which real-world practices are ambiguous, ask students to decide how to represent them, and then analyze the consequences of reporting and execution. An concrete example is from the ERP2 course, where pricing on products is unclear, the student needs to interpret the pricing structure and implement it in the system. (C:) The implicit comparison is either a naive realism stance ("the ERP mirrors reality") or a purely technical stance ("the ERP is only software"). (O:) Students have crossed this threshold when they can articulate how technical choices constrain or enable work practices. This aligns with IS education research that frames a key threshold concept as recognizing information systems as social systems rather than purely technical artifacts [19].

Abstractions & instantiations of abstractions. (P:) ERP learning requires students to operate across multiple abstraction layers. ERP systems are standard systems, designed to be usable by a variety of businesses. Thus, they inherently need abstractions to be configured to fit a business. This includes, for example, being able to set different VAT percentages based on item type, customer categories, and so on. To configure the VAT, there is a need to work with five different screens in the mid-size ERP system used on the courses. Doing this entails mowing from quite concrete concepts, such as Customer and Product, to abstractions, such as product template and the relation between product template and customer templates. This, combined with the fact that, for the student, new basic concepts such as "posting" or "ledger" are complicated. Students often struggle to coordinate these layers: they either follow configuration recipes without understanding what is being instantiated or understand the concept but cannot realize it in the system. This difficulty is compounded by high element interactivity, where many interacting concepts must be coordinated simultaneously [20, 21]. (I:) To address this, we lay the groundwork for understanding basic concepts early, then target more abstract concepts later. Concretely, basic concepts are taught early i ERP1 by documenting the core ERP concepts, the student are then taking a quiz confirming their understanding of the concepts. Later, special assignments are designed so that the student delves into the abstract design of an ERP system, thereby discovering the higher-level abstractions it represents. This is done by letting the students document abstractions via conceptual modeling. (C:) The baseline for comparison is either a "recipe configuration" without conceptual framing or a purely conceptual discussion without instantiation, both of which inhibit integration. (O:) Crossing the threshold is evident when students can explain both the abstraction and its instantiation, and can reason about the consequences of alternative instantiations.

Overwhelming menus. (P:) ERP user interfaces can impose substantial extraneous cognitive load on novices. Deep menus, dense navigation structures, and limited guidance can divert attention from conceptual learning goals to the task of "finding the right place" within the system. ERP UI research notes that navigation guidance and related difficulties in mastering interface complexity are missing [22]. Cognitive load theory similarly stresses that reducing unnecessary load is essential to free working memory for schema acquisition [23, 20, 21]. (I:) We scaffold navigation through curated task paths, the progressive release of interface scope, and role-relevant views. Scaffolding is gradually removed as students build mental models of system structure and become more autonomous. For example, the first lab in ERP1 is similar to a guided track that the students follow, while the second lab is more open. For ERP2, and especially PROAFF the students are expected to develop their own "map" of the system. (C:) The baseline is unscaffolded exploration of the full interface from the start, which often leads students to compensate through imitation or superficial completion strategies. (O:) Students are considered to have crossed this threshold when they can locate relevant functionality, explain why it is relevant to a process goal, and transfer this competence to unfamiliar tasks without excessive trial and error.

4.2. Student's use of AI

Generative AI (genAI) tools are readily available to students and increasingly used for writing, summarization, and drafting academic content [24, 25]. In our courses, genAI use emerged most clearly in two situations: (i) seminar presentations where students were expected to identify and present a scientific article, and (ii) post-exam grade discussions where students argued for additional points on the exam.

In the ERP1 seminar, students were required to independently find a scientific article, discuss candidate articles in groups, select one, and prepare a brief presentation of its contents. During the seminar, we observed that many groups presented slides whose wording strongly indicated LLM-generated text, for example having advertisement-like language. In several cases, presenters read the slide text or LLM prepared speeches verbatim, and follow-up questions revealed limited understanding of both the source article and the content on their own slides. From a learning perspective, this pattern is problematic because the seminar is intended to develop academic reading skills (identifying key arguments, methods, and results) and to strengthen students' ability to explain and defend their interpretation.

This observation is consistent with concerns in the literature that genAI can encourage superficial learning habits when students use it to replace core cognitive work (reading, interpreting, synthesizing) with ready-made text [25]. It also aligns with empirical findings that students may adopt genAI for disburdenment, i.e., to offload the burdens of learning, and may exhibit forms of automation bias (over-trusting or over-relying on AI-generated output despite known limitations) [26].

