Revista Brasileira de Informática na Educação – RBIE
Brazilian Journal of Computers in Education
(ISSN online: 2317-6121; print: 1414-5685)
https://sol.sbc.org.br/journals/index.php/rbie

Submission: 06/03/2025; 1st round notif.: 23/11/2025;
Camera ready: 08/12/2025; Edition review: 09/12/2025; Available online: 09/12/2025; Published: 09/12/2025;

Designing an Academic Software Factory for AI-Based Systems Development from Professors' Perspectives

Alinne C. Corrêa Souza
Federal Technological University of Paraná
(UTFPR – Dois Vizinhos)
ORCID: 0000-0002-2525-7801
alinnesouza@utfpr.edu.br

Francisco Carlos Monteiro Souza
Federal Technological University of Paraná
(UTFPR – Dois Vizinhos)
ORCID: 0000-0002-8953-8653
franciscosouza@utfpr.edu.br

Thiago Damasceno Cordeiro
Federal University of Alagoas (UFAL)
ORCID: 0000-0003-0976-7040
thiago@ic.ufal.br

Abstract

The increasing complexity of AI-based systems poses significant challenges in software engineering education, requiring an integrated approach that combines AI techniques with software development methodologies. Traditional educational models often fail to bridge this gap, limiting students' ability to apply theoretical knowledge to real-world AI-driven software solutions. This paper proposes the design of an Academic Software Factory (ASF) tailored for AI-based systems development within undergraduate software engineering education. The proposed ASF integrates Competency-Based Learning, Bloom's Taxonomy, and Agile Scrum methodologies to provide a structured and industry-aligned learning experience. The ASF framework is developed through a four-phase methodology: (i) identifying characteristics and roles in AI-based system development, (ii) designing the ASF curriculum, (iii) defining a teaching methodology aligned with AI-specific software engineering challenges, and (iv) evaluating the ASF design through expert validation. A survey was conducted with ten professors specializing in software engineering and AI for preliminary validation. The feedback highlighted the ASF's potential to enhance educational outcomes, suggesting refinements such as assigning specialist roles to faculty members, reducing redundancy in pre-existing content, and emphasizing developing and presenting a minimum viable product as a key learning outcome.

Keywords: *Academic Software Factory; AI-Based Systems; Software Engineering; Survey.*

Cite as: Souza, Alinne C. C., & Souza, F. C. M., & Cordeiro, T. D. (2025). Designing an Academic Software Factory for AI-Based Systems Development from Professors' Perspectives. Revista Brasileira de Informática na Educação, vol. 33, 1567–1595. https://doi.org/10.5753/rbie.2025.5647.

1 Introduction

In academia, a software factory provides a collaborative environment where students work on real-world projects, using systematic production methods to design, develop, and test software components in controlled environments, enhancing productivity and integrating diverse development practices (X. Wang et al., 2014). Establishing a software factory dedicated to intelligent systems in undergraduate computing presents a unique opportunity. Such a factory can serve as a practical training ground for students, enabling them to gain hands-on experience with cutting-edge technologies and methodologies. By simulating real-world development environments, a software factory can bridge the gap between theoretical knowledge and practical application, better-preparing students for industry demands (Tvedt et al., 2001).

Developing intelligent systems that employ artificial intelligence (AI) involves addressing numerous challenges, primarily from their complexity and the integration of diverse technologies. These challenges encompass (i) defining requirements for systems that learn and adapt, (ii) modeling systems to accommodate dynamic and uncertain environments, (iii) selecting suitable algorithms from a wide range of artificial intelligence techniques, (iv) managing and pre-processing large datasets critical for effective algorithm development; (v) employing testing methods to ensure reliability and performance; and (vi) maintaining infrastructure, which often requires high-performance computing resources and specialized hardware, thereby increasing complexity.

For software engineering students, the challenge lies in the growing need to integrate knowledge from traditionally separate disciplines. Students often face difficulties applying software development methodologies into AI-based systems due to their unique characteristics. This is mainly because there are few well-established techniques for the stages of the software development lifecycle for AI-based systems, and those that do exist have yet to be widely known (Martínez-Fernández et al., 2022).

The absence of a dedicated curriculum for developing software products that utilize AI algorithms further exacerbates these difficulties. Students often need more opportunities to apply software engineering principles in the context of AI, making it challenging to bridge the gap between theoretical knowledge and practical application. In response to this, this paper proposes the design of an Academic Software Factory (ASF) tailored for AI-based systems within the course for a Bachelor's Degree in Software Engineering. We use the Action Research cycle (Thiollent, 2005) to assist the ASF, provide students with hands-on experience, and equip them with the skills to integrate AI techniques and software development methodologies, thereby creating robust AI-based systems.

We surveyed ten professors with software engineering and artificial intelligence expertise for preliminary validation of the ASF proposal and obtained feedback. The feedback was positive, highlighting the potential of the factory to improve educational outcomes. The professors suggested a more streamlined process, including i) considering some professors/tutors of the discipline as specialist roles and emphasizing ii) reducing content in subjects that have already been seen during graduation, and iii) creating and presenting a minimum viable product at the end of the course.

Overall, the contributions of the present paper can be summarized into the following points: *i*) Design of an Academic Software Factory for AI-based systems development integrating Competency-

Based Design, Bloom's Taxonomy, and Scrum; *ii*) Definition of roles and responsibilities in AI-based software teams; *iii*) Formulation and development of a curriculum aligned with industry and academic needs; and (iv) Empirical validation through expert evaluation.

The remainder of this paper is structured as follows: Section 3 presents related works, highlighting existing approaches to software factories and their applications in education. Section 4 details the design of the proposed Academic Software Factory, including its curriculum, methodology, and role definitions. Section 5 discusses the results obtained from the evaluation conducted with experts, providing insights into the ASF's effectiveness and areas for improvement. Finally, Section 6 presents concluding remarks and future research directions.

2 Background

2.1 Software Factory

In the literature, various definitions of Software Factory highlight its structured and process-oriented nature. A Software Factory is an environment where software development activities follow predefined processes, incorporating best practices from industrial manufacturing, such as standardized workflows, quality models, compliance with schedules, and structured deliveries (Borsoi, 2008). These concepts are closely related to software product lines, domain-specific languages, specialized computational tools, and standardized processes, ensuring efficiency and repeatability in software production.

Beyond its industrial applications, the Software Factory model can be adapted for educational settings, integrating theoretical knowledge with practical experience. In academia, a Software Factory serves as an educational environment where students engage in real-world software development challenges, reinforcing their learning through hands-on application. Ahmad et al., 2014 describe Software Factories as testbeds for software engineering ideas and sources for scientific research on software development. Additionally, they emphasize their role as educational vehicles for universities, where the artifacts produced not only enhance student learning but also serve as instructional materials, fostering collaboration between academia and industry.

An example can be found in Tvedt et al., 2001, which proposed a model that integrates traditional computer science coursework with experience-based learning in a Software Factory environment, where students actively participate in real software development projects. Their approach immerses students in all roles of the software development lifecycle, allowing them to gain practical experience as requirements analysts, developers, testers, and project managers. This structure prepares students for industry demands, ensuring they develop both technical expertise and project management skills while working in team-oriented settings. By adopting the Software Factory model, students gain hands-on experience in a controlled, structured, and collaborative environment, where they can apply software engineering methodologies, explore innovative approaches, and develop industry-relevant competencies.

Also, to keep pace with technological advancements, particularly in artificial intelligence, the Software Factory model must evolve to address the unique challenges of intelligent systems development. The increasing integration of AI into software solutions has created a demand for

Intelligent Systems Factories (ISFs) to provide students with practical experience in designing, training, and deploying AI-driven applications. While traditional Software Factories emphasize standardized workflows and engineering best practices, an ISF must incorporate data-driven experimentation, model optimization, and continuous learning, reflecting AI-based development's iterative and adaptive nature.

In addition to conventional software engineering roles, an intelligent systems factory requires AI specialists, data scientists, and domain experts, fostering interdisciplinary collaboration essential for AI-driven innovation. Ethical considerations such as algorithmic bias, transparency, and responsible AI deployment must also be embedded into the development lifecycle.

2.2 AI-based system

Artificial Intelligence is the branch of computer science that focuses on creating mechanisms capable of executing tasks typically performed by humans, and it includes subfields with different goals (Anthes, 2017). However, in this work, we will address intelligent agents, evolutionary computing, and pattern recognition due to their versatility in solving complex problems and for being widely used successfully in various domains as (Jiang et al., 2020; Kaswan et al., 2022) and (Z. Wang et al., 2022; Zhou et al., 2022) and (Li et al., 2020; Xie et al., 2023).

Intelligent agents are systems capable of observing the environment and acting autonomously to meet their designed objectives, demonstrating planning and decision-making behaviors (Bartneck et al., 2021). Evolutionary computing is a subfield inspired by biological evolution and employs techniques such as genetic algorithms to solve complex optimization problems through genetic operators (Russell & Norvig, 2020). Machine Learning enables systems to learn from data, improve through experience, and make decisions without being explicitly programmed for each task (Russell & Norvig, 2020) and (Bishop, 2006).

For the software engineering context, an artificial intelligence-based system is a technology that can be *1*) an intelligent component developed in software; *2*) an intelligent service that provides some resource to one or more software; *3*) an entire software of AI or *4*) an AI embedded in hardware devices accessed by software (Bartneck et al., 2021) (Figure 1). Intelligent Systems find applications across diverse domains such as healthcare, where they might predict patient outcomes, in finance for fraud detection, or in autonomous vehicles for navigating and making real-time decisions.

Although intelligent systems have unique characteristics and capabilities, their development follows the conventional software development lifecycle, encompassing problem identification, requirements gathering, modeling, design, implementation, testing, and maintenance. Nonetheless, it is crucial to recognize the distinctive requirement for algorithm development and validation inherent to each AI subfield. Intelligent systems rely heavily on specialized algorithms, each with its unique pipeline.

