People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University of Abbès Laghrour Khenchela
Faculty of Science and Technology
Department of Mathematics and Computer Science
Laboratory of Knowledge Engineering and Computer Security (ICOSI)

جامعة عباس لغرور خنشلة
ABBES LAGHROUR UNIVERSITY KHENCHELA

Doctoral Thesis

in partial fulfillment of the requirements for the degree of Doctor in Computer Science (L/M.D).

Field: Mathematics and Computer Science
Program: Computer Science
Specialty: Software Engineering and Distributed Systems

Descriptive Modeling of Collaborative Practices in a Computer-Based Human Learning Environment

Publicly defended on 07/2025, by:
Sara Ghaoui

Before the jury composed of :

Dr. NESSAH Jamel Associate Professor, University of Khenchela	**Chair**
Prof. Sofiane Mounine Hemam Full Professor, University of Khenchela	**Thesis Supervisor**
Dr. Tarek Djouad Associate Professor , University of Khenchela	**Co-Supervisor**
Dr. HIOUAL Ouided Associate Professor, University of Khenchela	**Examiner**
Dr. BARDOU Dallel Associate Professor, University of Khenchela	**Examiner**
Dr. MARRIR Toufik Associate Professor, University of Oum El Bouaghi	**Examiner**
Dr. BENABBOUD Rohallah Associate Professor, University of Oum El Bouaghi	**Examiner**

People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University of Abbès Laghrour Khenchela
Faculty of Science and Technology
Department of Mathematics and Computer Science
Laboratory of Knowledge Engineering and Computer Security (ICOSI)

Doctoral Thesis

in partial fulfillment of the requirements for the degree of Doctor in Computer Science (L.M.D).
Field: Mathematics and Computer Science
Program: Computer Science
Specialty: Software Engineering and Distributed Systems

Descriptive Modeling of Collaborative Practices in a Computer-Based Human Learning Environment

---

Publicly defended on 07/2025, by:
Sara Ghaoui

Before the jury composed of :

Dr. NESSAH Jamel Associate Professor, University of Khenchela	**Chair**
Prof. Sofiane Mounine Hemam Full Professor, University of Khenchela	**Thesis Supervisor**
Dr. Tarek Djouad Associate Professor , University of Khenchela	**Co-Supervisor**
Dr. HIOUAL Ouided Associate Professor, University of Khenchela	**Examiner**
Dr. BARDOU Dallel Associate Professor, University of Khenchela	**Examiner**
Dr. MARRIR Toufik Associate Professor, University of Oum El Bouaghi	**Examiner**
Dr. BENABBOUD Rohallah Associate Professor, University of Oum El Bouaghi	**Examiner**

Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my sincere gratitude to all those who contributed to the completion of this doctoral thesis and supported me throughout my PhD journey.

First and foremost, I warmly thank my thesis supervisor, **Professor Sofiane Mounine Hemam**, for his continuous support, patience, scientific rigor, and valuable guidance throughout this endeavor. His expertise and constructive feedback have helped me to grow, refine my ideas, and stay on course. He guided me with kindness during challenging times and encouraged me to push beyond my limits.

I would also like to extend my heartfelt thanks to **my colleagues and friends at the ICOSI laboratory**. Their stimulating discussions, moral support, and scientific exchanges have greatly enriched this work.

I would like to express my sincere gratitude to all the jury members for their valuable time, insightful feedback, and contribution to the evaluation of this thesis.

I am deeply thankful to **Dr. NESSAH Jamel**, Chair of the jury, for presiding over the defense and for his thoughtful remarks and encouragement.

My sincere thanks also go to the examiners: **Dr. HIOUAL Ouided**, **Dr. BARDOU Dallel**, and **Dr. BENABBOUD Rohallah**, for their time, thoughtful questions, and valuable suggestions that helped improve the quality of this work. Thank you all for your valuable contribution.

I am deeply grateful to **my husband**, who has supported me throughout this demanding journey. A special thanks to **my parents** for their unconditional love, understanding, and patience. Their presence has been an invaluable support during these years of research.

Thank you to all those who, directly or indirectly, contributed to the achievement of this thesis and helped me overcome difficulties. This thesis is the result of a collective effort, and I am deeply thankful to each one of them.

Abstract

In an increasingly digital educational context, Computer-Based Human Learning Environments (CBHLEs) play a crucial role in supporting both individual and collaborative learning. Collaborative learning supported by these environments offers a major opportunity to strengthen learner engagement and improve the quality of peer interactions. However, to fully leverage this potential, it is essential to understand and assess learning dynamics using reliable indicators derived from interaction trace analysis.

This thesis focuses on the design and computation of collaboration indicators in Computer-Based Human Learning Environments (CBHLEs). It proposes a model-driven architecture (MDA)-based approach to make the indicator computation platform-independent.

Three main contributions are presented:

- First contribution: a model transformation-based approach is proposed to compute collaboration indicators independently of the learning platform.
- Second contribution: a formal model (DECIN Model) is introduced to assist teachers in designing valid collaboration indicators tailored to their observation needs.
- Third contribution: the computation process is automated through the generation of Sequences of transformations, generated by a formal system (DECIN-AGST System).

To support this approach, a platform-independent trace model (T-PIM) is proposed, along with formal constraints expressed in OCL to ensure the validity and consistency of the collected traces. Similarly, a generic model for collaborative indicators (CI-PIM) is defined, supported by constraints that ensure the quality of the generated indicators.

Finally, the automation of transformation sequence generation is achieved through the implementation of the DECIN model in a formal system (DECIN-AGST), whose validity has been verified using the model checking method via the UPPAAL tool, thus ensuring safety, accessibility, and absence of deadlocks.

Keywords

Computer-Based Human Learning Environments (CBHLEs), collaboration, collaboration indicators, model-driven architecture (MDA), trace models, model transformations, OCL constraints, model checking, UPPAAL.

الملخّص

في سياقٍ تعليمي يشهد تزايدًا في الرقمنة، تلعب بيئات التعلم المعتمدة على الحاسوب (CBHLEs) دورًا أساسيًا في دعم التعلم الفردي والتعاوني. ويوفّر التعلم التعاوني ضمن هذه البيئات فرصةً قيّمةً لتعزيز تفاعل المتعلّمين وتحسين جودة التفاعلات بينهم. غير أنّ الاستفادة الكاملة من هذا الإمكان تتطلّب فهمًا دقيقًا لديناميكيات التعلم، وتقييمًا يعتمد على مؤشرات موثوقة مشتقّة من تحليل التفاعلات.

تركّز هذه الأطروحة على تصميم وحساب مؤشرات التعاون في بيئات التعلم المعتمدة على الحاسوب، من خلال اعتماد مقاربة ترتكز على الهندسة الموجّهة بالنماذج (MDA) بهدف فصل حساب المؤشرات عن المنصّة التعليمية المستخدمة.

وتقدّم هذه الأطروحة ثلاث مساهمات رئيسية:

- **المساهمة الأولى** :اقتراح مقاربة تعتمد على تحويل النماذج لحساب مؤشرات التعاون بشكل مستقل عن منصّة التعلم.
- **المساهمة الثانية** :تقديم نموذج يُساعد الأساتذة على تصميم مؤشرات تعاون سليمة تتماشى مع احتياجاتهم في الملاحظة والتحليل.
- **المساهمة الثالثة** :جعل عملية الحساب أوتوماتيكية من خلال توليد تسلسلات تحويل يتم إنشاؤها آليًا باستخدام نظام.(DECIN-AGST)

ولتحقيق هذا الهدف، تم اقتراح نموذج مستقل عن المنصة (T-PIM) تدعمه قيود معبّر عنها بلغة OCL لضمان صحة واتساق البيانات المُجمّعة. كما تم تطوير نموذج عام لمؤشرات التعاون (CI-PIM) مدعوم بقيود تضمن جودة المؤشرات المُنتجة.

وفي الأخير، تم إدراج نموذج DECIN ضمن نظام(DECIN-AGST) ، حيث جرى التحقق من صحته باستخدام تقنية التحقق بالنماذج (Model Checking) عبر أداةUPPAAL ، مما يضمن السلامة، وإمكانية الوصول، وخلوّ النظام من حالات الجمود.

الكلمات المفتاحية:

بيئات التعلم المعتمدة على الحاسوب، التعاون، مؤشرات التعاون، الهندسة الموجّهة بالنماذج(MDA) ، النماذج، تحويل النماذج، قيودOCL ، التحقق بالنماذج، .UPPAAL

General Introduction

Introduction

In the current context of digital education, Computer-Based Human Learning Environments (CBHLEs) play a central role in the training and skills development of learners. These systems not only provide access to a variety of educational resources, they also offer a structured, interactive framework for individual and collaborative learning. Thanks to their advanced functionalities, CBHLEs facilitate knowledge sharing, group problem-solving and the development of cross-disciplinary skills that are indispensable in a constantly evolving world.

Collaborative learning supported by these environments presents a major opportunity to improve learner engagement and the quality of peer-to-peer interactions. However, to take full advantage of the potential of CBHLEs, it is essential to understand and assess learner behavior in real time. This requires the use of reliable and relevant indicators, capable of providing detailed information on learning dynamics, participation and any difficulties encountered by learners.