After exam grading was announced, a subset of students submitted written arguments for re-evaluation of the exam that appeared to have been produced by a genAI. These texts were often generic, rhetorically well-formed, and framed in academic language, but they were frequently shallow and sometimes did not relate to the answers the students had actually provided in the exam. As a result, the submitted arguments did not provide the necessary evidence for a meaningful reevaluation. This behavior indicates that students may use genAI as a low-effort mechanism to generate plausible-sounding claims, rather than as a tool to support careful self-assessment and evidence-based reasoning.

Prior research helps interpret this pattern. First, systematic reviews and empirical syntheses show that ChatGPT interventions can reduce learners' mental effort, which may be beneficial when used to support learning but can also facilitate shortcuts when used to bypass demanding tasks [9]. Further, the observed "first-shot prompting" behavior is consistent with the notion of an AI literacy gap: students may lack strategies for prompting, verification, and reflective use, and therefore default to producing a single output that is submitted with minimal iteration or quality control [27].

4.3. Flipped classroom

Across ERP1 and ERP2, we implemented a flipped classroom approach. This design aligns with broader higher-education evidence indicating that flipped formats can be associated with increased student satisfaction and performance, provided that students engage with preparatory material and that in-class time is used for active learning rather than re-lecturing [28]. In ERP education specifically, Bhuiyan et al. [13] report that students in flipped ERP courses benefit particularly when in-class activities include structured concept checks and analytics, combined with simulation-based exercises. This supports the core rationale of our flipped setup: the primary value of contact time lies in sense-making, feedback, and collaborative problem-solving around complex and integrated ERP concepts.

Additionally, we offered students the option to participate in exercises online. During discussion phases, online participants were assigned to breakout rooms. However, we observed almost no interaction among online participants. Most online students remained passive and consumed the session as a live stream, with limited visible engagement in discussion tasks. This observation is consistent with evidence from synchronous online teaching research, which shows that a substantial share of students prefer passive participation and that lack of confidence and stress associated with speaking up can suppress active engagement [29].

A second observation is that attendance in scheduled lecture sessions is comparatively low. Furthermore, among those present, some students remain largely passive, reducing the effectiveness of the discussion-based segments. This is a critical limitation because flipped teaching relies on active learning phases to realize its benefits. The flipped classroom literature emphasizes that successful flipping requires a clear and credible link between pre-class preparation and in-class activities, and that learning gains are typically mediated by students' engagement in active, collaborative work rather than by the mere availability of recordings [28]. In our case, activating students remains challenging even when instructors attempt to initiate direct, personal interaction during in-class work. This suggests that the bottleneck is not only access to content but also students' willingness to participate in public, interactive problem-solving.

Course evaluations provide a nuanced picture: students frequently describe the discussion phases as a positive component of the course, but at the same time, they note that the number of actively participating students is too low to fully realize the potential benefits. This aligns with the ERP flipped-classroom findings reported by Bhuiyan et al. [13], where learning success in flipped ERP contexts is associated with structured, participation-requiring activities (e.g., concept checks) rather than unstructured time alone. Evaluations also indicate that the synchronous online option is perceived as low value, which aligns with research on hybrid delivery, which stresses the centrality of social presence and interaction quality in shaping learners' perceived value of participation modes [30, 31]. Based on these combined observations and the evaluation feedback, we conclude that offering online participation as an equivalent alternative to on-campus participation does not provide sufficient educational value in our context. Consequently, we do not plan to continue the synchronous online option and will instead prioritize on-campus interactive sessions where student-to-student discussion and instructor feedback can be more reliably facilitated.

4.4. Integrative tasks and student self-regulation

In both ERP1 and ERP2, we intentionally include tasks that require students to mobilize knowledge introduced in prior lectures, and to apply it in representations that are common in information systems practice (process models, data models, queries, and API-oriented reasoning). These tasks are designed to support conceptual integration and help students progress from transaction-level execution to an analytical understanding. However, a consistent observation is that many students express a strong dislike for such tasks, and in several cases, they perform poorly when assessed. This section reflects on two connected issues: (i) students' resistance to tasks that demand transfer of prior knowledge, and (ii) indications of limited academic self-regulation and independence in problem solving.

A recurring theme is that students prefer tasks with immediate procedural payoff ("do X in the system") over tasks that require them to apply prior theoretical material to create an abstract representation or to reason about system structure. In ERP1, a representative example is the assignment in which students are expected to model their process using BPMN. Although BPMN is a common method for documenting and communicating business processes, students often perceive the modeling step as detached from "real" ERP work. In ERP2, similar resistance is observed in the technical theme tasks, including drawing a UML class diagram to capture the observable structure of the ERP system, performing SQL queries on the database, and writing pseudo-code to retrieve equivalent information from the documented API. Notably, performance is weakest for the pseudo-code task, even though many students have previously completed programming courses.