In a simplified way, intelligent agents often follow a pipeline that includes modeling, developing, and validating the perception of the environment, interpretation of the perceived data, decision-making based on rules, and acting upon those decisions. In evolutionary computation, the pipeline starts with modeling the problem as a search problem, choosing a structure to represent candidate solutions, and creating fitness functions. An algorithm can be developed from these

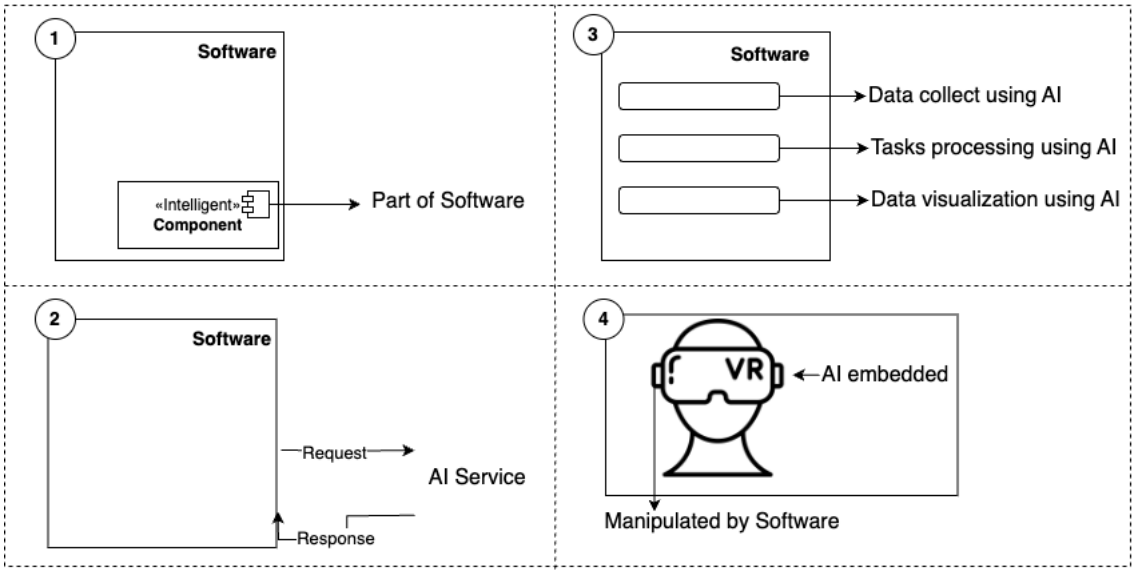

Figure 1: Types of AI-based systems.

essential elements, and an experimental analysis can be performed. Finally, pattern recognition pipelines generally begin with data collection and pre-processing for training. Feature selection or extraction is then performed for the learning task. A model is trained using this data, followed by testing and validation to assess its performance.

It is essential to underscore that developing an intelligent algorithm for a specific domain demands an experimental study reflective of the intricate nature of the problems. Therefore, an in-depth experimental analysis is crucial for assessing the algorithm's efficacy and adaptability across diverse situations.

2.3 Bloom's taxonomy

Bloom's Taxonomy consists of a hierarchical set of categories that describe different types of cognitive learning, widely used across various fields of education. Created in 1956 by Benjamin Bloom and his collaborators, it serves as a tool providing a foundation for developing assessment instruments and integrating simple skills to master more complex ones (Momen et al., 2022). Bloom's taxonomy, in its revised version Anderson and Krathwohl (2001), is composed of six levels of cognitive learning organized in ascending order of complexity (Figure 2).

To achieve a cognitive objective at a higher level, it is implied that the objectives of the previous levels have been attained and mobilized. The base of the Bloom's taxonomy —*Remembering*—represents skills in which students must recall specific facts. The next level — *Understanding* — represents skills in which students must grasp the meaning of instructional materials. At the next level — *Applying* — students must use information in a new (but similar) situation to one they have practiced in the past. At the *Analyzing* level, students must take apart and identify relationships among the material that is known. At next to highest stage—*Evaluating*—students examine information and make judgments. At the top of the Bloom's taxonomy—*Creating*—Students use

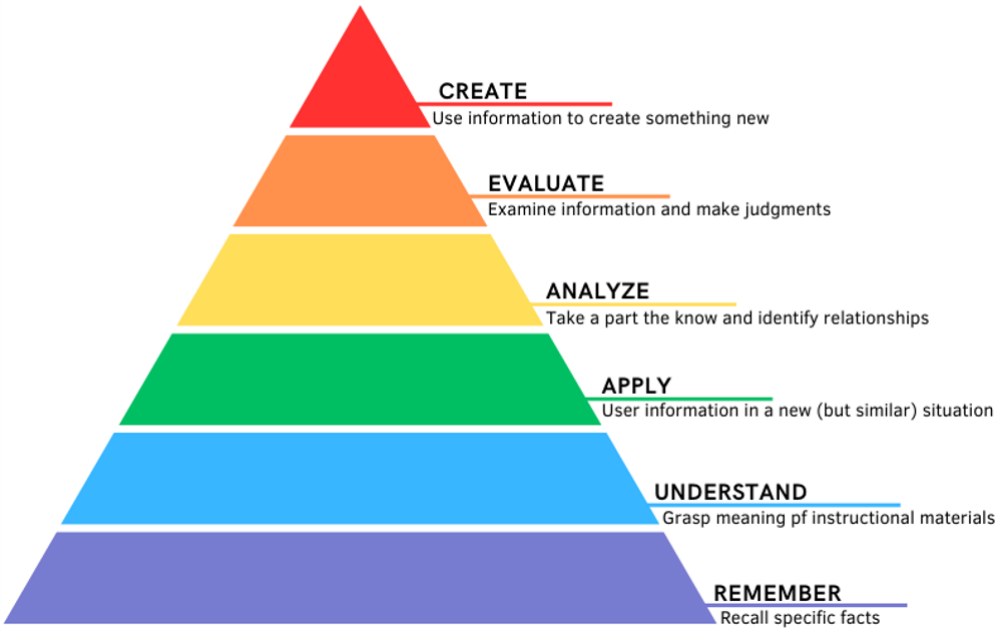

Figure 2: Bloom's taxonomy levels.

information to create something new. Figure 3 presents Verbs grouped in Bloom's taxonomy levels revised by Anderson and Krathwohl (2001).

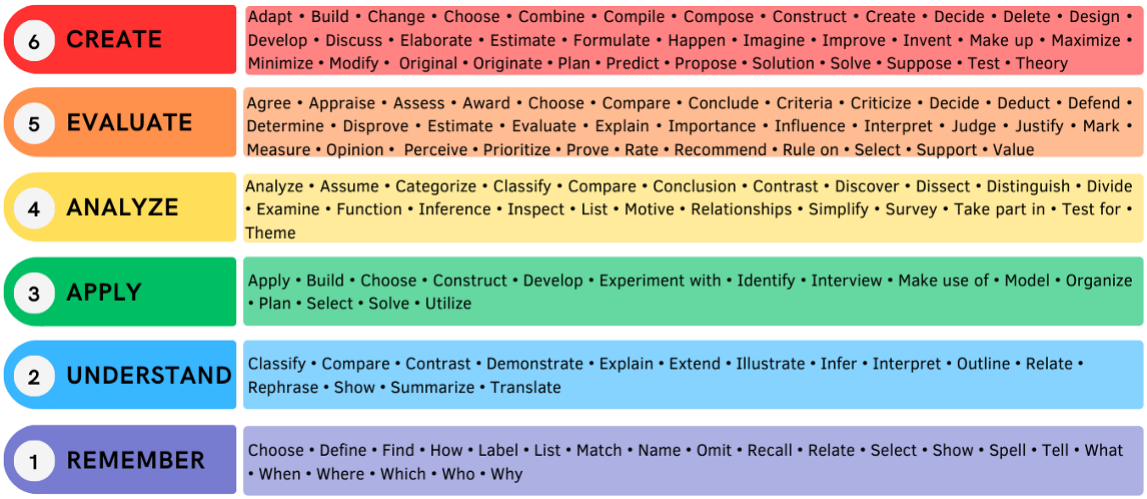

Figure 3: Verbs grouped by Bloom's taxonomy levels.

Bloom's taxonomy is probably the most widely used educational taxonomy for specifying learning objectives in Computer Science (Masapanta-Carrión & Velázquez-Iturbide, 2017), and it is even recommended in the ACM/IEEE curriculum (Ardis et al., 2015).

2.4 Competency-based Design

Competency-based design is an approach that focuses on the development of skills and the mobilization of knowledge. In this work, the definition of competence adopted by Scallon (2017) is used, which refers to the "possibility, for an individual, of mobilizing an integrated set of resources in an internalized manner to solve a family of problem-situations" (Roegiers & Ketele, 2001). In other words, students are prepared to act in varied situations that resemble the daily life of the profession or occupation learned through teaching and assessment methodologies that allow the acquisition of these competencies.

In the competency-based design approach, three types of knowledge are distinguished: i) declarative (or *knowing*), which corresponds to theoretical and formal knowledge, such as rules, techniques, norms, etc.; ii) procedural (or *know-how*) which involves the appropriate combination of knowledge and develops in the practice of the learned profession or occupation; and iii) conditional (or *know-being*) which includes affective, social, and worldview aspects, such as autonomy, critical thinking, creativity, collaboration, among others.

3 Related Works

This section reviews five studies on integrating software factories into educational curricula, emphasizing various methods to bridge the gap between theoretical knowledge and practical skills. Chao and Randles (2009) discuss an Agile Software Factory at a university to enrich student learning by combining community service with software engineering education. The initiative addressed challenges like project sustainability by creating a project repository and involving students in ongoing services. Courses included Agile methodologies, UML modeling, project planning, and customer interaction. Plans involve continuous assessments of community engagement, project completion rates, customer satisfaction, expanding service-learning, and promoting Agile practices through research and education.

Ahmad et al. (2014) aimed to identify factors impacting student learning, academic achievements, and professional skills within the Software Factory (SWF) environment. Using instruments like the Computer Laboratory Environment Inventory and Attitude Towards Computers and Computing Courses Questionnaire, supplemented with constructs for Kanban board and collaborative learning, data were gathered via an online survey of master's degree students. Findings showed that SWF's environment, cooperative learning, and Kanban board positively influenced learning outcomes and skill development. Students reported high satisfaction, noting the course's relevance to future careers. The study concluded that SWF effectively prepares students for real-world software engineering challenges, offering an impact learning environment.

The study conducted by Oliveira et al. (2017) aimed to assess the significance of an interdisciplinary software factory, explore innovative pedagogical methods to engage students in the learning process, and identify the skills demanded by the industry. Their research involved an exploratory survey with 104 participants, including IT professionals, educators, and students from diverse educational institutions. Findings revealed that 96% of educators favored problem-based learning as the most effective approach, while 88% of students perceived participation in a software factory as advantageous to their education. Furthermore, 87% of professionals expressed

concerns that educational institutions primarily impart theoretical knowledge, often failing to prepare students for real-world challenges adequately.