These indicators, whether related to interaction, performance or collaboration, are essential tools for teachers and tutors. They enable them to better monitor learners' activities, identify sticking points, propose appropriate solutions, and encourage more productive interactions. What's more, these indicators play a key role in personalizing learning paths, facilitating rapid feedback tailored to the specific needs of each learner or group.

Despite their importance, the design and use of indicators in CBHLEs still poses a number of challenges. These include the need to offer tools that are accessible to teachers, the ability to customize indicators according to pedagogical needs, and the integration of approaches that enable effective capitalization and reuse of indicators across different learning platforms.

Definition of key terms used

A Computer-Based Human Learning Environments (CBHLEs)[1] [2] refers to a computer system designed to support learning and teaching processes. These environments use advanced

technologies to provide learners with an interactive, dynamic, and personalized environment. They integrate tools, resources, and functionalities to enhance understanding, engagement, and collaboration in a variety of educational contexts.

Collaborative learning[3] is a pedagogical approach in which learners work together in small groups or teams to achieve common learning goals. This method is based on the idea that interaction between learners promotes the acquisition of knowledge, the development of social skills, and mutual enrichment through shared perspectives and experiences. Unlike cooperative learning, where tasks are often divided between members, collaborative learning places greater emphasis on the joint construction of knowledge.

Collaborative learning aassisted by a CBHLE [7] refers to a specific form of learning where digital tools and technology platforms are used to facilitate interaction, coordination, and co-construction of knowledge between learners. In this context, CEHLs serve as technological intermediaries supporting collaborative activities through functionalities such as communication, resource sharing, task management, and analysis of learner interactions.

A common example is the use of the Moodle learning platform, where learners collaborate toward the common goal of learning.

The functionalities offered by these environments make it possible to coordinate efforts (e.g., task management), share resources (e.g., file repository), and communicate effectively (e.g., forums or videoconferences).

In this type of learning, collaboration enables learners to share their points of view, explain their ideas, confront their opinions, and co-construct solutions. This process stimulates cognitive mechanisms such as explanation, argumentation, and problem-solving, where each learner actively builds their knowledge through exchanges and collaboration with group members.

In collaborative learning supported by a CBHLE, the teacher plays a key role as facilitator, activity designer, and mediator. They guide learners in collaboration, design adapted scenarios, and use indicators to monitor and evaluate interactions. By intervening to resolve blockages and encourage reflexivity, the teacher fosters an effective collective dynamic while supporting the acquisition of knowledge and skills.

In the context of CBHLEs, the tutor's role can take on a new dimension compared with traditional learning. Unlike traditional learning, where the teacher is generally the only actor responsible for providing support, the introduction of the tutor's role in CEHLs can enable collaborative learning to be better structured and supported [8].

The teacher is more involved in the planning and organization of collaborative learning, while the tutor focuses on the daily support, engagement, and development of learners. Both roles are essential to ensure a successful collaborative learning experience.

If necessary, the teacher can take on the role of tutor in a collaborative learning assisted by a CBHLE. This dual role of teacher and tutor enriches the students' learning experience, combining the benefits of direct instruction and interactive support in a collaborative setting.

In a collaborative learning situation, the roles of teacher and tutor are complementary but distinct.

In collaborative learning supported by a Computer-Based Human Learning Environment (CBHLE), it is important to evaluate how students work together. Being in a group does not always mean real collaboration is happening. Some students may be very active, while others stay quiet or do not contribute much. By evaluating collaboration, we can check if learners are really sharing ideas, helping each other, and building knowledge as a team. It also helps teachers or the system give support to students who need help. Feedback from the evaluation helps students understand how they worked with others and how they can improve. CBHLE systems can also use this information to adapt the activity or group structure to make collaboration better. Evaluation makes group work fairer, especially when grades are involved, because it shows what each student actually did. It also helps students develop teamwork skills like communication and cooperation, which are important for their future. For teachers and researchers, collaboration evaluation provides useful information to improve teaching strategies, design better learning activities, and understand how students learn together with digital tools.

Evaluation of online collaborative work remains one of the current limitations of collaborative distance learning [9]. It involves not only assessing knowledge acquisition at an individual level, but also measuring the participation rate of each learner or group of learners in collaborative work. The construction of an evaluation grid in a CEHL is based on verifying knowledge acquisition and observing behavior, as well as the quality of the work methodology [10]. These observations are used to define an indicator [11]. An indicator is a variable in the mathematical sense that provides information based on observation.

In a Computer-Based Human Learning Environment, an indicator is a measure or criterion used to evaluate the activity, performance, or collaboration of learners within the system.

A collaboration indicator in a Computer-Based Human Learning Environment is a criterion specifically designed to measure and evaluate the quality and effectiveness of collaboration between learners within the system, enabling teachers and tutors to monitor the progress of collaborative work, identify any blockages, and propose interventions to improve the effectiveness and quality of collaboration.

When using interactive CBHLEs, numerous actions and events are generated (connection, disconnection, chat, reading, etc.). These can serve as valuable sources of information, known as “traces”.

In recent years, much of the research into e-learning environments has focused on the evaluation of learner activities and the calculation of indicators.

So, in the context of Computer-Based Human Learning Environments (CBHLEs), the evaluation of collaboration is based on indicators calculated from learning traces. Calculating these indicators to support collaboration and collaborative learning is a very active

research field. However, existing work shows several limitations. First, the design and calculation of many collaboration indicators are often done in an ad hoc way, specific to a platform or a given context, which limits their reuse and generalization. Next, the current tools and methods are based on complex logic and languages, making them inaccessible to teachers or tutors, even though they are the main users of these indicators. In addition, the closed nature of some systems forces users to rely only on predefined indicators, without the possibility to adapt or customize them for specific needs. Moreover, the available tools are often designed for specific environments, such as Moodle or other proprietary systems, which limits their flexibility and adaptability to other learning contexts. Finally, despite efforts to automate the design of indicators, current approaches struggle to ensure complete formalization, effective reuse, and long-term scalability. Faced with these challenges, it becomes essential to develop a method that is platform-independent, accessible to teachers, and allows the design, customization, and calculation of collaboration indicators while ensuring their automation, reuse, and adaptability to various collaborative learning contexts.

The Problem of the Thesis

Computer-Based Human Learning Environments (CBHLEs) have deeply changed the way we learn and teach, by introducing new digital tools to facilitate exchanges and group work. These systems no longer just deliver content: they also allow learners to interact, collaborate, and build knowledge together. However, despite these advances, several problems remain, especially regarding learner motivation and the quality of exchanges between them.

Many studies show that learner isolation is one of the main causes of dropout in online training. This phenomenon can be linked to several factors: lack of time, organizational difficulties, feeling of loneliness, or lack of support. Recent studies [4, 5, 6] highlight the importance of the "sense of community": feeling like a member of a group is essential to maintain engagement and encourage success. Social interactions are therefore not secondary: they are at the heart of the online learning process.

Yet, in current systems, teachers rarely have tools that would allow them to quickly detect learners who are struggling or becoming isolated. Without precise information on collaborative activity, it becomes difficult to act in time to help these learners. That is why it is important to be able to measure and analyze collaboration using precise indicators.

Setting up such indicators would allow teachers to track the evolution of interactions and intervene more effectively to support students in difficulty. But their design presents several challenges: it must allow teachers — who are not necessarily IT specialists — to easily define the behaviors they want to observe, while ensuring that the indicators can be adapted to different tools and educational contexts.

Thus, the problem we address is the following: How to design, model, and automatically compute collaboration indicators in CBHLEs, independently of the platform used, while

making their creation accessible to teachers or tutors?

To answer this question, it is necessary to propose a method based on the modeling of activity traces (that is, the recordings of actions performed by learners in digital systems). This method must be flexible enough to adapt to various learning contexts, while being simple to use for teachers.

We assume that by using a model-driven approach (Model-Driven Architecture, MDA) and by representing traces in a structured form (m-traces), it is possible to meet these needs. Thanks to this approach, teachers can express what they want to observe, without having to worry about the complex technical aspects related to data collection and processing.

So, our main problem is how to develop a method that is platform-independent and accessible to teachers, which allows them to design, customize, and calculate collaboration indicators in Computer-Based Human Learning Environments (CBHLEs), while ensuring their automation, reuse, and adaptability to different collaborative learning contexts?

0.1 Research Objectives

Our research pursues several complementary objectives.

First, we want to propose a general method that allows the creation of collaboration indicators independently of the technical platform (Moodle, forum, etc.). This method is based on two main models:

T-PIM: a generic model to describe activity traces,

CI-PIM: a generic model to describe collaboration indicators.

These models serve as a basis to then produce adapted versions for the specific tools used in training.

Next, we aim to make it easier for teachers to create indicators themselves. For this, we are developing a model called DCIN Model (Design Collaboration Indicators Model), which allows them to visually build indicators using simple elements, such as “steps” and “temporal links” between these steps.

Finally, we want to automate the entire process. A system called DCIN-AGST will automatically transform the needs expressed by teachers into collaboration indicators.