The pseudo-code task is particularly illustrative. Students are asked to express, in an abstract yet precise manner, how information observable in the ERP system could be retrieved from the API. In exam settings, many students either avoid the task or produce superficial answers that do not specify data structures, requests, or control flow in a meaningful way. This suggests that prior programming exposure does not automatically translate into competence in API reasoning. A plausible interpretation is that the task requires a different form of abstraction than typical introductory programming exercises, as it combines domain understanding, information modeling, and interaction logic. In other words, it is less "writing code" and more "expressing an information-retrieval strategy in a technical vocabulary." This again aligns with cognitive and threshold concept perspectives: students may not yet have a stable scheme that connects ERP concepts, data structures, and service interfaces into a coherent mental model that can be externalized as pseudo code.

This behavior suggests that some students have not internalized the norms of academic self-regulation: at the university level, students are typically expected to (i) attempt independent problem solving before seeking assistance, (ii) use resources strategically, and (iii) actively maintain prerequisite skills. In our context, the dependency pattern is consequential because ERP learning requires iterative troubleshooting, careful interpretation of system feedback, and sustained engagement with documentation. If students default to seeking immediate help without making an initial attempt, they reduce their opportunities to develop these core competencies. The literature on troublesome knowledge provides a useful framing: students may experience discomfort when moving from guided procedures to self-directed problem solving, and may respond by seeking external guidance as a way to minimize uncertainty [15, 16].

5. The student perspective of the courses

Our reflection on how students perceive the courses is based on two sources: our *interactions* with students during the three courses, and the course *evaluations*.

Our *interactions* with students take place during lectures, exercises, tutoring sessions, and the final examination. Based on these interactions, one observation stands out: students' perceptions of ERP systems are bimodal. One group of students strongly dislikes ERP systems, mainly because of their complexity and the perceived loss of control. This experience may be compared to working with source code written by someone else: to make effective use of an ERP system, one must first understand another person's design choices and way of thinking. In the course evaluations, these students sometimes answer the question "What was bad about the course?" with a single statement: "Microsoft Business Central" (the ERP system used in the course).

Another group of students, in contrast, genuinely enjoys working with ERP systems. They realize that with a few days of work and a few hundred clicks, they can configure a complete IT environment for an organization. This includes warehouse management, CRM functionality, product structures, bookkeeping, and more. These students appreciate the core idea of standard systems: providing a comprehensive set of functionalities without requiring programming. As a result, they recognize the power they themselves gain from mastering such systems: the ability to deliver business value quickly.

5.1. Evaluation ERP1 and ERP2

Our second source of feedback consists of the course evaluations. We examined the two most recent years of evaluations from the three courses. Approximately 10–15% of students complete the course evaluations; although this response rate is low, it is common within the department. For the courses, we obtained 61 evaluations for ERP1, 36 for ERP2, and 19 for the smaller PROAFF course. We analyzed questions about what students perceived as good and bad, and about suggestions for improvement. Since ERP1 and ERP2 are structurally similar, we combined their results, yielding 97 comments that were analyzed for themes. For PROAFF, we analyzed 42 comments.

Across responses for ERP1 and ERP2, a prominent theme is the value of authentic, hands-on engagement with an enterprise system. Students repeatedly framed the course as "real" and professionally relevant, emphasizing that working directly in the ERP environment supported understanding beyond abstract description. Typical comments highlighted that "That you got to work with a real system" and that practical tasks made learning feel more applied: "It was fun to do the exercise and the lab; it feels more 'real' than just theoretical." This experiential focus was also linked to perceived employability and deeper insight into what business systems entail.

A second (positive) recurring theme concerns instructional competence, support, and course structure. Several valued aspects of the course organization of ERP1 and ERP2, and flexibility, especially recorded materials: "That the lectures were recorded so you could watch them when you yourself had the time and opportunity." At the same time, the same design choices (notably flipped classroom) were experienced by others as reducing engagement and clarity, with critiques such as "I didn't like the flipped classroom" and concerns that a few recordings were incomplete or hard to follow.