The paper of Santos et al. (2021) reported on a teaching project conducted at the Federal Technological University of Parana to integrate real-world software development into the Bachelor of Software Engineering curriculum. Over five months, professors and five students from different semesters participated in the project, which aimed to bridge the gap between academic learning and industry demands by simulating a software development environment. The project followed Agile Scrum methodology, with bi-weekly sprints to incrementally develop the system. The students reported that participating in the entire development process, from conception to final product, and working with real clients helped them understand the complexities and demands of professional software development.

The reviewed studies demonstrate the importance of Software Factories in educational settings, focusing on their role in bridging the gap between theoretical knowledge and practical experience. Each study explores distinct methodologies, such as Agile practices (Chao & Randles, 2009), collaborative learning (Ahmad et al., 2014), and problem-based learning (Oliveira et al., 2017), all of which contribute to improving students' preparedness for the industry. As presented in Table 1, unlike these works, the present research proposes an ASF for AI-based systems, addressing AI-specific development challenges such as algorithm training, data management, and ethical considerations. While previous studies analyze the benefits of Software Factories in traditional software engineering education, our work extends this concept to intelligent systems development, incorporating roles such as AI engineers, data scientists, and domain experts.

However, none of these studies specifically address the implementation of intelligent systems factories. A more comprehensive literature review is needed to confirm this gap and identify potential opportunities for intelligent systems factories to bridge it, ensuring that educational practices evolve alongside software industry demands.

4 Academic Software Factory Project Design for AI-based Systems

We propose and evaluate the design of an ASF course for AI-based systems development in the Software Engineering undergraduate from teachers' perceptions regarding the curriculum, roles, and methodology proposed for ASF conduction.

The primary motivation for applying the ASF concept to this course was to create an environment closer to a real software development organization. Besides, the software factory characteristics introduce some other advantages, e.g., students use their knowledge in a controlled environment (which allows teachers to track them more closely) and well-defined processes (which ease the understanding of what should be done). Figure 4 presents the process adopted for the Design of the ASF for AI-based systems development, divided into four phases, 11 activities, and four results, one per phase.

Table 1: Comparison of Related Works and the Proposed ASF.

Study	**Main Focus**	**Educational Approach**	**Software Development Focus**	**Empirical Validation**
Chao and Randles (2009)	Agile Software Factory	Community service + Agile methods	Traditional software projects	Project sustainability, user satisfaction
Ahmad et al. (2014)	Learning Outcomes in SF	Collaborative learning, Kanban	Traditional software projects	Student feedback and performance evaluation
Oliveira et al. (2017)	Interdisciplinary SF	Problem-based learning, industry engagement	Traditional software projects	Survey with students, educators, and professionals
Santos et al. (2021)	Real-world project experience	Scrum, client collaboration	Traditional software projects	Qualitative student feedback
Our Research	**AI-Based Software Factory**	**Competency-Based Learning, Bloom's Taxonomy, Scrum**	**AI-based systems**	**Empirical validation through expert evaluation**

4.1 Phase 1 - Identification of characteristics and roles in AI-based systems development

In this we comprehend the concepts and unique characteristics that define AI sub-areas, we turn to the literature such as Russell and Norvig (2020) for Intelligent Agents (IA), Simon (2013) for Evolutionary Computing (EC), and Bishop (2006) for Pattern Recognition (PR). We recognized the need to understand these subfields more deeply, aim to generalize common activities and address their unique distinctions at the software development level. The subfields and their focus, goals, and examples are presented in Table 2.

Furthermore, we analyzed eight studies (Souza et al. (2024), Franch et al. (2023), Khuat et al. (2023), Pei et al. (2022), Nahar et al. (2022), Deshpande and Sharp (2022), Barclay and Abramson (2021) and Vogelsang and Borg (2019)) to identify the different roles and their collaborations during AI-based systems development process. Although the studies specifically address RE activities for ML systems, we identify them as potential for the various types of AI subfields. We analyzed these papers that address the team roles in AI-based system development and their responsibilities (Table 3).

Based on the roles presented in Table 3, we decided to use the roles defined by Souza et al. (2024) works once these roles are flexible and can become part of the development process based on the characteristics of the AI-based system. Our vision aligns with agile software development using Scrum because the leading roles are Product Owner, Scrum Master, and developers team (Schwaber & Sutherland, 2020), and it is the most widely adopted agile methodology (Digital.ai,

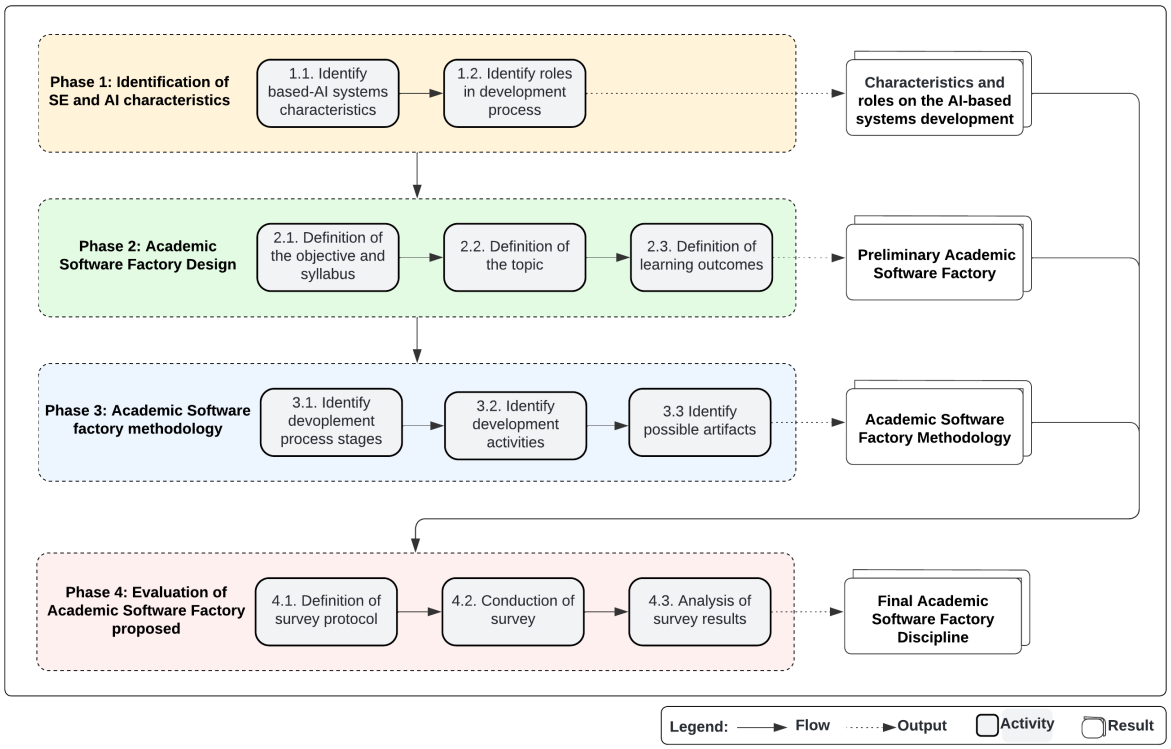

Figure 4: Methodology for design of the ASF for AI-based systems..

2023). Figure 5 illustrates the roles in AI-based systems development considering the Scrum framework, their skills, and responsibilities defined in Souza et al. (2024) works.

4.2 Phase 2 - Academic Software Factory Design

The ASF was designed considering the context of the course "Software Factory for AI-based system development' offered to undergraduate students (6th semester) of Software Engineering at University X. It will last one academic semester (16 weeks) and have a total workload of 320 hours, with 20 hours per week. The course intends to integrate the practical activities developed by students (supported by professors) into the routine of AI-based systems development. This scenario implies professors' direct and effective participation in the planning, controlling, and evaluating actions developed in ASF projects.

ASF should have some courses from the SE undergraduate curriculum as prerequisites; that is, students can enroll in an ASF course only after completing these courses from the SE curriculum. This constraint ensures that students know how to execute all tasks demanded by the project since these tasks may involve practices in all kinds of SE activities and AI knowledge within projects carried out at SF.

We design the ASF curriculum using a competency-based approach, incorporating knowledge, know-how, and know-being development and integration Scallon (2017), Bloom's Taxonomy (Anderson & Krathwohl, 2001) and Action Research cycle (Thiollent, 2005) to assist methodology procedure.

Table 2: Overview of subfields, focus, goals, and examples in AI-based systems.

Subfield	Focus	Goals	Examples
IA	Control and automation problems	Systems can perceive their environment through sensors and act upon it using actuators to achieve their goals.	In the agricultural industry, more robust irrigation management systems are necessary. In these systems, sensors installed in the farm field collect real-time data on soil moisture, weather conditions, and other relevant factors that assist in decision-making on when and how much to irrigate each part of the planting.
EC	Optimization problems	Algorithms that traverse large search spaces to find optimal solutions to complex problems.	Logistics companies face the complex task of distributing products efficiently to minimize costs with the process, material, and delivery time. Thus, it is possible to model algorithms that reduce costs with the number of boxes, space for box allocation, delivery costs, fuel, etc.
PR	Pattern recognition problems	Allow models to learn and make classifications, predictions, forecasts, or decisions based on data.	Banks recurrently need to analyze large amounts of data to assess potential customers and recommend financial products. In this scenario, it is possible to analyze customers' transaction history and identify consumption behavior patterns, spending profiles, and economic preferences.

The competency-based design refers to the "possibility, for an individual, of mobilizing an integrated set of resources in an internalized manner to solve a family of problem situations" (Roegiers & Ketele, 2001). The competency-based design communicates what is expected of students at each stage of their learning journey. The project design of the ASF course was inspired by answering *"What should each student know how to do?"* following competency-based design approach. We used the revised Bloom's Taxonomy to establish the course's learning objectives, guiding professors in planning and developing activities that enhance comprehension and cognitive development (Anderson & Krathwohl, 2001). This process is composed of four main steps.

- **Step 1 – Identification of Essential Competencies:** identifying the key competencies students need to acquire to design and develop AI-based systems.
- **Step 2 – Definition of Learning Outomes (LOs):** based on the identified competencies, clear and measurable learning outcomes are set. Each objective describes what students should be able to achieve by the end of the course or each phase of the process.
- **Step 3 – Curriculum Development:** The curriculum is designed to integrate the identified competencies cohesively. This may involve selecting theoretical content, practical case studies, software development projects, and other activities that help students apply their knowledge in practice.

Table 3: Roles involved in AI-based systems development.