Work done

Our proposal focuses on the design and calculation of collaboration indicators in CBHLEs. Thus, our contribution can be summarized as follows:

- **First contribution**: It concerns the calculation of collaboration indicators in CBHLEs, regardless of the platform used. To do this, we propose applying a model transformation approach and a process based on Model-Driven Architecture (MDA).

- **Second contribution**: It focuses on the design of collaboration indicators in CBHLEs. For this purpose, we propose a formal model that helps design valid and relevant collaboration indicators, adapted to teachers' observation needs.

- **Third contribution**: Based on the application development process proposed by the Model-Driven Architecture, the calculation of collaboration indicators is seen as a model transformation process. In this framework, the trace model goes through a series of successive transformations to result in the collaboration indicator model. To meet the need for automation of the indicator calculation, we propose a solution to automatically generate transformation sequences. This contribution focuses on the automatic generation of these sequences, which will be applied in the m-trace-based system to produce the collaboration indicator model.

Our work aims to define an approach based on MDA for calculating collaboration indicators, regardless of the platform used. In this approach, we proposed a platform-independent trace model (T-PIM) that serves as a meta-model for collaborative and non-collaborative traces in CBHLEs.

To avoid errors due to an incomplete or incorrect proposal of the T-PSM (Trace Platform Specific Model), we introduced a set of formal constraints expressed in OCL (Object Constraint Language). These constraints are intended to frame the temporal and structural relationships of the model elements, thus ensuring the validity of the generated traces. They include rules about the mandatory presence of obsels, the validity of dates, and the categorization of elements. By enforcing these rules in the model, we make sure that only valid and coherent data are collected.

We also proposed a platform-independent model for collaboration indicators, CI-PIM, which represents a meta-model of collaboration indicators in CBHLEs. The calculation of collaboration indicators using our approach can be seen as a sequence of transformations applied to a trace model "T-PSM", which compliant with our proposed T-PIM, to obtain the collaboration indicator model "CI-PSM", which compliant with our CI-PIM. These transformations must be written in a specific transformation language (transformation meta-model), which allows the steps of the process to be formalized. In our case, we chose a simplified version of this language, which will then be translated depending on the transformation language used in the trace-based system. To ensure the transformations are performed correctly and the results meet expectations, our collaboration indicator model (CI-PIM) is accompanied by OCL (Object Constraint Language) constraints.

To ensure valid and simple design of collaboration indicators, we proposed a formal model in the form of timed automata, the 'DECIN model' (DEsigning Collaboration INdicator Model), which simplifies the modeling process by guiding the designer through the key steps needed to create a valid and suitable collaboration indicator.

To automate the calculation of collaboration indicators, our DECIN model was implemented in a formal system for automatic generation of transformation sequences (DECIN-AGST system). Once the designer finishes the design, a transformation sequence will be automatically generated to be applied to the trace model T-PSM, in order to obtain the collaboration indicator model CI-PSM.

To ensure the validity and safety of the DECIN-AGST system, we verified with the formal verification method 'model checking' using the verification tool **UPPAAL** essential properties such as safety, reachability, deadlock-freeness, and liveness.

Thesis Outline

The thesis is organized as follows: The thesis is divided into six chapters in addition to the introduction and the general conclusion.

Introduction : The introduction outlines the context, motivation for this research, objectives, as well as the problem statement and contributions.

Chapitre I : This chapter is dedicated to general information on Computer-Based Human Learning Environments (CBHLEs) and collaborative learning supported by CBHLEs. A theoretical study of these two domains is undertaken to lay the conceptual foundations necessary for the rest of this work. In parallel, this chapter explores the different dimensions of collaborative learning supported by CBHLEs. We analyze the specificities of this type of learning, highlighting its educational benefits-such as improved motivation, creativity, and quality of interactions-as well as its limitations, particularly in terms of conflict management, evaluation of collaboration, and group heterogeneity. Finally, a classification of collaboration approaches in CBHLEs-supported learning is proposed, highlighting the essential facets for analyzing, structuring, and assessing collaborative interactions in these digital environments.

Chapitre II : The Chapter 2 presents the theoretical foundations and related work directly connected to the research. Its goal is to position the contribution within the state of the art concerning trace analysis in CBHLEs, trace modeling, and the computation of interaction and collaboration indicators.

The chapter begins by introducing essential notions related to digital traces: their collection, challenges, and the limitations of raw log files. This analysis justifies the transition to modeled traces, which are more structured and better suited to educational analysis. Modeled-Trace-Based Systems (MTBS) are then presented, with emphasis on the fundamental concepts, model-driven engineering (MDE) approaches applied to traces, and expected system features.

The second part of the chapter focuses on the languages and methods for computing indicators from modeled traces, with particular emphasis on interaction and collaboration indicators. It defines concepts related to evaluating collaboration in online learning environments and outlines existing indicator templates and languages used for their formalization.

The third part reviews previous work closely related to the addressed issue. Two main categories of approaches are identified:

- Ad hoc approaches for computing collaboration indicators, often specific to a given context;
- More generic methods and tools based on modeled formalisms, particularly those derived from model-driven engineering.

The chapter concludes with a critical discussion and scientific positioning of the contribution in relation to existing work, highlighting the limitations identified in the literature and the specific contributions of the proposed approach.

Chapitre III : The Chapter 3 presents the methodological approach adopted to address the thesis problem. It starts with a detailed presentation of the main research problem. Next, the research objectives are outlined, followed by the presentation of the chosen methodological approach to solve the problem.

Chapitre IV : The Chapter 4 presents the proposed MDA-based approach to design a generic framework for collaboration analysis in CBHLEs. This chapter details the concrete implementation of the approach by presenting the developed models, the transformation process used, and the adopted transformation contracts.

Chapitre V: This Chapter is dedicated to presenting a formal system for the design and computation of collaboration indicators in CBHLEs. This chapter introduces the DCIN model in the form of a timed automaton, aiming to simplify the process of modeling collaboration indicators by guiding designers through the necessary steps to create valid indicators adapted to the specific needs of learning environments. The implementation of the DCIN model in a formal system, DECIN-AGST, automates the generation of transformation sequences, which are then applied to the T-PSM trace model to produce the CI-PSM collaboration indicator model.

Chapitre VI: This Chapter focuses on testing the system based on the DCIN model. It presents the practical application and validation of the proposed system by detailing the implementation steps in specific use cases. The main objective is to verify the validity and robustness of the DCIN model in real collaboration analysis scenarios by applying the concepts and methodologies developed throughout the thesis. It also includes a

formal verification phase to ensure the consistency, compliance, and reliability of the DCIN-AGST formal system in computing collaboration indicators.

Conclusion and Future Work : Finally, the thesis concludes with a general conclusion summarizing the key points of the work presented, along with suggestions for future improvements to this research.

Part I

State of the Art

Chapter 1

Collaborative Learning Supported by Computer-Based Human Learning Environments

1.1 Introduction

Computer-Based Human Learning Environments (CBHLEs), by integrating digital tools, bring a new dimension to collaborative learning. They help overcome some limitations of traditional learning while offering personalized support. Collaborative learning supported by CBHLEs is based on continuous interaction between learners, tutors, and technologies, with notable advantages such as improved motivation, creativity, and quality of interactions. However, this type of learning also faces challenges, especially concerning conflict management, learner assessment, and group heterogeneity.

In this context, this chapter explores the different aspects of collaborative learning supported by CBHLEs. We will start by defining what a CBHLE is, by tracing its history and evolution. Then, we will discuss the specific features of collaborative learning, highlighting its benefits and limitations. Finally, a classification of collaboration approaches in learning supported by CBHLEs will be proposed, emphasizing key aspects that help analyze and structure collaborative interactions in these environments.

1.2 Computer-Based Human Learning Environment

1.2.1 What is a CBHLE?

A Computer-Based Human Learning Environment is a computer environment that integrates human agents (learner, teacher, tutor) and artificial agents (software agents, which can also

take on different roles). It offers them situations of interaction, either locally or through computer networks, as well as access to resources (human and/or media-based), again either locally or distributed. It represents the combination of a didactic intention and a computer environment, with the main goal being human learning. It is designed to be used by learners in a learning situation. Learners are usually placed in an active situation of knowledge acquisition while being more or less guided by the system at different levels[25].

1.2.2 History of CBHLEs

Research on the use of computer tools for human learning is not new. Many researchers have been interested in this field since the middle of the 20th century. In this section, the goal is not to provide an exhaustive history of computer applications for educational purposes. Our objective is simply to give an overview of the main stages that have marked the history of computing in education. Readers interested in more details can refer to [26], which traces the evolution of ideas and major research trends in this field.

CBHLEs have gone through different phases and approaches, with the main goal of designing computer systems whose use promotes learning in their users [28].