A third theme centers on misalignment and friction in assessment, workload, and learning activities. There was a perceived disconnect between theory, project work, and labs—one student described it as "Almost as if you were taking two different courses." Practical implementation also generated friction: system instability ("Very slow and sluggish"), constraints on participation ("That only one person can work in the system [at a time]"), uneven group contribution, and limited access to timely help ("Fewer teaching assistants … [and] we kept being told it had to go quickly"). Proposed changes, therefore, converged on improving alignment (e.g., concept lists, updated instructions), strengthening support capacity, and reducing technical and collaboration bottlenecks.

5.2. Evaluation PROAFF

In contrast to ERP1 and ERP2 the PROAFF course is solely focused on a larger open-ended project with an ERP system. The course structure is thereby different; it simulates a realistic project where regular status meetings are held with a "customer".

Across the student responses, several coherent themes emerge regarding the course's strengths and areas for improvement. A dominant positive theme concerns engaged and supportive teaching, with students repeatedly emphasizing the instructor's approachability, exemplified by comments such as "the engagement made a big difference, and you really felt the support". Closely related is the theme of authentic, integrative project-based learning, where students valued working on a large, open-ended project that connected prior coursework; "it tied together knowledge from earlier courses and gave us clear frames but freedom to build the content". A third theme highlights collaborative structures and feedback, particularly weekly status meetings and scrum-style planning, which were seen as motivating and inspiring: "The status meetings meant you could get inspiration, discuss what you had done, and receive feedback". In contrast, critical feedback clustered around the need for sufficient guidance and practical support, especially early in the course and during technical implementation. Students noted a lack of scheduled supervision and clarity, as illustrated by the concern that "I didn't always understand what was required, both in terms of content and assessment criteria". Overall, the feedback portrays a highly engaging course with a strong project focus, alongside a clear desire for more structured guidance and technical support.

Overall, the comments about a lack of guidance and support are not surprising: The students are free to select their own tools; some selected Microsoft Power BI, some Google App Script, and so on, thereby providing support for these tools is not amenable within the course budget.

6. Conclusions

This paper presents reflections on a three-course ERP curriculum at Stockholm university, positioning our teaching practices in relation to ERP education research. Our experience confirms prior work that experiential, practice-driven designs are well-suited to ERP education, particularly when they help students develop an integrated view of business processes rather than isolated transaction skills [5, 6].

Taken together, our experiences suggest several practical recommendations for ERP educators. First, course activities should be designed explicitly for enterprise integration, including end-to-end tasks, and projects that require cross-role reasoning. This approach helps students develop a process orientation and understand interdependencies, rather than merely learning isolated transactions [6]. Second, in flipped classroom settings, contact time should be reserved for activities that make students' reasoning visible. Structured participation mechanisms, such as concept checks, peer explanations, and short oral reasoning exercises, are typically more effective than unstructured "work time" because they create accountability and provide opportunities for immediate feedback [28, 13]. Third, modeling and analytical artifacts should be treated as core elements of ERP competence rather than optional add-ons.

ERP ecosystems and student practices continue to evolve rapidly. For our curriculum, future development will focus on (i) strengthening mechanisms for active participation and accountability in contact sessions, (ii) refining interventions around the identified threshold concepts, (iii) improving scaffolding for API and data-oriented reasoning, and (iv) introducing explicit AI literacy activities and assessment designs that reward understanding rather than text production.

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT-5.4 and Grammarly in order to: Grammar and spelling check. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] T. H. Davenport, Putting the enterprise into the enterprise system, Harvard Business Review 76 (1998) 121–131.

[2] H. Klaus, M. Rosemann, G. G. Gable, What is erp?, Information Systems Frontiers 2 (2000) 141–162.

[3] E. E. Watson, H. Schneider, Using erp systems in education, Communications of the Association for Information Systems 1 (1999).

[4] M. I. Wijaya, A decade of erp teaching practice: A systematic literature review, Education and Information Technologies 28 (2023) 14117–14137.

[5] U. Ruhi, An experiential learning pedagogical framework for enterprise systems education in business schools, The International Journal of Management Education 14 (2016) 198–211.

[6] R. Seethamraju, Enhancing student learning of enterprise integration and business process orientation through an erp business simulation game, Journal of Information Systems Education 22 (2011) 19–30.

[7] L. Motiwalla, J. Thompson, Enterprise Systems for Management: Pearson New International Edition, Pearson Education, 2013.

[8] S. A. Bin-Nashwan, M. Sadallah, M. Bouteraa, Use of chatgpt in academia: Academic integrity hangs in the balance, Technology in Society 75 (2023) 102370.