Reference	Roles
Souza et al. (2024)	Regulatory expert, End-User, Product Owner, Domain Expert, Software Architect, DevOps Analyst, AI Engineer, UX Designer, Scrum Master, Developer, Tester, Data Scientist, Statistics and Mathematics Experts and Computer/Mechatronic Engineer
Franch et al. (2023)	Requirements Engineer, Customer, Domain Expert, Data Engineer, AI Engineer, Software Engineer, Ethics Manager, Regulations Expert
Khuat et al. (2023)	business analysts, domain experts, and project managers, data scientists, data analysts, ML engineers, software developers, and system engineers, governance staff, third-party auditors, and government agencies, user that interacts with solution
Pei et al. (2022)	business experts, requirement engineer, domain expert, software engineer, data scientist
Nahar et al. (2022)	Domain Expert, Managers, Data Scientist, Software Engineer, End User
Deshpande and Sharp (2022)	Researchers, AI experts, HCI experts, developers, engineers, technology companies, professional bodies and research institutes; and regional/national legislative/regulatory agencies, Non-users of AI systems, users
Barclay and Abramson (2021)	Domain Practitioners, Systems Integrator, ML Engineer, Data Scientist
Vogelsang and Borg (2019)	Legal experts, requirement engineer, Data scientist, developers, ML experts

- **Step 4 – Aligned with Competencies:** teaching methodologies that promote the development of desired competencies are chosen, such as project-based learning, case studies, and simulations, among others.

The ASF curriculum, illustrated in Figure 6, was created using Competency-based Design and Bloom's taxonomy illustrated. In the curriculum, we defined the Objective/Summary (Figure 6-A) using study themes (Figure 6-B). Each study theme was composed of a set of curricular contents (Figure 6-C), and for each, we identified the respective LOs (Figure 6-D).

4.3 Phase 3 - Definition of teaching methodology

For the execution of ASF and considering the complexity of AI-based systems, we decided that the most appropriate methodology is Scrum. Given the particularities of these types of systems, we will adapt the Scrum process into sprints of defined duration. During each sprint, essential ceremonies, such as planning and retrospectives, will be held to align the team, track progress, and identify opportunities for improvement. At the end of each sprint, the generated artifacts (such as prototypes, code, documentation, and models) will constitute the project deliverables, ensuring iterative evaluations and continuous evolution of AI systems and algorithms.

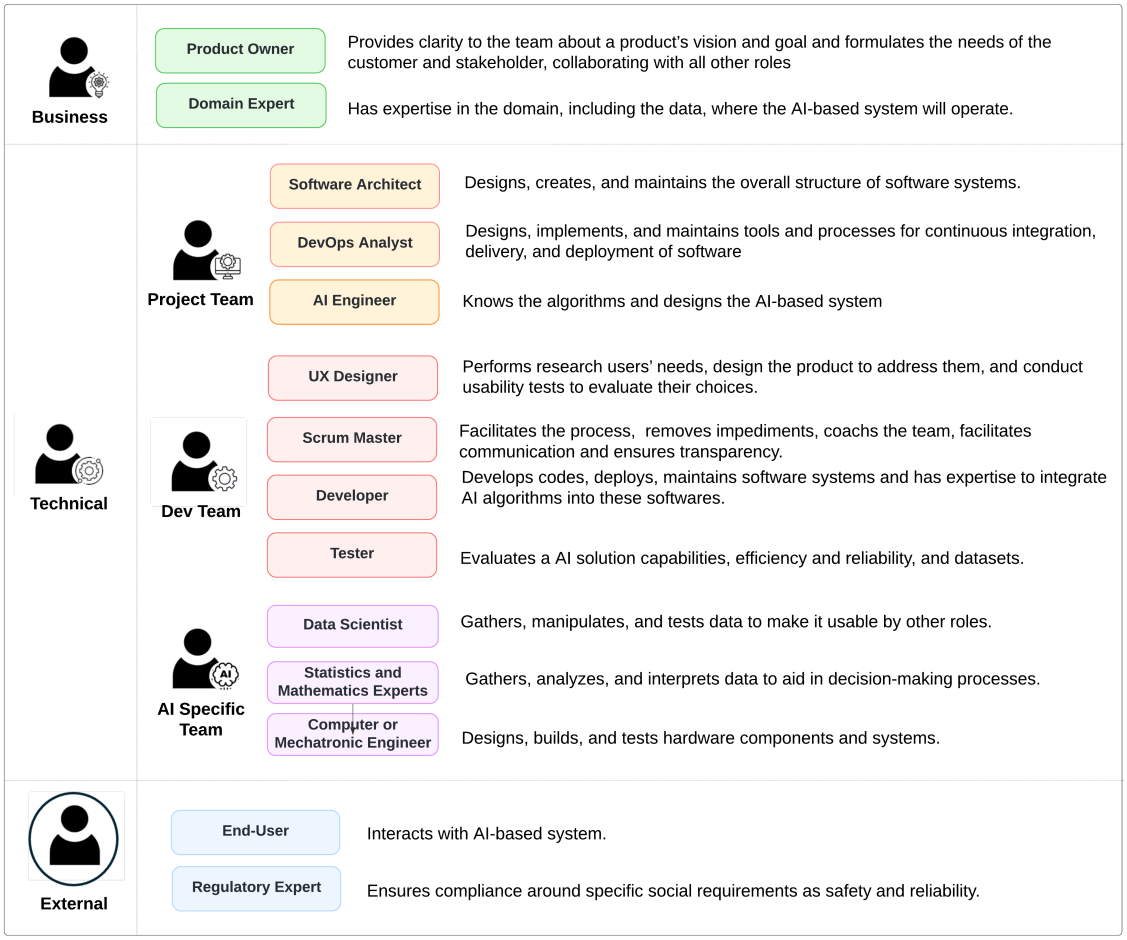

Figure 5: Roles in AI-based systems development identified by Souza et al. (2024).

The management of the ASF will divided into two phases: (i) Theoretical, revisiting some previously discussed conceptual aspects in other courses; (ii) Practical, project development. In the theoretical phase, we will revisit and reinforce fundamental concepts previously discussed, such as types of AI-based systems, minimum viable product (MVP), design thinking principles, and project management basics. This theoretical foundation is crucial for equipping students with the knowledge necessary for effective project execution. In the Practical phase, students will apply their theoretical knowledge to the development of the defined projects. They will be encouraged to work collaboratively, engage in brainstorming sessions, and utilize project management tools to facilitate their workflow. Additionally, the Kanban method will be employed to support the practical aspects of the software factory. Kanban offers a visual management system that helps the team quickly identify and address bottlenecks, ensuring an efficient workflow throughout the project.

To execute ASF, we recommend following a standardized methodology procedure. We have defined three phases for our methodology, which must be conducted during the academic semester: Definition, Development, and Closure, as shown in Figure 7. These phases follow the Action Research cycle (Thiollent, 2005): problem identification, solution planning, implementa-

(A) **Objective/Summary**	
This course aims to equip students with the skills needed to develop artificial intelligence (AI)-based systems, including (1) identifying the characteristics of AI-based systems according to subfields, (2) defining minimum viable product (MVP) functionalities for AI-based systems, (3) specifying requirements based on the characteristics of AI-based systems, (4) managing AI-based systems, (5) modeling and developing AI-based systems, and (6) conducting tests.	
(B) **Themes** (C)	(D) **Learning Outcomes**
(1) Artificial Intelligence: • Types of AI-based systems • Subfields and their characteristics	• Recall the different types of AI-based systems. • Interpret the different types of problems. • Differentiate the characteristics of each type of system.
(2) MVP: • Types of MVP • Market and user discovery • Functionalities	• Recall the different types of MVPs for product development • Understand the current state and desired state of the product to be developed. • Identify the type of MVP considering the problem and the characteristics of AI-based systems. • Identify candidate functionalities for the MVP based on the activities of the proto-personas and characteristics of AI-based systems. • Develop documentation of the functionalities using CASE tool(s)
(3) Requirements: • Specification techniques, models or algorithms using user stories according to the characteristics of AI-based systems • Specification of MVP using user stories	• Structure candidate functionalities into user stories according to the type of problem. • Structure candidate functionalities into user stories according to the characteristics of AI-based systems. • Structure candidate functionalities into user stories considering techniques/algorithms to support the development of AI-based systems. • Structure candidate functionalities into user stories of the MVP • Develop user story documentation using CASE tool(s).
(4) Management: • Product backlog • Prioritization • Deliverables	• Create the product backlog • Prioritize the items in the backlog • Define the deliverables • Manage the development process using CASE tool(s)
(5) Modeling and Development: • Modeling and development considering the characteristics of AI-based systems	• Model the system considering aspects related to data, infrastructure, and techniques/algorithms • Use tools to assist in modeling the AI-based system • Develop the AI-based system • Use tools for project versioning
(6) Testing and Validation: • Test plan • Testing practices	• Design and structure tests considering the characteristics of AI systems • Validate the developed techniques/algorithms Conduct unit tests • Use tools to assist in conducting and recording the tests performed

Figure 6: Academic Software Factory Curriculum using Competency-based Design and Bloom's taxonomy.

tion, monitoring, and evaluation of effectiveness, leading to reflection and adjustment. Each phase comprises a set of stages that students must complete during ASF, which are detailed below.

4.3.1 Definition phase

This phase is composed by six stages: (1) projects definition; (2) team creation; (3) problem, needs and users identification; (4) problem, needs and users identification; (5) requirements specification of the algorithm and solution; and (6) modeling of the algorithm and solution.

Definition phase begins in the first class day with first stage **(Stage 1 - Project definition)**. The projects to can be carried out during the semester is selected based on the number of enrolled students, considering the individual characteristics of the project, such as scope, complexity, development stage, and interdependence between project activities. The projects arise from problems directly related to the day-to-day activities of the University. The selection of a project also considers the professors' competencies, the project's maturity in terms of scope definition, requirements stability, and the availability of external stakeholders.

In the second stage **(Stage 2 - Team creation)**, students fill out a form to identify their profile. Table 4 illustrates sample questions from the profile assessment form. This strategy allows a more appropriate allocation of students to project teams. Each team must be composed of up to 7 students, and some roles will represented by professors from the SE and AI field.