The teaching machine developed by Skinner in 1958 is a rectangular box containing a paper roll. The student controls the movement of the tape using a knob on the front. Questions appear one by one in a window. The student writes their answer in a blank space reserved for that purpose, then turns the knob. This allows them to see the correct answer. They compare their result with the given answer, and if it is correct, they activate a lever that, by perforating the program sheet, records correct answers. Then they move on to the next question. In Skinner's machine, teaching is provided through immediate correction. But it is mainly a self-evaluation machine, not a knowledge support tool as we understand it today. The machine also offers linear teaching: the student's answer does not affect the next question. For Skinner (1968), "the student must compose their own answer, rather than choose from multiple options, as in multiple-choice devices." He believed that multiple-choice questions lead to errors that students would not make without those suggestions. Also, in Skinner's machine, the student self-corrects by comparing their answer to the program's [30].

The arrival of the computer gave birth to CAI (Computer-Assisted Instruction) systems [30]. These systems aimed to use artificial intelligence techniques to create more flexible, interactive, responsive systems, better adapted to their users, commonly referred to as "learners", to motivate and help them learn.

The 1980s were marked by a deeper use of Artificial Intelligence (AI) techniques in learning systems. Intelligent Tutoring Systems (ITS) marked the transition from CAI to ICAI, then called "Intelligently Computer-Assisted Instruction", and later "Interactive Computer-Based Learning Environments". This second version emphasized learning, based on a constructivist view where the learner builds knowledge by interacting with their environment in

a didactic sense, replacing the vision of knowledge transfer from teacher to learner. This evolution also supported the use of the term “learning environment” rather than just “computer system” [26].

The approaches presented so far focused on the teaching process, where computer systems mainly tried to play the role of the teacher.

A second approach developed in parallel, in which the computer was seen as a way to enable discovery and exploratory learning by giving control to the learner. This movement is based on constructivist learning theories [31, 32].

Microworlds [32] represent environments where learners can experiment with their ideas and observe the consequences. The idea is to create environments where learners interact with simplified objects to manipulate abstract concepts and build their own knowledge.

The main drawback of these open learning environments is the lack of support for learners. If a learner faces difficulties in reaching their goal, the system cannot assist them.

Intelligent Tutoring Systems (ITS) [28] are computer-based learning environments that aim to simulate the behavior of a human tutor, including their expertise in teaching a subject. These systems can guide the learner through a task and provide relevant feedback on their actions.

It is also worth noting that other aspects have been integrated into these systems, such as simulation [35]. These systems are designed to study phenomena (for example, in physics) or to learn tasks in controlled situations (for example, diagnosing malfunctions, operating industrial processes or vehicles...). These approaches targeted various learning situations and activities, all aiming to design systems with interactive and adaptive capabilities to support learning, either under teacher supervision or more autonomously.

In recent years, we have seen a shift from ICAI to environments that support communication and interaction between distributed machines and humans. In this sense, a new term has recently emerged to describe learning environments with computers: Computer-Based Human Learning Environments (CBHLEs). This acronym reflects recent advances in telecommunication technologies and the shift toward distance learning. The term CBHLE includes both ICAI systems as described earlier and ICAI systems more focused on remote learning.

The move toward considering social interaction and collaboration among learners, by focusing on the notion of exchange and communication, has led to a focus on collaborative learning [36]. From the early 1990s to the present day, with the advent of the Internet, there has been a strong desire to design CBHLEs that work on the Web, such as distance learning platforms.

Educational hypermedia systems and adaptive hypermedia systems represent a trend inspired by Ted Nelson’s idea of a network of documents, adapting the presentation of learning content based on the learner’s pedagogical needs, level of understanding of the subject, and learning style.

In conclusion, the evolution of computer tools for human learning reflects a dynamic and ongoing transformation in educational technology. From Skinner's early teaching machines focused on self-evaluation, to the rise of Computer-Assisted Instruction (CAI) and the development of Intelligent Tutoring Systems (ITS), each stage has contributed to a deeper understanding of how technology can support and enhance learning. The integration of artificial intelligence and constructivist theories led to more interactive and learner-centered environments, such as microworlds and adaptive learning systems. Over time, the focus expanded from individualized instruction to include social interaction, collaboration, and distributed learning, giving rise to the concept of Computer-Based Human Learning Environments (CBHLEs). These environments now combine adaptability, interactivity, and connectivity, aligning with the diverse and evolving needs of learners in the digital age.

1.2.3 Classes of CBHLEs

The first class involves CBHLEs that have largely inherited research on intelligent tutors and microworld systems [28, 32]. These are knowledge-based systems, where interaction occurs with a single learner, developing assistance and problem-solving functions. These are CBHLEs that guarantee individual learning where the learner is more or less supported in their learning activity, which implies significant autonomy from the learner.

The second class focuses on distance learning devices that support learning based on learning scenarios [37]. More specifically, this class of CBHLE is made up of systems for self-learning at a distance, which can alternate between individual and collective sequences, synchronous or asynchronous [38]. They allow the creation and updating of new learning situations that can complement traditional teaching, offering rich, interactive applications and enabling collaborative computer-assisted learning, along with individualized training paths adapted to each learner's needs [39]. These systems are based on distance learning platforms (Learning Management Systems), which can manage the learning process [40]. These CBHLEs allow for the online management of content, personalized learning paths, synchronous and asynchronous collaborative applications, administrative tracking, and management of classroom training sessions, including schedules and tutor organization. This requires additional human and material resources, as well as adequate organizational and working methods.

The major difference between these two classes of CBHLEs is that the first class features a type of computer system that can manage and automatically plan the learning content it offers to an isolated learner. The knowledge offered by the content proposed by the system is the only reference for the learner. The second class consists of a set of actors (teachers, tutors, learners, etc.) who interact through a platform [41]. This platform allows the teacher/tutor to plan learning paths for an individual or group of learners. The learners can collaborate with the tutor or with their peers, either synchronously or asynchronously. The knowledge

here has a social status, no longer an individual one [31].

1.3 Collaborative Learning Assisted by CBHLE

1.3.1 Collaborative Learning

We take here the definition of Henri and Lundgren-Cayrol [7] which retains several principles taken from learning theories and values associated with constructivism. "Collaborative learning is an active approach by which the learner works on building their knowledge. The instructor plays the role of a facilitator of learning while the group participates as a source of information, a motivation agent, a means of mutual help and support, and as a privileged place of interaction for the collective construction of knowledge." The collaborative approach combines the individual and reflective nature of learning as well as its social nature by linking it to group interactions [31]. In fact, the collaborative approach couples two approaches: that of the learner and that of the group [43]. The learner commits to working with the members of the group with a view to achieving the common goal while taking into account their interests and personal objectives. They collaborate within the framework of group interactions by sharing their discoveries. Exchanges with the group and the completion of a collective task allow them to share their discoveries, to negotiate the meaning to give to their work, and to validate their newly constructed knowledge [36]. In this approach, they show autonomy and take responsibility for their learning just as they feel responsible for achieving the goal they share with everyone. The group is a catalyst for learning. By setting a goal, by committing to the collective completion of a task, it allows each learner to confront others and it nourishes the learning of each one [44].

1.3.2 Towards Collaborative Learning Assisted by CBHLE

Many studies conducted on the first class of CBHLE still show the weaknesses of this type, in which the poverty of the human and emotional dimension is a source of dropout and failure [45, 46]. The learner may feel isolated, "abandoned" when using this type of CBHLE.

The deployment of Information and Communication Technologies (ICT) in distance learning platforms has introduced major changes in terms of pedagogy [47]. Initially and from an exclusively technical point of view, the communication tools included in CBHLEs have largely gone beyond the stage of "communication instrument". They aim to reverse pedagogical practices and open the way towards a "collective construction of knowledge" [7]. The range of collaborative tools associated with CBHLE, such as forum, wiki, blog, etc., has encouraged the emergence of collective learning [48].

The particular context of most distance learning programs assisted by a CBHLE requires certain specificities such as: the physical distance between learners, the absence or limited number of meetings between learners and/or with their teaching teams, the diversity of learn-

ers from a social, personal situation, and age point of view. Therefore, this context requires more connections between the actors of the system, more emotional comfort, relationship building, knowledge of others, as in any other face-to-face training.

Many studies and analyses show today the importance of communication mediated by the collaboration tools offered by the CBHLE [49]. Some surveys indeed show that this type of collaboration has characteristics similar to face-to-face communication and allows learners to find "emotional or social support through gestures of empathy and sharing shown by their interlocutors" [50]. It thus appears that this type of group communication, within a CBHLE, constitutes a "social and symbolic environment in which participants can develop a sense of belonging to the group and build a collective identity there, whether community-based or social" [27].

A collaborative learning situation assisted by a computer-based human learning environment encourages the desire to exchange, communicate, share, and also participation and collaboration, where belonging to a group or a community in general also provides support, mutual help, or simply moral comfort [51].

1.4 The Importance of Collaboration in the Learning Process

Collaboration in learning is essential to enrich the educational experience and optimize cognitive and social processes. It is based on the idea that interactions between learners allow for an exchange of perspectives, co-construction of knowledge, and the development of essential social and intellectual skills.

- It promotes the co-construction of knowledge:
  Collaboration allows learners to share their views, explain their ideas, confront their opinions, and co-construct solutions. This process stimulates cognitive mechanisms such as explanation, argumentation, and problem-solving. According to Vygotsky (1978)[31], learning is a social process, and collaborative interactions allow individuals to reach their zone of proximal development (ZPD), a space where they perform tasks with the help of peers or a mentor.