[9] R. Deng, M. Jiang, X. Yu, Y. Lu, S. Liu, Does chatgpt enhance student learning? a systematic review and meta-analysis of experimental studies, Computers & Education (2024) 105224.

[10] P. Hawking, B. McCarthy, Integrating e-learning content into enterprise resource planning (erp) curriculum, Issues in Informing Science and Information Technology 1 (2004) 22–30.

[11] M. Hwang, K. Cruthirds, Impact of an erp simulation game on online learning, The International Journal of Management Education 15 (2017) 60–66.

[12] V. Gerogiannis, P. Fitsilis, A project-based learning approach for teaching erp concepts, International Journal of Learning 12 (2006) 261–268.

[13] S. Bhuiyan, S. Goodarzi, J. Large, C. Serrano, S. Bristow, Understanding erp learning success in flipped classrooms, in: Proceedings of the Americas Conference on Information Systems (AMCIS), 2018.

[14] G. Akçayır, M. Akçayır, The flipped classroom: A review of its advantages and challenges, Computers & Education 126 (2018) 334–345.

[15] J. H. Meyer, R. Land, Threshold concepts and troublesome knowledge (2): Epistemological

considerations and a conceptual framework for teaching and learning, Higher education 49 (2005) 373–388.
[16] G. Cousin, An introduction to threshold concepts, Planet 17 (2006) 4–5.
[17] R. O'Donnell, A Critique of the Threshold Concept Hypothesis and an Application in Economics, Working Paper Series 164, Finance Discipline Group, UTS Business School, University of Technology, Sydney, 2010. URL: https://ideas.repec.org/p/uts/wpaper/164.html.
[18] K. A. Alshare, P. L. Lane, Predicting student-perceived learning outcomes and satisfaction in erp courses: An empirical investigation, Communications of the association for information systems 28 (2011) 34.
[19] C. Cope, L. Staehr, Improving student learning about a threshold concept in the is discipline, Informing Science: The International Journal of an Emerging Transdiscipline 11 (2008) 349–364.
[20] J. Sweller, Cognitive load theory, learning difficulty, and instructional design, Learning and Instruction 4 (1994) 295–312.
[21] F. Paas, A. Renkl, J. Sweller, Cognitive load theory and instructional design: Recent developments, Educational Psychologist 38 (2003) 1–4.
[22] C. Lambeck, R. Groh, Mastering erp interface complexity-a scalable user interface concept for erp systems, in: International Conference on Enterprise Information Systems, volume 2, SCITEPRESS, 2013, pp. 170–178.
[23] J. Sweller, Cognitive load during problem solving: Effects on learning, Cognitive science 12 (1988) 257–285.
[24] C. Baek, T. Tate, M. Warschauer, "chatgpt seems too good to be true": College students' use and perceptions of generative ai, Computers and Education: Artificial Intelligence 7 (2024) 100294.
[25] R. H. Mogavi, C. Deng, J. J. Kim, P. Zhou, Y. D. Kwon, A. H. S. Metwally, A. Tlili, S. Bassanelli, A. Bucchiarone, S. Gujar, L. E. Nacke, P. Hui, Chatgpt in education: A blessing or a curse? a qualitative study exploring early adopters' utilization and perceptions, Computers in Human Behavior: Artificial Humans 2 (2024) 100027.
[26] L. K. C. Yeung, D. P. L. Chow, P. H. Wong, S. S. S. Lau, In chatgpt they trust: a study of students' perceptions and misuse of chatgpt in higher education, AI and Ethics 6 (2026) 11.
[27] M. C. Murray, Bridging the generative ai literacy gap: A guide to introducing prompt engineering in university courses, Issues in Informing Science and Information Technology 22 (2025) 010.
[28] J. O'Flaherty, C. Phillips, The use of flipped classrooms in higher education: A scoping review, The Internet and Higher Education 25 (2015) 85–95.
[29] H. Donelan, K. Kear, J. Rosewell, A. Okada, K. Sheehy, K. Amor, C. Edwards, A. Mooney, P. Cuffe, T. Elder, Synchronous online learning: why some students don't actively participate, Online Learning 29 (2025) 311–337.
[30] L. Breitschwerdt, C. Hümmer, R. Egetenmeyer, Online and on-site participation in synchronous hybrid settings: Reasons from the perspective of higher education students, Higher Education (2025) 1–22.
[31] A. Raes, L. Detienne, I. Windey, F. Depaepe, Learning and instruction in the hybrid virtual classroom: An investigation of students' engagement and the effect of quizzes, Computers & Education 143 (2020) 103682.