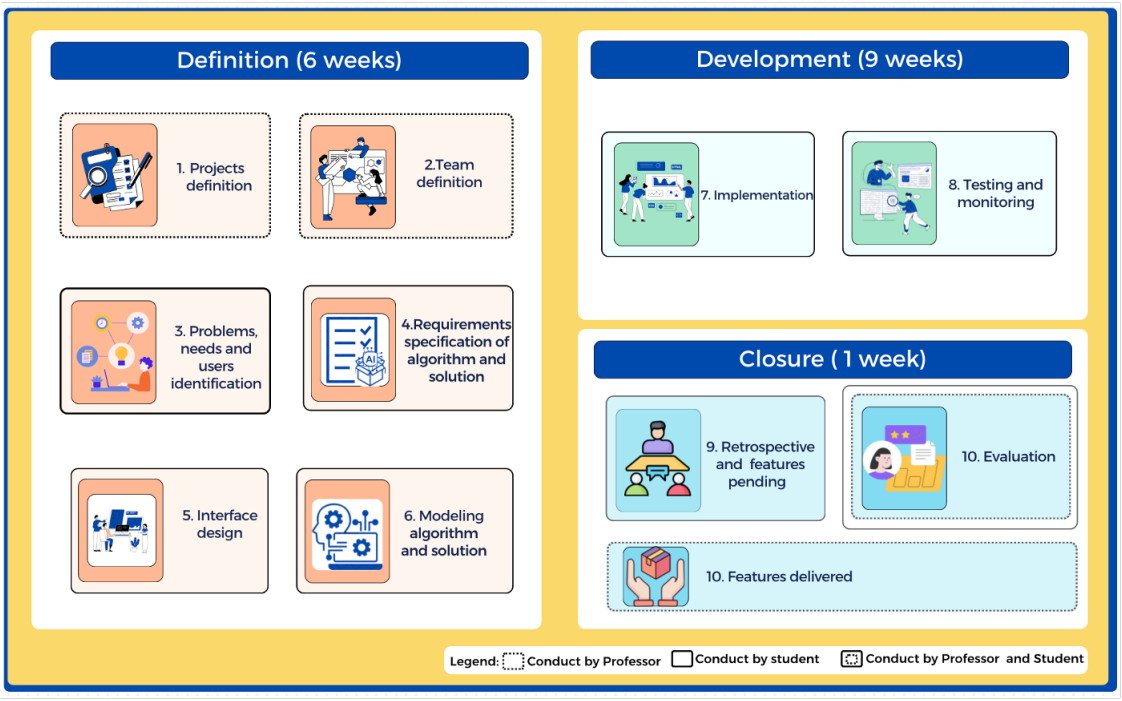

Figure 7: ASF course methodology procedure.

In the third stage **(Stage 3 - Problems, needs and users definition)**, students should analyze the current problems and market niche of the project, focusing on the specifics required for developing AI components. This analysis involves identifying user challenges and collecting high-quality, diverse, and unbiased data essential for training the algorithms. Additionally, it is crucial to conduct a competitive study to identify innovative opportunities in terms of scalability, performance, and algorithm transparency. Finally, students must clearly define the profiles of the end users by creating detailed personas to guide the ethical and effective development of the solution

In the fourth stage **(Stage 4 - Requirements specification of algorithm and solution)**, students dedicate themselves to specifying the project requirements in detail, covering the collected and processed data, the algorithm, and the system as a whole. For the specification of AI-based systems, students must define the algorithm's performance criteria—such as accuracy, robustness, scalability, and interpretability—as well as the ethical and compliance guidelines necessary for the responsible use of AI. They should also develop a robust set of business rules that establish operational guidelines and restrictions, in addition to creating a glossary of terms to standardize communication among all stakeholders. Additionally, the requirements need to be structured as user stories, each accompanied by acceptance criteria, ensuring that the functionalities meet the users' needs.

In the fifth stage **(Stage 5 - Interface Design)**, students develop interface prototypes that emphasize usability, accessibility, and AI transparency for an intuitive and inclusive experience. For AI-based systems, the prototypes must highlight how the algorithms process and present information, facilitating user understanding and trust. In this phase, students convert the collected specifications into sketches and wireframes that outline the system's structure and navigation flow,

Table 4: Examples of questions from the student profile assessment form.

Questions	**Response options**
1. Name	Open-ended
2. Email	Open-ended
3. What role do you want to play in ASF?	PO - Requirements Analyst - UX Designer - Software Architect - Scrum Master - Backend Developer - Frontend Developer - Fullstack Developer - Tester
4. What is your level of seniority in the indicated role?	Junior - full - senior
5. What is your experience time in the indicated role?	less than 1 - 1 to 3 years - 4 to 7 years - 8 to 11 years - 12 to 15 years - over 15 years
6. In which SE areas do you have the most experience?	Business - Requirements - Architecture - Development - Testing - None
7. In which areas of AI have you had any contact?	Intelligent Agents (IA) - Evolutionary Computing (EC) - Pattern Recognition (PR) - None

with a focus on clear visual communication of the AI components and decision-making processes. These drafts are then refined into interactive prototypes using design tools, simulating both functionality and interaction mechanisms that help users understand how the algorithms process and present information.

In the sixth stage **(Stage 6 - Modeling of algorithm and solution)**, students will transform the previously identified requirements into functional and structural models that will serve as a development guide. Initially, they will define and model the algorithm by detailing its logic, processing flow, and the machine learning techniques to be employed, with special attention to data quality and diversity, bias mitigation, and result interpretability. Next, the students will design the integrated solution by defining a modular and scalable architecture that facilitates the continuous integration of AI components. They must also select appropriate design patterns, establish the main modules, and organize communication between them, ensuring a robust and easily maintainable implementation. This process ensures that the solution not only meets performance, scalability, and maintenance requirements but also incorporates best practices and ethical guidelines essential for the development of intelligent systems.

4.3.2 Development phase

This phase is composed by two stages: (7) MVP implementation; and (8) Testing e monitoring. In the seventh stage **(Stage 7 - MVP implementation)**, students must create source code repositories to begin implementing the AI component and the integrated solution. This stage is especially critical for AI-based systems, as it involves defining a modular architecture that facilitates the continuous integration of learning algorithms, the management of training data, and the execution of iterative tests. The team should collaboratively choose the programming language and technologies to be used, taking into account compatibility with AI libraries and frameworks, the scalability of the MVP solution, and ease of maintenance.

In the eighth stage **(Stage 8 - Testing e Monitoring)**, students must develop comprehensive test plans and test cases for AI-based systems, emphasizing the validation of algorithms in terms

of performance, accuracy, robustness, scalability, and ethical compliance. It is essential that the tests include realistic scenarios and exception cases, addressing both individual modules and the complete system integration. This approach ensures the obtainment of interpretable results and the reliability of the solution, in line with the previously established requirements.

4.3.3 Closure phase

The closing phase consolidates the learning, validates the achieved results, and establishes a feedback loop that enriches the development experience in AI-based systems. This phase is composed by t stage: (9) Analysis of results and (10) Evaluation ASF.

In the ninth stage **(Stage 9 - Analysis of Results)**, the team must hold a final retrospective ceremony, where all members share their experiences, discuss the positive aspects, and identify the challenges faced throughout the project. This collaborative reflection is essential for the continuous improvement of the processes and methodologies adopted. At the same time, the team identifies pending functionalities by documenting features they did not implement during the current cycle and prioritizing them for future iterations. Furthermore, this stage involves the formal delivery of the artifacts and the developed system, which includes all technical documentation, source code, prototypes, and other materials produced throughout the project. This delivery officially marks the end of the development cycle and serves as a reference for maintenance and future updates.

The tenth stage **(Stage 10 - Evaluation ASF)** consists of three parts: (i) MVP presentations consist of pitches in which the teams demonstrate the results achieved and reflect on the learning gained during the project; (ii) the final evaluation by the professors, who meticulously analyze each team's results and deliverables, and (iii) the evaluation by the students on the ASF process, providing feedback on the methodology, communication, and overall project experience. Throughout the development, the teacher acts as a facilitator, offering continuous feedback and promptly clarifying doubts. At the end of each stage, summative evaluations are conducted to validate the deliverables and ensure the quality of the work.

At the end of the project, students should evaluate the ASF process by considering aspects such as the methodological approach, communication and collaboration, impact on learning, suitability of resources and tools, and the evaluation format. In addition, the students must share their insights regarding the challenges encountered and provide detailed suggestions for improvement, contributing to the enhancement of future ASF editions.

4.4 Phase 4 - Evaluation of Academic Software Factory Project Design

We conducted an exploratory survey to analyze the perceptions of 10 professors concerning roles envolved, curriculum, and methodology of the ASF in the context of AI-based systems development as a learning environment. The survey was planned following the process proposed by Kasunic (2005) and Kitchenham and Pfleeger (2008) for the effective design of surveys for the software engineering area.

We used the Goal-Question-Metric (GQM) model (Basili & Weiss, 1984) to set out the objectives of the experiment that can be summarized as follows:

*"Analyse **Academic Software Factory Project Design for AI-based systems development** for the purpose of **evaluation** with respect to **roles, curriculum, and methodology** from the point of view of **professors** in the context of **undergraduate software engineering course**."*

For achieving the goal, we seek to investigate the four Research Questions (RQs), presented in Table 5.

Table 5: Research Questions according to Survey Goal.

Research Questions	Description
RQ_1**:** What is the professors' profile?	To answer this RQ, we identify some information such as age group, gender, university location, experience in teaching, area, experience time in area.
RQ_2**:** What are professors' perceptions of the roles required for ASF dedicated to AI-based systems development?	To answer this RQ, we analyzed the professor's perception of the roles and their responsibilities and collaborations in the AI-based system development process.
RQ_3**:** What are professors' perceptions on the ASF curriculum proposal for an undergraduate software engineering course focused on AI-based systems development?	To answer this RQ, we analyzed the professor's perception of the objective, theme, content, and LOs.
RQ_4**:** What are professors' perceptions of the ASF methodology proposal for an undergraduate software engineering course focused on AI-based systems development?	To answer this RQ, we analyzed the professor's perception of the methodology structure in terms of educational objectives, stages and processes, resources and tools adequacy, and evaluation forms.

4.4.1 Target audience and sample identification

The participants were recruited through convenience sampling by sending emails to different social media (WhatsApp and Facebook) and higher education institutions (Thompson, 2012). The target audience of this survey is made up of men and women professors who working with in area of AI and Software Engineering.

Participants received no incentives for taking part in the research (to avoid bias) and were asked to provide informed consent before completing the survey. This approach ensured that their participation was entirely voluntary and fully informed.

4.4.2 Survey instrument design

The survey was conducted using the Google Forms tool for questionnaire design. The survey consists of 20 questions (in Brazilian-Portuguese) that were divided into two groups of questions: *(i)* demographic, socioeconomic and experience data of professors (with six questions); and *(ii)* information of professors' perceptions about software factory themes (with 14 questions).