- It develops social skills:
  Collaboration enables learners to develop social skills such as communication, active listening, conflict management, and teamwork. These skills are crucial not only in the educational setting but also in professional life. Group work also promotes individual and collective responsibility, as each member must contribute to the common success[52].

- It improves motivation and engagement:
  Working collaboratively strengthens learner engagement, as they feel supported by their

peers and have a common goal to achieve. This can reduce the feeling of isolation, especially in digital or distance learning contexts. According to **Slavin (1995)**[44], cooperative learning methods increase learners' intrinsic motivation by emphasizing mutual help and collective success.

- It stimulates critical and creative thinking:
  Group discussions allow ideas to be confronted and different perspectives to be analyzed. This process stimulates critical thinking by encouraging learners to justify their opinions and evaluate those of others. The diversity of viewpoints also fosters creativity, as it allows for exploring varied solutions to a given problem[53].

- It prepares learners for the professional world:
  In a professional context, collaboration is a key skill. By promoting collaboration early in the educational environment, learners are better prepared to work in multidisciplinary teams and adapt to complex environments. Practices such as collaborative problem-solving and collective decision-making are valuable simulations of the real workplace[54].

- It helps reduce inequalities in learning:
  Collaboration allows learners to support each other, with more advanced learners helping those who face difficulties. This creates a peer learning dynamic that can reduce performance gaps within a group[36].

1.5 Actors in a Collaborative Learning Situation

In a collaborative learning situation supported by a CBHLE, several actors are involved and fulfill complementary roles. Each of these roles contributes to the setup, progress, and evaluation of the learning process. These actors can be human or technical, and their coordination is essential to promote effective collaboration. The main identified roles are: teacher, tutor, learner, IT designer, and administrator. We present below the specific responsibilities of each.

1.5.1 Role of the teacher

The teacher plays a key role in designing and planning collaborative learning [36]. They are responsible for:

- **Setting up the learning environment:** The teacher selects or creates appropriate tools and platforms to foster collaboration (such as forums, shared workspaces, etc.)[53].

- **Pedagogical guidance:** They define learning objectives, create collaborative activities, and structure tasks to encourage interaction between learners [44].

- **Overall supervision:** They ensure the proper implementation of collaborative learning by monitoring group progress and intervening when necessary to adjust the process [54].
- **Global assessment:** The teacher evaluates the group outcomes, but also the overall effectiveness of the collaborative process using collaboration indicators [55].

1.5.2 Role of the tutor

The tutor, usually closer to the learners, acts as a facilitator and guide [56]. He is responsible for:

- **Individual support:** helping learners in their collaborative interactions, answer specific questions, and guide them in solving problems or achieving learning objectives.
- **Encouraging participation:** helpingIn collaborative learning supported by a Computer-Based Human Learning Environment (CBHLE), it is important to evaluate how students work together. Being in a group does not always mean real collaboration is happening. Some students may be very active, while others stay quiet or do not contribute much. By evaluating collaboration, we can check if learners are really sharing ideas, helping each other, and building knowledge as a team. It also helps teachers or the system give support to students who need help. Feedback from the evaluation helps students understand how they worked with others and how they can improve. CBHLE systems can also use this information to adapt the activity or group structure to make collaboration better. Evaluation makes group work fairer, especially when grades are involved, because it shows what each student actually did. It also helps students develop teamwork skills like communication and cooperation, which are important for their future. For teachers and researchers, collaboration evaluation provides useful information to improve teaching strategies, design better learning activities, and understand how students learn together with digital tools. students engage actively in group tasks, ensuring that each member contributes in a balanced way.
- **Conflict management:** In case of tensions or conflicts within the group, the tutor intervenes to facilitate communication and resolve disagreements.
- **Formative assessment:** The tutor provides more occasional and formative evaluation of learners' progress, giving feedback and adjusting support strategies.

1.5.3 Other essential roles

In a collaborative learning situation supported by a CBHLE, these roles are considered as main: teacher, IT designer, tutor, learner, and administrator [7]. This organization of roles

is illustrated in Figure 1.1, which provides a synthetic overview of their respective responsibilities and interactions within the CBHLE-supported collaborative learning environment.

The first role is that of the teacher, who is responsible for designing the educational content of the courses. The IT designer, or sometimes the teacher themself, then creates the media (text, image, video, etc.). The teacher also creates learning paths (collaborative learning situations) that use the educational content.

The tutor monitors learners' work and assists in both individual and collaborative learning. They may also regulate their learning paths. The learner consults or downloads the recommended educational content online, organizes their work, does exercises, self-assesses, submits work to the tutor who evaluates it, shares, communicates, and collaborates with other learners for a common goal, which is learning.

The administrator installs and maintains the system, manages the students' administrative registration, and controls access and rights to educational resources. Thus, the administrator refers to a platform-specific role, not a usual administrative role.

Furthermore, these actors must acquire specific skills to benefit from the proposed collaborative learning situations. Designers must be trained so that the content is presented in a way conducive to collaboration, and tutors must also adapt to the monitoring of collaborative exchanges.

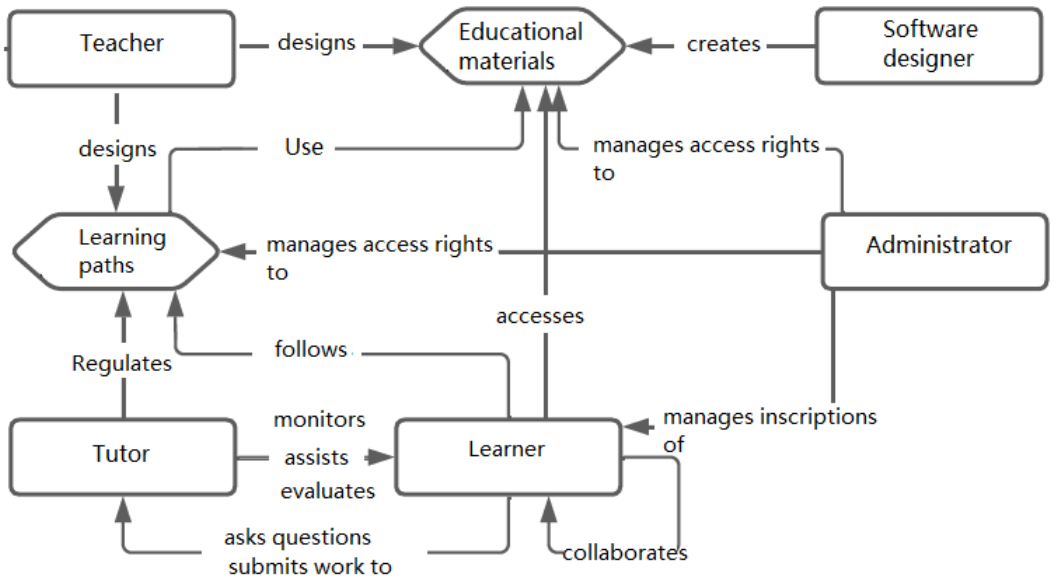

Figure 1.1: Actors in a collaborative learning situation supported by a CBHLE

1.5.4 The role of the online tutor

The shift from face-to-face training to distance learning changes all the roles of the actors (designer, trainer, learner, etc.). In addition, new actors appear, such as the distance tutor, the administrator, and the learning platform manager [45].

Tutors have roles such as guide, facilitator, instructor, and evaluator. They adopt and implement strategies related to the chosen learning/teaching paths in the collaborative learning situation. They also have a role in supervising, supporting learners, encouraging learning, and communicating rules in the work environment [47, 1].

The distance tutor has a role as a facilitator and mediator: "The tutor must know how to guide, listen, advise; anticipate upcoming difficulties; think in terms of objectives and not time spent; share everyone's contributions (...)" [57].

1.5.4.1 What skills are needed to become an online tutor?

First of all, a skill is "the ability to use organized sets of knowledge, know-how, and attitudes to carry out a number of tasks" [47].

Tutoring is a pedagogical guarantee of online training, so it is important that tutors have a skills profile that matches the roles they must fulfill [56].

In general, the tutor must have teaching and relational skills, technical skills, and subject-specific skills [1] [57] [27] [7].

The tutor's teaching and relational skills are those usually required for a trainer or teacher. However, distance learning requires different approaches than face-to-face, whether in the timing of interventions or in the communication tools used.

It is necessary for the tutor to handle the CBHLE platform tools, not only to access the resources and interact with learners, but also to help them if basic technical problems occur.

The tutor should fully master the content or at least be able to provide relevant resources to learners, especially if they are not the author of the course [1].

1.5.4.2 The role of the tutor in a collaborative learning situation supported by CBHLE

Many researchers [48] highlight the complexity of the role of the online tutor in the context of an online collaborative learning situation.

In addition to their roles as an online tutor, the tutor in this learning approach must also act as a moderator, where they must play a social role by developing and regulating interactions, setting rules for interactions, monitoring and intervening to redirect the work group in a productive direction, and evaluating collaborative work. So, this is a motivational and relational support [57].