The first section aims to gather background information about the professors and location, experience in teaching and area. The second section aims to gather insights from professors regarding the roles, curriculum, and methodology essential for implementing the ASF in the undergraduate course.

A 5-point Likert scale (Likert, 1932) was adopted to understand the professors' perceptions about each question, where "1 = Strongly disagree" and "5 = Strongly agree". The questionnaire items were reviewed by experts to ensure content validity.

4.4.3 Survey instrument evaluation

After the first version of the survey has been released, we performed a pilot study aim to analyze instrument validity. According to Kasunic (2005), conducting a pilot survey is fundamental, as it allows detecting possible existing problems.

In this sense, we applied four open questions proposed by Hauck (2012) to evaluate survey content, such as (1) Does the questionnaire contain everything that is expected to meet your goal?; (2) Does the questionnaire contain unnecessary information for the context and purpose of the survey?; (3) Did you adequately understand the questions?; and (4) Are there any errors or inconsistencies in the questionnaire?

We conducted the pilot study with only three professors (who did not participate in the final version of the survey). The pilot participants were recruited through convenience sampling (since they were easy to access and could answer the survey within a week); only one had previous contact with the software factory, while the others did not. The professor's evaluation was positive; only one suggested including at least an open-ended question.

4.4.4 Data Collection and Analysis

After piloting the questionnaire to checking its consistency and legibility, the survey request was available to the professors. The questionnaire was distributed through emails and social networking communities and open for three weeks, from 20th November to 10th December 2024.

To assist the analysis process, two activities were carried out previously Kitchenham and Pfleeger (2008): *(i)* we performed the data validation, checking the consistency and completeness of responses; and *(ii)* we organized the professors' responses according to RQs before data analysis. We use descriptive statistics for the quantitative data interpretation, and discourse analysis and data visualization for the qualitative analysis. For the qualitative analysis, the content summarization technique (Bardin, 2011) was applied to the responses of the subjective questions.

5 Results and Discussion

This section presents the professors' profiles and perceptions of Academic Software Factory Design from the artifacts presented in Figure 5 of Phase 1, Figure 6 of Phase 2, and Figure 7 of Phase 3.

5.1 Professor's profile

Of the 10 professors, four are concentrated in the Southeast region, three in the South, two in the North, and only one in the Northeast. Furthermore, half of them have experience between 10 and 16 years in Software Engineering and another half have experience between 10 and 16 years in AI. Only one professor knows both areas (SE and AI).

We observed that 70% (N = 7) of the professors said that they know of a software factory, while 20% (N = 2) think they might know what a software factory is, and one participant stated they do not know (Table 6). Concerning previous software factory conduct in education, 40% (N = 4 – P6, P7, P8, and P10) of the participants, specialists in the SE area, stated that they had conducted it before, and 60% (N = 6) stated they had not conducted a software factory before for traditional development.

Table 6: Professors profile – Distribution of the 10 participants by region of Brazil, experience time per area (AI - Artificial Intelligence or SE - Software Engineering) and software factory know (Y - Yes, M - Maybe, N - No)..

Participants	Region	Experience Time	Area	Software Factory Know
P1	South	15 years	AI	N
P2	South	15 years	AI	Y
P3	North	12 years	AI	M
P4	Southeast	10 years	AI and SE	Y
P5	Southeast	10 years	AI	M
P6	North	16 years	SE	Y
P7	Northeast	15 years	SE	Y
P8	Southeast	13 years	SE	Y
P9	Southeast	10 years	SE	Y
P10	South	10 years	SE	Y

5.2 ASF Roles required for AI-based system development

The professors indicated the roles that should make up the team in developing AI-based systems (Table 7). All participants indicated UX Designer and Developer roles. The roles of "Scrum Master and "Tester" were not mentioned only by 30% (N = 3 – P1, P3, and P5) of professors who are specialists in the AI field and might or might not know what a software factory is. Additionally, none of the professors indicated the role of Regulatory External as part of the ASF team.

70% of professors (N= 7 – P1, P2, P3, P4, P5, P6, and P7) indicated the need for an "AI Engineer" (AI Eng.) role, and all professor's, specialists in the AI area, mentioned the "AI Team Specific" (AI-ST) role to compose the team because of the characteristics of AI-based systems. In this case, P2 stated: *"All AI-specific team roles should be part of the development process to complement the knowledge of the AI Engineer role."* In this context, two professors mentioned that the AI-Specific Team (AI-ST) and AI Engineer roles can be grouped because they will act according to the problem's nature. One of them suggested that these roles act as consultants.

The professors indicated the inclusion of "Requirements Analyst (RA)" and "Project Manager (PM)" roles in the ASF for AI-based systems. 30% of them (N= 3 – P8, P9, and P10) suggest "RA". P8 states *"I believe that the analyst's role can contribute to the factory because*

Table 7: Roles definition – The professor's perceptions of the roles to compose the team for ASF. The roles include Domain Expert (DE), Product Owner (PO), Software Architect (SA), AI Engineer (AI Eng.), DevOps Analyst (DevOps), UX Designer (UX), Scrum Master (SM), Developer (DEV), Tester, AI Specific Team (AI-ST), End-User (EU), Regulatory External (RE).

Part.	PO	DE	SA	AI Eng	DevOps	UX	SM	DEV	Tester	AI-ST	EU	RE
P1												
P2												
P3												
P4												
P5												
P6												
P7												
P8												
P9												
P10												

they translate the needs into requirement specifications that will guide the product development.", followed by P9, which states *"It helps prevent misunderstandings and rework by ensuring that all system requirements are identified, validated, and managed throughout the development cycle."*. Finally, P10, again, states *"It assists in delivering a product that meets business objectives and user needs."*

50% of professors (N= 5 – P2, P4, P8, P9, and P10) considered the inclusion of "PM" role necessary. P2 *"I think the Project Manager's role can greatly contribute to the factory by aligning the team's efforts with project objectives and managing schedules effectively."*, followed by P4 commented *"A Project Manager helps prevent delays and resource misallocation by ensuring that tasks are properly tracked, deadlines are met, and risks are proactively addressed throughout the development cycle."*. Finally P6 *"This role is crucial in delivering a product that not only meets business objectives but also satisfies stakeholder and team expectations"*.

Based on the professors perception, we grouped the AI Eng./AI-ST roles into "AI Specialists" role and incorporated the roles of "PM" and "RA" in the ASF — functions not included in Souza et al. (2024) work. The "PO", "PM", and "AI Specialists" roles will represented by professors from the SE and AI area, respectively. Figure 8 illustrates the roles defined (Clients, PM, PO, AI Specialist, and the Dev Team - composed of Requirement Analyst, UX Designer, Scrum Master, Devs, and Tester) for ASF and their communication flow-based on the evaluation and suggestions from the professors.

The PM is essential for coordinating activities, managing deadlines, and optimizing resource allocation in AI projects, which demand an iterative approach and effective risk management. Meanwhile, the RA complements the work of the PO by conducting a detailed technical survey of user needs, translating these demands into precise specifications that form the foundation for developing the algorithms and the integrated MVP solution. The inclusion of these roles strengthens the multidisciplinary approach of the ASF, ensuring greater rigor in defining and executing requirements and ultimately enhancing project quality and alignment with market demands.

According to Figure 8, the PO plays a central role in the ASF by serving as a bridge that connects the Client, PM, Development Team, and AI Specialist. **(1)** The interaction starts with

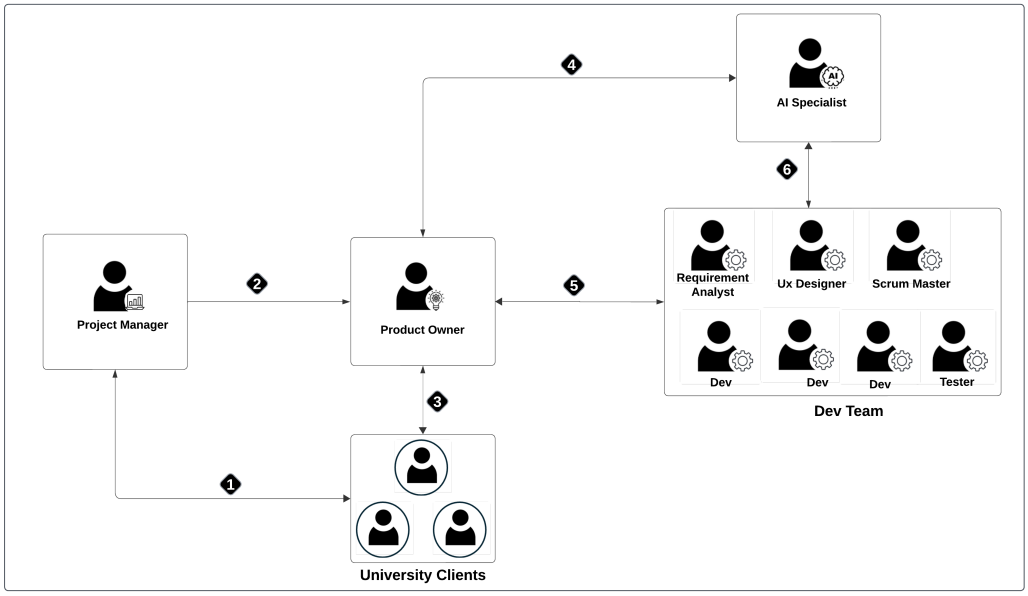

Figure 8: The ASF team roles and their interactions.

the University Clients sharing with the PM and PO the real needs, expectations, and problems that need to be solved. **(2)** PM and PO interact to refine the needs and expectations presented by the Client and to align priorities and resources. The PM oversees the entire process, ensuring delivery within the available deadlines and resources. PO translates the needs of the University Clients into more detailed requirements, prioritizing them in the backlog. **(3)** The PO collaborates with the AI specialist to define and prioritize AI-related functionalities in the backlog. **(4)** The Dev team collaborates closely with the PO to understand the project vision and priorities. The PO provides the team with the product backlog, clarifies requirements, and sets the development goals. Dev Team develops the MVP solution and AI components in iterative cycles. At the end of each iteration, the Dev Team presents deliverables (prototypes, developed features, documentation, etc.), which the PO and PM validate and subsequently by the University Clients. **(5)** The AI Specialist provides expertise and guidance on integrating AI models within the project. They collaborate with the Dev team to ensure that AI components are effectively incorporated and optimized, addressing any technical challenges related to AI.

5.3 ASF Curriculum for AI-based system development

From the Academic Software Factory Design presented Figure 6, the professors indicated six suggestions (S) that are summarized in Table 8. The complete curriculum with the suggested changes is available at the link https://zenodo.org/records/14977568.