1.6 Collaboration and Cooperation

The terms "cooperation" and "collaboration" are often used without distinction when it comes to working together. However, research fields analyzing human work groups make a clear distinction between the two terms [36].

The term collaboration is used for situations where actors share the same goals throughout the completion of tasks [53].

In cooperation, different actors have distinct sub-goals related to a common goal.

These definitions of collaboration and cooperation can even go beyond the context of human collective work to include collective human/machine activities [8].

Cooperation has been defined as solving a common problem by several agents with a distribution of tasks to be performed among the agents. On the other hand, collaboration is defined as solving a problem by several agents where the tasks making up the problem are performed together by all the agents.

1.6.1 Collaborative learning and cooperative learning

We have seen that there is some ambiguity between the terms "cooperation" and "collaboration". This ambiguity extends to the expressions "cooperative learning" and "collaborative learning". According to [36], cooperative learning often involves collaborative practices. So, it is impossible to apply the previous definitions of cooperation and collaboration directly to learning.

Cooperative learning can be seen as a series of processes that help people interact toward a particular goal or to achieve a final product [36].

These same members are involved in a shared task or achievement, for which they work together. It is work within small groups, where members share a common goal, which helps optimize each person's learning [44].

Collaborative learning can be seen as a personal philosophy and not just a classroom technique. Learners are responsible for their own learning as well as that of others [60].

Collaborative learning is based on an approach that gives learners a lot of freedom, where activities are not highly directed and they are responsible for managing much of their group work.

When we talk about collaborative and cooperative learning, similarities and differences appear. The similarities lie in the fact that both types of learning refer to working together and group activity where group members have common goals. They aim for collective learning.

Cooperative learning and collaborative learning are distinguished by four main criteria: the structuring of activities, teacher control, the roles of learners, and the learners' social

skills [58].

First, in a cooperative learning situation, activities are more strongly structured and teachers have more control over the activities than in collaborative learning.

And although both promote the transfer of authority from teachers to learners, cooperative learning allows the teacher to move from group to group, observe interactions, listen to conversations, and intervene when they deem it necessary.

On the other hand, collaborative learning does not allow this kind of behavior and the teacher in this type of learning does not actively control the groups. In other words, there is a relatively partial transfer of authority in cooperative learning, and a more complete one in collaborative learning [61].

Furthermore, learners do not have the same responsibilities in these two approaches. Cooperative learning tends to be more structured in its approach to small-group learning and assumes that learners are trained in group activity.

That means in cooperative learning, learners' roles are clearly defined in advance and they are always taught social skills. In contrast, in collaborative learning, it is assumed that learners already have the skills to work in a group and they are allowed to negotiate their own roles and organize themselves freely.

The internal organization of the group thus reveals a second point of difference. In cooperative learning, each person plays a specific role even if rotations are planned so that everyone can take on different roles.

Collaborative learning does not follow the same principle. On the contrary, it encourages a non-structured group setting where all learners can take the same roles at the same time and the group activity in this type of learning is characterized by exploring, exchanging, and sharing ideas while considering group members as equals.

The group structuring specific to cooperative learning is linked to an additional feature: interdependence. In learning contexts, specialists see it this way: learners in cooperative groups can reach their learning goals if, and only if, the other students they are cooperatively associated with reach theirs [61].

When we talk about interdependence as a specific feature of cooperative learning, we must also talk about equity among learners as an additional feature characterizing group activity in collaborative learning.

Since group members are seen as equals and role choices are made through negotiation, it is possible to encounter a situation of equity among learners where one learner may take only one role — such as leader or follower — throughout the learning period.

In some conditions, the teacher's role should not be overlooked. For example, in primary school where spelling, grammar, mathematics, historical facts, and geography are considered basic knowledge for the learner.

In this case, group functioning and collective activity alone, ignoring the teacher's role,

cannot fully suffice. So, cooperative learning is more suitable in this case.

Conversely, higher education is more oriented toward developing critical thinking, seeking non-fundamental knowledge. It mainly involves addressing questions that involve uncertain or ambiguous answers. So, collaborative learning is more appropriate in these situations.

The Table 1.1 is a comparative table between the two types of learning that summarizes what has been previously presented.

Distinctive criteria	Cooperative learning	Collaborative learning
Activity structuring	Structured (Interdependence principle)	Unstructured (Sharing, pooling of knowledge)
Teacher control	Real (Group observation)	Weak (Learners' autonomy)
Learner responsibility	Guaranteed by interdependence	Uncertain (At each one's discretion)
Equity among learners	Impossible (Each has a role with rotation)	Likely (Free group organization)
Learning objectives	Fundamental knowledge related to various school activities	Non-fundamental knowledge: critical thinking, reasoning, collective discovery.

Table 1.1: Comparative aspects between cooperative and collaborative learning.

1.7 Advantages of Collaborative Learning

In General, face-to-face collaborative learning offers good interaction between each learner, their teachers, and classmates, but it requires fixed time and location. On the other hand, individual learning supported by CBHLE offers learners flexibility in time and location, but it limits their interactions with teachers and peers. Collaborative learning supported by CBHLE combines the advantages of face-to-face teaching (interaction) and individual learning (flexibility) and avoids their disadvantages.

1.7.1 Advantages of Collaborative Learning Supported by CBHLE Compared to Individual Learning

According to Panitz[58], collaborative learning offers a way of relating between learners and values everyone's abilities and contributions. It is based on acceptance of others and mutual respect, but also on sharing responsibilities, the absence of competition, and the presence of consensus. According to Bruffee (1999) [78], collaborative learning is grounded in construc-

tivist values where autonomy, active engagement, reflection, and social interaction are central to the learning process. It is defined as "a dynamic and reflective process that supports learner growth through collaboration and mutual understanding." In this type of learning, learners adapt to the demands and benefits of collaboration by engaging in discussion, negotiation, and joint problem-solving with peers. As a result, they develop greater autonomy, maturity, and self-regulation, reinforcing both cognitive and personal development .

1.7.2 Advantages of Collaborative Learning Supported by CBHLE Compared to Face to Face Learning

1. **Time Flexibility** Distance collaborative learning gives learners time flexibility, allowing them to choose activities better suited to their needs and interests and manage their own learning.

2. **Time to Reflect** A major advantage of distance collaborative learning over face-to-face learning is the reflection time. In face-to-face learning, even attentive listening to a teacher's or peer's presentation allows only limited reflection. In contrast, distance learning gives more opportunities for reflection and promotes learning autonomy, helping learners better understand and think more critically about taught concepts and completed tasks.

3. **Text-Based Communication Is More Demanding and Educational Than Oral Communication** Distance collaborative learning does not exclude synchronous communication, but asynchronous written communication requires collecting ideas and supporting them rigorously to present them in a logical and coherent form. This is more demanding and educational than oral contributions, where reflection time and speaking time are limited.

4. **Written Messages Focus on Content and Balance Relationships Between Learners** In face-to-face collaborative learning, social and physical cues from the speaker often draw attention away from the message content. The absence of such cues in online learning highlights content and strengthens its educational impact on participants.

5. **Emulation and Mutual Help Effects** During distance collaborative learning, familiarity grows among learners despite differences in age, culture, prior knowledge, and profession. In a climate of spontaneity and mutual help, relationships between learners and their teachers/tutors become less hierarchical.

6. **Supervision Capacity of Tutors and Trainers** Trainers, tutors, and facilitators of distance collaborative learning can supervise many small groups online, something they cannot easily do in face-to-face learning.

1.8 Limits of Collaborative Learning

Despite the advantages of collaborative learning supported by CBHLE, there are some limits to consider.

1.8.1 Limits of Collaborative Learning in Terms of Group Balance

Collaborative learning makes groups more independent, but learners may not take equal responsibility, and their levels of involvement are usually not the same [62]. We know that teachers have limited control over collaborative groups, making it hard for them to detect imbalances and intervene [63]. Collaborative groups are therefore not protected from serious differences in how much each member invests in the collective activity [64]. Some learners may invest a lot, while others much less. In such situations, it's unlikely that learners see themselves as equals.

On the other hand, the high level of autonomy in this learning mode can lead some learners to take on too much responsibility and others too little [65].

Who participates? Who doesn't? Who helped whom? Who did what? These questions are often hard to answer when analyzing how a collaborative group functions [36].

1.8.2 Current Limits of Collaborative Learning Supported by CBHLE Regarding Group Heterogeneity

Research in this area has shown that a very heterogeneous collaborative group does not work well [66]. So, to create good dynamics, it's better to form groups with learners who have similar profiles, levels, and work habits. But then, what about the value of social and cognitive conflict? Forming groups based on level seems to go against the philosophy of collaborative learning. Yet, the search for homogeneity among group members can lead to the creation of different types of groups (groups of the best collaborators and groups of independents).

The author in [67] proved that the best learners contribute more to the group synthesis. So, what is the point of collaborative learning if it only involves the best learners?

1.8.3 Current Limits of Collaborative Learning Supported by CBHLE Regarding the Size of Tutored Groups

Fjuk notes that collaborative learning scenarios work better with small groups [68]. A collaborative team of four or five members can work if no one hides their inaction behind others. But what should we do if we are required to form larger groups?