5.4 ASF methodology for AI-based system development

This section presents the professors' perception of ASF methodology presented in Section 4.3. For this, we asked each professor to describe the positive and negative aspects of the ASF methodology. We used an approach based on the coding phase of Ground Theory (Stol et al., 2016) to

Table 8: Summary of the professor's suggestions by theme.

ID	Theme	Description
S_1	T2	Add learning outcome *"Identifying proto-personas, what they do, and what they expect"*.
S_2	T2	Remove learning outcome *"Develop documentation of the functionalities using CASE tool(s)"*.
S_3	T3	Remove learning outcome *"Structure candidate functionalities into user stories according to the type of problem"*.
S_4	T4	Add learning outcome *"Estimate potential risks related to the development of AI-based systems" and "Perform practical activities and decision-making in a software project team"*.
S_5	T4	Combine two learning outcomes *"Prioritize the items in the backlog" and "define the deliverables"*.
S_6	T6	Add learning outcome *"Conduct non-functional requirements testing"*.

analyze the responses. To this end, the responses were individually analyzed, and the relevant segments were marked with "codes" (keywords). In this way, it was possible to count the number of occurrences of codes and the number of items in each category to understand which positive and negative aspects the participants repeatedly pointed out.

For the positive aspects, the categories are as follows: **(i) Learning Process** – group codes related to professors' comments on how the methodology can assist in acquiring, retaining, and deepening knowledge, as well as facilitate the understanding of SE and AI topics; **(ii) Professionalism** – group codes related to professors' perceptions of how the ASF methodology can simulate a work environment similar to the professional context of SE and AI; and **(iii) Practice** – group codes related to the positive aspects of the practical nature of ASF. Table 9 presents the positive aspects grouped by categories and the occurrence number. Regarding the positive elements, we identified nine unique codes. The data show that the positive aspects are evenly distributed across all categories, indicating that there can be equally perceived benefits regarding practice, learning process, and professionalism.

Table 9: Summary of the positive aspects of the ASF methodology proposal as perceived by professors.

Category	Positive aspects	Total
Learning Process	Better understanding of SE topics	1
	Better understanding of AI topics into development system	4
	Integration of new concepts into the development process	2
Professionalism	Contact with real work environment	2
	Realistic market experience	2
	Model close to reality	2
Practice	Apply knowledge	1
	Practice knowledge	3
	Solve a real problem	2

For the negative aspects, the code categories are:(i) Lack of Time; (ii) Implementation; (iii) ASF organization; and (iv) Use of the Scrum as software process. Table 10 presents the negative

aspects grouped by categories and the occurrence number. Regarding the negative aspects, we identified nine unique codes. The data show that the negative aspects are more concentrated in the category "ASF Organization" and "Use the software process", with teachers reporting possible challenges mainly in relation to the creation of several templates, including practical examples and lack of prior knowledge. Points for improvement in the methodology, according to teachers, are the inclusion of a leveling method to minimize the lack of previous knowledge, as well as the inclusion of more practical examples and more frequent feedback.

Table 10: Summary of the negative aspects of the ASF methodology proposal as perceived by professors.

Category	Negative aspects	Total
Lack of time	Extensive content	3
	Short time available	4
Implementation	Difficulty in solving the problem	2
	Lack of previous knowledge	6
ASF organization	Formation of groups by the teacher	2
	Lack of feedback on each delivery	4
	Creation of several templates, including practical examples	3
Use the software process	Difficulty in doing something new	3
	Lack of template for specifying requirements considering the characteristics of AI-based systems	5

Based on the professors' analysis, we revised the methodology and detailed the steps included in each phase. Figure 9 details the stages, outlines the tasks, educational goals they aim to achieve, and provides suggestions for tools, models, techniques, technologies, and deliverables to adopt.

It is important to highlight that stages 1 and 2 involve tasks that must be executed by the professors leading the discipline to ensure a clear understanding of the forthcoming process and methodology. Stages 1 and 2 involve tasks that must be executed by the professors leading the discipline to ensure a clear understanding of the forthcoming process and methodology. Stages 3, 4, 5, 6, 7, 8, and 9 are directly related to the AI-based systems development process and to the educational objectives. This process will follow an incremental structure based on deliveries, with deliverables made throughout the course. In particular, stages 3 through 9 are incorporated into the Scrum flow and organized into sprints, ensuring iterative development and continuous improvement. In these stages, the theoretical part will be used to level knowledge related to the topic in question. For Stage 4, a specific requirement document will be included to illustrate the requirements specification considering the characteristics and peculiarities of the AI-based system.

In the context of this ASF, we decided that each stage of a phase corresponds to a deliverable, serving as a milestone for the subsequent phases. For each delivery, the professors will provide feedback to guide and improve the development of the project.

5.5 Limitations

Concerning the limitations of our work, surveys do not provide robust evidence of cause and effect, the small sample size and selection by convenience may not yield a fully generalizable

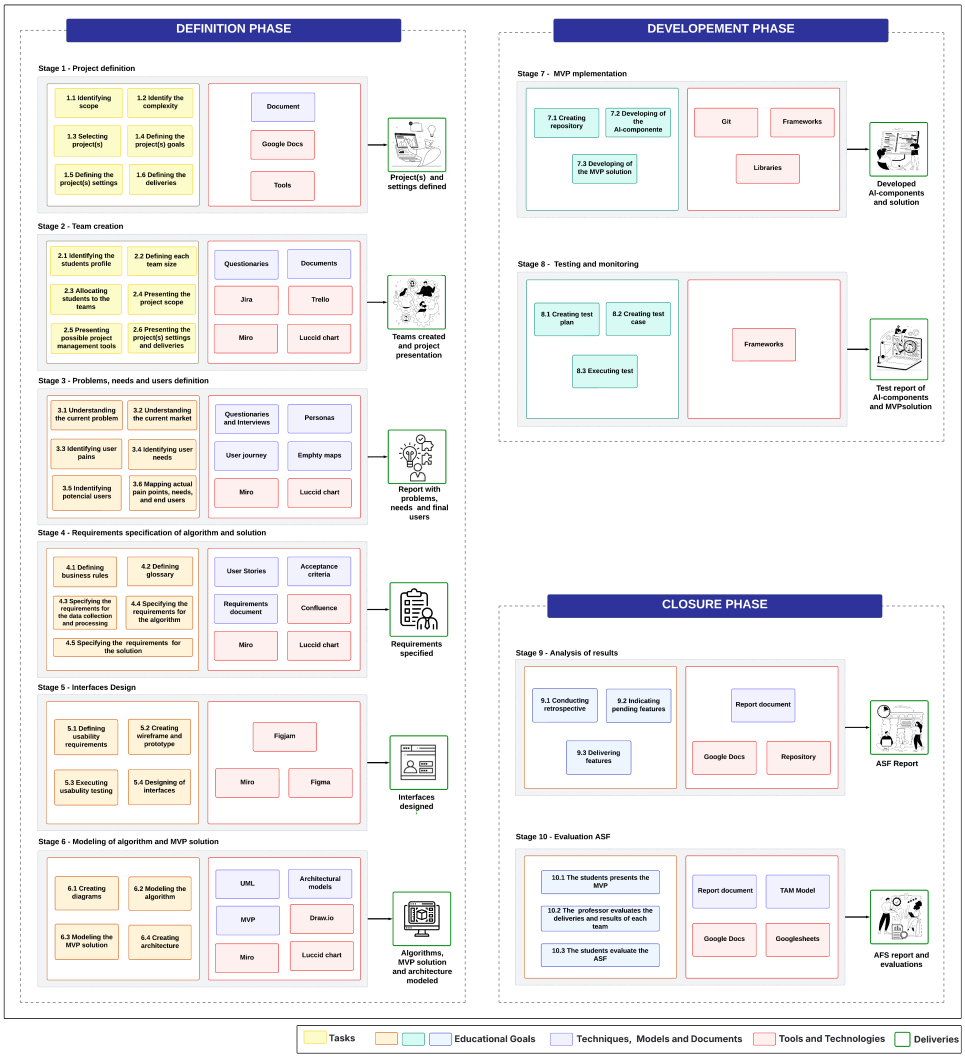

Figure 9: ASF methodology - Stages and their delivers grouped per phase.

results However, given the exploratory nature of our study, these methods remain suitable for field exploration (Kitchenham & Pfleeger, 2008) and (Kasunic, 2005).

Our survey was specifically designed by researchers to assess professors' knowledge and practices in creating an ASF for AI-based systems, addressing a gap in the literature. As our target audience was undergraduate professors, groups, and institutions reached through social media, our data predominantly represented those professional segments. While the data could be used to triangulate potential relationships between practitioners' characteristics and their responses, our study did not focus on generalizing these relationships due to the sample size obtained. Nevertheless, we have made the dataset available for further analysis, allowing other researchers to explore additional insights that may extend beyond the scope of this study.

6 Conclusion

The development of AI-based systems presents unique challenges due to their complexity and the need to integrate diverse technologies. These challenges encompass defining adaptive system requirements, modeling for dynamic environments, selecting suitable AI algorithms, managing large datasets, ensuring system reliability through rigorous testing, and maintaining high-performance computing infrastructure. For software engineering students, the difficulty lies in bridging theoretical knowledge with practical application, as traditional courses often separate AI techniques from software development methodologies.

In response to this gap, this paper proposed the design of an Academic software factory AI-based systems development, integrating competency-based learning, Bloom's Taxonomy, and Scrum methodologies. The ASF design includes a well-defined curriculum, role distribution, and an iterative development methodology to simulate real-world AI-based system development. To validate the proposed ASF, we conducted an expert evaluation with ten software engineering and artificial intelligence professors. The positive feedback highlighted the ASF's potential to enhance educational outcomes and practical training. Key suggestions included refining the curriculum to streamline content, incorporating specialist roles for professors/tutors, and emphasizing the creation and presentation of a minimum viable product by the course's end.

Future work should focus on (i) Validating this proposed team structure in real-world projects to refine role definitions and collaboration practices further, (ii) Validating the methodology in conducting the ASF, (iii) gathering students' perceptions regarding the teaching and learning of AI-based systems development.

References

Ahmad, M. O., Liukkunen, K., & Markkula, J. (2014). Student perceptions and attitudes towards the software factory as a learning environment. *2014 IEEE Global Engineering Education Conference (EDUCON)*, 422–428. https://doi.org/10.1109/EDUCON.2014.6826129 [GS Search].

Anderson, L. W., & Krathwohl, D. R. (Eds.). (2001). *A taxonomy for learning, eaching, and assessing: A revision of Bloom's taxonomy of educational objectives complete edition* (2nd ed.). Longman. [GS Search].

Anthes, G. (2017). Artificial intelligence poised to ride a new wave. *Commun. ACM*, *60*(7), 19–21. https://doi.org/10.1145/3088342 [GS Search].

Ardis, M., Budgen, D., Hislop, G. W., Offutt, J., Sebern, M., & Visser, W. (2015). SE 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. *Computer*, *48*(11), 106–109. https://doi.org/10.1109/MC.2015.345 [GS Search].

Barclay, I., & Abramson, W. (2021). Identifying roles, requirements and responsibilities in trustworthy AI systems. *Adjunct Proceedings of the 2021 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2021 ACM Inter. Symp. on Wearable Computers*, 264–271. https://doi.org/10.1145/3460418.3479344 [GS Search].

Bardin, L. (2011). *Análise de conteúdo*. São Paulo: Edições 70. [GS Search].

Bartneck, C., Lütge, C., Wagner, A., & Welsh, S. (2021). What is AI? In *An introduction to ethics in robotics and ai* (pp. 5–16). Springer International Publishing. https://doi.org/10.1007/978-3-030-51110-4_2 [GS Search].

Basili, V. R., & Weiss, D. M. (1984). A methodology for collecting valid software engineering data. *IEEE Transactions on software engineering*, (6), 728–738. https://doi.org/10.1109/TSE.1984.5010301 [GS Search].

Bishop, C. M. (2006). *Pattern recognition and machine learning (information science and statistics)*. Springer-Verlag. [GS Search].

Borsoi, B. T. (2008, July). *Arquitetura de processo aplicada na integração de fábricas de software* [Doctoral dissertation, Universidade de São Paulo] [Biblioteca Digital de Teses e Dissertações da Universidade de São Paulo]. https://www.teses.usp.br/teses/disponiveis/3/3141/tde-25092008-093143/pt-br.php [GS Search].

Chao, J., & Randles, M. (2009). Agile software factory for student service learning. *2009 22nd Conference on Software Engineering Education and Training*, 34–40. https://doi.org/10.1109/CSEET.2009.26 [GS Search].

Deshpande, A., & Sharp, H. (2022). Responsible AI systems: Who are the stakeholders? *AIES '22: 2022 AAAI/ACM Conference on AI, Ethics, and Society*, 227–236. https://doi.org/10.1145/3514094.3534187 [GS Search].

Digital.ai. (2023). The 17th state of agile report. https://info.digital.ai/rs/981-LQX-968/images/RE-SA-17th-Annual-State-Of-Agile-Report.pdf

Franch, X., Jedlitschka, A., & Martínez-Fernández, S. (2023). A requirements engineering perspective to AI-based systems development: A vision paper. *Requirements Engineering: Foundation for Software Quality*, 223–232. https://doi.org/10.1007/978-3-031-29786-1_15 [GS Search].

Hauck, J. C. R. (2012, October). *Um método de aquisição de conhecimento para customização de modelos de capacidade/maturidade de processos de software* [Doctoral dissertation, Universidade Federal de Santa Catarina]. http://repositorio.ufsc.br/xmlui/handle/123456789/95479 [GS Search].

Jiang, C., Ge, N., & Kuang, L. (2020). AI-enabled next-generation communication networks: Intelligent agent and AI router. *IEEE Wireless Communications*, *27*(6), 129–133. https://doi.org/10.1109/MWC.001.2000100 [GS Search].

Kasunic, M. (2005). *Designing an effective survey*. Pittsburgh: Carnegie Mellon University. [GS Search].

Kaswan, K. S., Dhatterwal, J. S., & Balyan, A. (2022). Intelligent agents based integration of machine learning and case base reasoning system. *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 1477–1481. https://doi.org/10.1109/ICACITE53722.2022.9823890 [GS Search].

Khuat, T. T., Kedziora, D. J., & Gabrys, B. (2023). The roles and modes of human interactions with automated machine learning systems: A critical review and perspectives. *Foundations and Trends® in Human–Computer Interaction*, *17*(3-4), 195–387. https://doi.org/10.1561/1100000091 [GS Search].

Kitchenham, B. A., & Pfleeger, S. L. (2008). Personal opinion surveys. In F. Shull, J. Singer, & D. I. K. Sjøberg (Eds.), *Guide to advanced empirical software engineering* (pp. 63–92). Springer London. https://doi.org/10.1007/978-1-84800-044-5 [GS Search].

Li, J. P., Haq, A. U., Din, S. U., Khan, J., Khan, A., & Saboor, A. (2020). Heart disease identification method using machine learning classification in e-healthcare. *IEEE Access*, *8*, 107562–107582. https://doi.org/10.1109/ACCESS.2020.3001149 [GS Search].

Likert, R. (1932). A technique for the measurement of attitudes. *Journal Archives of Psychology*, *22*(40), 1–55. [GS Search].

Martínez-Fernández, S., Bogner, J., Franch, X., Oriol, M., Siebert, J., Trendowicz, A., Vollmer, A. M., & Wagner, S. (2022). Software engineering for AI-based systems: A survey. *ACM Trans. Softw. Eng. Methodol.*, *31*(2). https://doi.org/10.1145/3487043 [GS Search].

Masapanta-Carrión, S., & Velázquez-Iturbide, J. (2017). Una revisión sistemática del uso de la taxonomía de bloom en la enseñanza de la informática. *Atas do XIX Simposio Internacional de Informatica Educativa e VIII Encontro do CIED–III Encontro Internacional*, 294–297. [GS Search].

Momen, A., Ebrahimi, M., & Hassan, A. M. (2022). Importance and implications of theory of bloom's taxonomy in different fields of education. *International Conference on Emerging Technologies and Intelligent Systems*, 515–525. https://doi.org/10.1007/978-3-031-20429-6_47 [GS Search].

Nahar, N., Zhou, S., Lewis, G., & Kästner, C. (2022). Collaboration challenges in building ml-enabled systems: Communication, documentation, engineering, and process. *Proceedings of the 44th International Conference on Software Engineering*, 413–425. https://doi.org/10.1145/3510003.3510209 [GS Search].

Oliveira, M., Oliveira, S. R. B., & Meira, S. (2017). Condução de uma fábrica de software e o processo de aprendizagem em cursos de graduação em TI: Uma aplicação de um survey sobre a percepção da importância. *Anais do XXVIII Simpósio Brasileiro de Informática na Educação (SBIE)*, 92–101. https://doi.org/10.5753/cbie.sbie.2017.92 [GS Search].

Pei, Z., Liu, L., Wang, C., & Wang, J. (2022). Requirements engineering for machine learning: A review and reflection. *30th International Requirements Engineering Conference Workshops (REW)*, 166–175. https://doi.org/10.1109/REW56159.2022.00039 [GS Search].

Roegiers, X., & Ketele, J.-M. d. (2001). *Une pédagogie de l'intégration: Compétences et intégration des acquis dans l'enseignement*. De Boeck. [GS Search].

Russell, S., & Norvig, P. (2020). *Artificial intelligence: A modern approach* (4th ed.). Pearson.

Santos, A. S., Neves, M., Rodrigues, Y., Oliveira, N. H., Kuhn, D., Santos, G., & Silva, R. A. (2021). Experiência do projeto fábrica de software em um curso de engenharia de software. *Anais da V Escola Regional de Engenharia de Software (ERES)*, 89–98. https://doi.org/10.5753/eres.2021.14869 [GS Search].

Scallon, G. (2017). *Avaliação da aprendizagem numa abordagem por competências*. PUCPRess. [GS Search].

Schwaber, K., & Sutherland, J. (2020). The scrum guide the definitive guide to scrum: The rules of the game [Accessed 10 April 2024.]. https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf [GS Search].

Simon, D. (2013). *Evolutionary optimization algorithms: Biologically-inspired and population-based approaches to computer intelligence*. https://api.semanticscholar.org/CorpusID:60429433 [GS Search].

Souza, F. C. M., Souza, A. C. C., Amorim, B. F., & Cordeiro, T. D. (2024). Towards role definition in agile AI-based system development: Perspectives and reflections. *Proceedings of the*

XXIII Brazilian Symposium on Software Quality, 220–230. https://doi.org/10.1145/3701625.3701661 [GS Search].

Stol, K.-J., Ralph, P., & Fitzgerald, B. (2016). Grounded theory in software engineering research: A critical review and guidelines. *Proceedings of the 38th International Conference on Software Engineering*, 120–131. https://doi.org/10.1145/2884781.2884833 [GS Search].

Thiollent, M. (2005). Insertion of action-research in the context of continued university education. *International Journal of Action Research*, *1*(1), 87–98. [GS Search].

Thompson, S. K. (2012). *Sampling* (3rd ed.). John Wiley Son. [GS Search].

Tvedt, J., Tesoriero, R., & Gary, K. (2001). The software factory: Combining undergraduate computer science and software engineering education. *Proceedings of the 23rd International Conference on Software Engineering. ICSE 2001*, 633–642. https://doi.org/10.1109/ICSE.2001.919137 [GS Search].

Vogelsang, A., & Borg, M. (2019). Requirements engineering for machine learning: Perspectives from data scientists. *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*, 245–251. https://doi.org/10.1109/REW.2019.00050 [GS Search].

Wang, X., Lunesu, I., Rikkila, J., Matta, M., & Abrahamsson, P. (2014). Self-organized learning in software factory: Experiences and lessons learned. In G. Cantone & M. Marchesi (Eds.), *Agile processes in software engineering and extreme programming* (pp. 126–142). Springer International Publishing. https://doi.org/10.1007/978-3-319-06862-6_9 [GS Search].

Wang, Z., Tian, J., & Feng, K. (2022). Optimal allocation of regional water resources based on simulated annealing particle swarm optimization algorithm. *Energy Reports*, *8*, 9119–9126. https://doi.org/10.1016/j.egyr.2022.07.033 [GS Search].

Xie, J., Zhao, Y., Zhu, D., Yan, J., Li, J., Qiao, M., He, G., & Deng, S. (2023). A machine learning-combined flexible sensor for tactile detection and voice recognition. *ACS Applied Materials & Interfaces*, *15*(9), 12551–12559. https://doi.org/10.1021/acsami.2c22287 [GS Search].

Zhou, G., Zhu, Z., & Luo, S. (2022). Location optimization of electric vehicle charging stations: Based on cost model and genetic algorithm. *Energy*, *247*, 123437. https://doi.org/10.1016/j.energy.2022.123437 [GS Search].