1.8.4 Current Limits of Collaborative Learning Supported by CBHLE Regarding Assessment Procedures

Assessment methods are a source of difficulty due to the many elements to consider in a collaborative context, especially those related to group interactions and each participant's involvement [69]. It's not just about checking individual knowledge and summarizing ability, but also about evaluating participation in team work.

The second question is how much of the grade should be assigned to this aspect. The assessment grid must check both knowledge acquisition and learners' collaborative participation [70].

1.9 Classification of Collaboration Approaches in Computer-Based Human Learning Environments (CBHLE)

There are a number of learning approaches, such as traditional approaches (teacher-centered) and collaborative approaches (learner-centered). This classification is based on four different perspectives: subject, objective, method, and tool [71]. Each perspective captures a particularly relevant aspect of the approach. Each approach can be characterized by four perspectives: what? (subject), why? (objective), how? (method), and which? (tool).

Each perspective consists of several facets that present various attributes. The attributes are defined by appropriate values. Furthermore, there is an intersection between the perspectives.

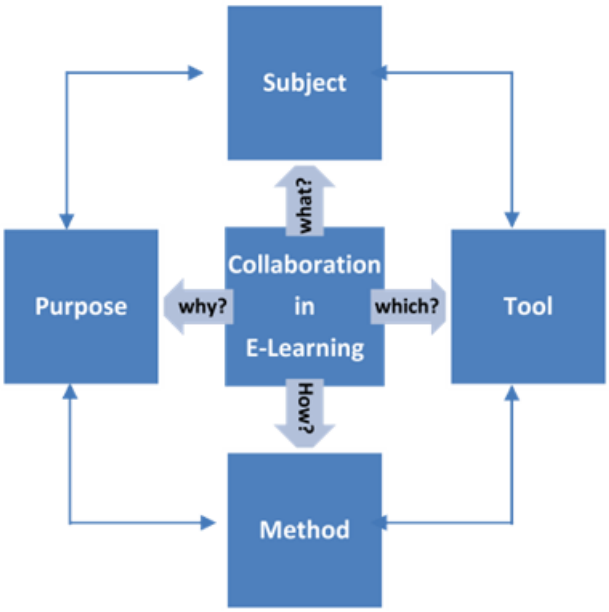

Figure 1.2: Framework for Classifying Collaboration Approaches in Collaborative Learning via a CBHLE

1.9.1 Subject Perspective

This perspective addresses the "what" aspect. It defines the approach and its components, including type, context, actors, and adaptability.

1.9.1.1 Type Facet

Represents the type of approach to be developed: technical, social, or pedagogical.

Some approaches can be classified as both technical and at the same time social or pedagogical.

- A collaborative learning approach is **technical** if it ensures institutional and administrative management of data and learning processes.
- A collaborative learning approach is **social** if it facilitates features such as interactions between learners and external experts.
- For a collaborative learning approach to be **pedagogical**, it must be designed primarily for educational purposes. This type also supports both individual and collaborative assessment.

In summary, the attributes of the type facet can be classified as follows:
SET: (ENUM{ Technical, Social, Pedagogical}).

1.9.1.2 Context Facet

The second facet of the subject perspective is the context of the approaches.

- Learning can be **content-based** according to a single learner model where the learner interacts only with content objects.
- Learning can be **scenario-based** such as problem-based or case-based learning. During this process, students must apply their knowledge and skills in a safe and real context.

In summary, the attributes of the context facet can be classified as follows:
SET: (ENUM{ content-based, scenario-based }).

1.9.1.3 Actor Facet

The actor facet represents the actors involved in the life cycle of the approach, namely: the learner, the teacher, the administrator, and the expert.

- For a collaborative learning approach to be successful, it requires active interactions primarily between the **learner** and the **teacher**.

- **Administrators** play an important role as managers.
- An **expert** from outside the educational environment could be a researcher or even a social network member of the learner.

The actor facet can be defined as follows: SET: (ENUM{ Learner, Teacher, Administrator, Expert }).

1.9.1.4 Adaptability Facet

This refers to the level of adaptability of the collaboration process in the implemented approach. Measuring the level of adaptability is very important to determine to what extent the approach adapts to the needs and skills of the learner. The adaptability level can be stated as follows: SET: (ENUM{ High, Medium, Low }).

1.9.2 Objective Perspective

This perspective addresses the "why?" aspect of the approach. The "why?" perspective explains in detail the development objectives of the collaborative approach. These objectives can be discussed based on various facets, which are classified according to the role that the collaborative approach aims to play.

1.9.2.1 Collaborative Scenario Facet

Collaborative scenarios in a collaborative learning approach using a CBHLE can include:

- **Project-based learning**: This form of learning requires learners to work together and independently over extended periods to develop a specific project/product/presentation.
- **Collaboration team**: This form of collaboration aims to set up team or group work activities where team members must communicate, collaborate, and perform tasks independently of time and space.
- **Mentoring and coaching**: Mentoring is a process that includes elements of coaching. It is a relationship between a coach and a learner where the coach plays a role in guidance, support, advice, and assessment.
- **Peer-to-peer (P2P) feedback**: In contrast to team collaboration, P2P feedback focuses on the peer-to-peer network. Each peer acts as both a client and a server for exchanging messages and transmitting files.
- **Debate and discussion**: A form of collaborative scenario that allows discussions and debates between learners.

- **Social interaction**: Another form of collaborative scenario where the learner can build their social network and external knowledge (friends, family, etc.) during the learning process.

In summary, the collaborative scenario facet can be summarized as follows:

SET: (ENUM{ Project-based learning, Collaboration team, Mentoring and coaching, P2P, Debate and discussion, Social interaction }).

1.9.2.2 Collaborative Service Facet

The second facet of the perspective concerns the collaboration features. The collaborative approach should provide services such as monitoring, tracking, visualization, and assessment of learner behavior by teachers. Therefore, the attributes of the collaborative service facet can be summarized as follows:
SET: (ENUM{ Monitoring, Learner teaching functionality, Visualization, Grading and evaluation }).

1.9.2.3 Principle Facet

This facet determines the basic principles for a collaborative approach, which are:

- **User-centered**: This means that students are responsible for extracting, creating, and modifying content or knowledge through collaboration.
- **Participatory architecture**: The CBHLE should encourage learners to add value to the collaborative space they use, for example by modifying the content.
- **Openness**: This principle provides actors with a shareable and open space for a wide audience.
- **Interaction**: The CBHLE must enable students to interact and learn effectively.
- **Social networks**: Learners can build and expand their personalized social networks.
- **Collaboration**: The key principle in the development of the CBHLE is to support communication and collaboration.

The principle facet can be stated as follows:
SET: (ENUM{ User-centered, Participatory architecture, Openness, Interaction, Social networks, Collaboration }).

1.9.2.4 Process-based Objectives Facet

This facet describes how to measure the level of achievement of learning objectives. For example, by using a process of survey, observation, assessment, comparison, etc. In summary, the process-based objectives facet can be defined as follows:

SET: (ENUM { Survey, Observation, Evaluation, Comparison, etc. }).

1.9.3 Method Perspective

The method perspective describes **how** collaboration was presented in the collaborative learning process. It has three facets: collaboration method, modeling, and evaluation.

1.9.3.1 Collaboration Method Facet

The collaboration method facet indicates the interaction and integration methods used to achieve the required level of collaboration.

- **Interaction methods** can be classified into seven techniques: collaborative writing, chat communication, social interaction, file sharing, brainstorming, sharing links and bookmarks, and sharing media and e-learning services.
- **Integration methods** represent the software tools needed to facilitate collaboration and interactions. They are classified into the following categories:
  - Category 1: General-purpose social media extensions (plug-ins) with educational support functions.
  - Category 2: Creation of standalone educational social media platforms.
  - Category 3: Integration of multiple social media tools into the educational platform.

In conclusion, the method facet has two attributes: interaction and integration with the following enumerated values:
Interaction: SET: (ENUM { Collaborative writing, Chat communication, Social interaction, File sharing, Brainstorming, Sharing links and bookmarks, Sharing media }).
Integration: SET: (ENUM { Plug-in, Stand-alone, Mashup }).

1.9.3.2 Modeling Facet

The modeling facet concerns the modeling of the collaboration context in terms of location and time. The collaborative learning process can be:

- **Co-located**: Taking place in the same location.
- **Distance-based**: Taking place in different locations.

- **Synchronous**: Taking place at the same time.
- **Asynchronous**: Taking place at different times.

The modeling facet is summarized as follows:
Collaboration space: SET: (ENUM { Co-located, Distance-based }).
Collaboration time: SET: (ENUM { Synchronous, Asynchronous }).

1.9.3.3 Process-based Evaluation Facet

This facet determines the methods for evaluating the collaborative learning process, which can be:

- **Descriptive**: By observing how two or more people learned and how the collaboration process went.
- **Experimental**: By comparing an intervention with a control condition in terms of one or more variables.
- **Iterative**: By evaluating evolving interactions in an iterative process.

The process evaluation facet is defined as follows:
Collaboration space: SET: (ENUM { Descriptive, Experimental, Iterative }).

1.9.4 Tool View

The last view is the tool view. This view describes the **who** of the comparison framework. It includes four facets: communication platforms, learning objects, interaction policies, and process description.

1.9.4.1 Communication Platforms Facet

This facet can have many values depending on the CBHLE used. These platforms can be wikis, social networks, forums... So, the communication platforms facet can be defined as follows:

1.9.4.2 Learning Objects Facet

A Learning Object (LO) represents any digital resource that can be used to support learning and any item that can be included in the course, such as animations, websites, tutorials... Each learning object has the following attributes:

- **Granularity** of the learning object is defined by the level of aggregation and the complexity of the supported content. The **aggregation level** concerns the size of the items. It starts at level 1 and gradually increases depending on the size of the web

page elements. On the other hand, the **complexity of content** concerns the degree of combination of content elements. This concept suggests that the reusability potential of the object reaches its maximum with a single basic object. The granularity attribute can be written as follows:
SET: (ENUM { Aggregation level, Complexity }).

- **Creation** of a learning object includes two main facets: the first is the **content of the object** itself and the **metadata** describing the learning object. The content can be classified into two types: multimedia object such as: text, pdf, image, audio, video... and real-world objects, for example a person, an organization, an event... The creation attribute can be written as follows:
  SET: (ENUM { Content (multimedia object, real-world object), Metadata }).

- **Standards**: Standardization aims to enable interoperability and reusability of online learning content in a transparent way without manual intervention. The standards attribute can be written as follows:
  SET: (ENUM { SCORM, ISM-LD, IEE-LOM, SOA }).

- **Languages**: This is the language used for creating learning objects. These may be web pages or mini applications. The language attribute can be written as follows:
  SET: (ENUM { Languages }).

1.9.4.3 Interaction Policies Facet

Interaction policies define how learners interact with each other. They can be:

- **Ad-hoc**: This is the informal way of interacting, without restrictions from the institution or teachers.

- **Protocol**: This form of interaction is built based on the policies of the institution or facilitator to create a more formal interaction framework.

- **Mix**: This technique is a kind of mix of ad-hoc and protocol interaction.

The interaction policies facet can be stated as follows:
SET: (ENUM { Ad-hoc, Protocol, Mix }).

1.9.4.4 Process Description

The process description indicates whether the flow and descriptions of the learning processes are available or not. The process description facet can be stated as follows:
SET: (ENUM { Available, Not available }).

The following Table 1.2 of collaborative learning approaches in CBHLEs provides a detailed classification of collaborative learning approaches in Computer-Based Human Learning Environments (CBHLEs). It is organized around multiple perspectives—such as subject, purpose, method, tool, and modeling—and includes key facets like actors, context, adaptability, and interaction policies. This classification offers a structured framework to better understand the diversity and specificities of collaborative learning situations in CBHLEs.

Classification	Subject view	Type facet	Type : SET(ENUM{Technical, Social, Pedagogical}).
		Context facet	Context : SET : (ENUM { content-based, scenario-based }).
		Actor facet	Learner : SET : (ENUM { Students, Employee }. Teacher : SET : (ENUM { Instructor, Facilitator }.. Administrator : SET : (ENUM { Institute, Developers }. Expert :SET : (ENUM { Researchers, Outsiders }.
		Adaptability facet	SET : (ENUM{ High, Medium ,Low }).
	Purpose view	Scenario facet	Project based :SET : (ENUM{ Product, Presentation}). Team and collaboration: BOOLEAN. Mentorship and coaching :BOOLEAN. Peer to peer feedback : BOOLEAN. Debate and discussion : BOOLEAN. Social interactions : BOOLEAN.

Classification		Services facet	Monitoring : BOOLEAN. Learner tracking : BOOLEAN. Visualisation : SET : (ENUM{graphic, Algorithm}). Grading and evaluation : BOOLEAN.
		Principle facet	Principle : SET : (ENUM { user-centred, participative architecture , openness, interactions, social network, collaboration }).
		Process based goal facet	Process based goal : SET : (ENUM { observation, investigation, Evaluation, Decision making, comparison, discovery, illustration }).
	Method view	Collaboration methods facet	Platform : SET : (ENUM {Collaborative writing, Communicating chatting, file sharing, brainstorming, sharing links and bookmarks, media sharing...}) Integration : SET : (ENUM { Plug-in, stand-alone, Mashup }). Collaboration space : SET : (ENUM { Collocated, Remote }).
		Modeling facet	Collaboration time : SET : (ENUM { synchronous, asynchronous}).
		Process goal evaluation facet	Process based goal : SET : (ENUM { Descriptive, Experiment, Iterative }).
	Tool view	Platform facet	Platform : SET : (ENUM { Wikis, Social networking, Document sharing, Mindmaps, Brainstorming, Discussion forums... }).

Table 1.2: Classification of collaborative learning approaches in CBHLEs

1.10 Conclusion

This chapter explored the foundations and issues of Computer-Based Human Learning Environments (CBHLEs), particularly their impact on collaborative learning. We saw that CBHLEs offer powerful tools to support collaboration among learners by facilitating interaction and knowledge management. These systems play a crucial role in improving learner engagement and providing personalized support, while allowing real-time analysis of progress and group dynamics.

However, challenges remain, particularly concerning conflict management within groups, the balance of contributions among members, and the diversity of learner levels. Group heterogeneity, as well as difficulties related to assessing learning in a collaborative context, require special attention. These challenges directly influence the effectiveness of collaborative learning supported by CBHLEs, hence the importance of adapting tools and methods to the specific needs of learners.

The classification of different collaboration approaches in CBHLEs, based on criteria such as subject, context, and actors, helps better understand the specificities of each learning situation. It also provides a framework for designing more suitable and effective systems for each type of collaboration.

In short, although CBHLEs have considerable advantages, their effective implementation depends on taking these various challenges into account.

Chapter 2

Related Work and Scientific Positioning

2.1 Introduction

This chapter presents the theoretical foundations and the work directly related to our research. It aims to situate our contribution within the state of the art on trace analysis in Computer-Based Human Learning Environments (CBHLEs), the modeling of these traces, and the calculation of interaction and collaboration indicators.

We begin by introducing essential concepts related to digital traces: their collection, their issues, and the limitations of raw log files. This leads to the need to move from a raw trace to a modeled trace, which is better structured and usable in educational analysis processes. We then present Modeled-Trace-Based Systems (MTBS), detailing the concepts, model-driven engineering (MDE) approaches applied to traces, and the main expected features of such systems.

In the second part, we address the languages and methods for calculating indicators from modeled traces, with a focus on collaboration indicators. We define the concepts related to evaluating collaboration in online learning situations and present existing indicator patterns as well as the main languages dedicated to their formalization.

The third part of this chapter is devoted to previous work directly related to our research topic. We distinguish two main categories:

- ad hoc approaches for calculating collaboration indicators, often specific to a particular context or environment;
- more generic methods and tools based on modeled formalisms, particularly those based on model-driven engineering.

Finally, this chapter ends with a critical discussion and a scientific positioning of our contribution in relation to existing work. We highlight the limitations identified in the state of the art and specify the specific contributions of our approach.

2.2 Traces in CBHLEs: From Collection to Modeling

This section presents the foundational concepts related to digital traces in Computer-Based Human Learning Environments (CBHLEs), highlighting their role in understanding and supporting learning processes. It explores the definitions, objectives, and technical characteristics of traces, starting from the raw data collection to their transformation into structured and meaningful representations. We also discuss the importance of modeling traces to facilitate pedagogical analysis, and we introduce key concepts such as temporal extension and the notion of "obsel" (observed element). Through this, the section provides a conceptual and technical basis for leveraging traces in the analysis and improvement of collaborative learning environments.

2.2.1 Definitions and Challenges of Tracing

According to the French Ministry of Economy, Finance and Industry, tracing is described as follows: tracing means to be able to follow—literally to pursue—the path and activity of the learner during their training journey. The Rennes Academy adds to this definition by specifying that it involves recording, noting, marking, storing information on a material medium in order to be able to reconstruct the history, chronology, and diachrony [36].

2.2.2 Trace in CBHLEs

In the field of Computer-Based Human Learning Environments (CBHLEs), observation is a crucial activity that allows teachers or tutors to better understand what is happening during a learning session, to evaluate learners' activities, or to adapt the learning scenario [66]. Observation mainly aims to collect relevant information about users' interactions with the system. This information, called *interaction traces*, consists of actions and events produced during the use of a CBHLE, such as logins, online discussions, or resource consultations [65].

The trace is thus defined as a structured recording of a user's activities during their learning. It makes it possible to reconstruct the learner's path, highlighting the chronology and dynamics of their actions [67].

From a more technical perspective, a digital trace is made up of objects arranged and inscribed in relation to a representation of the time of the traced activity, called "temporal extension" [75].

A trace is called a collection of temporally situated observed elements. The observed element "obsel" or simply the observed is an atomic object of the trace that represents any structured information resulting from the observation of an interaction [17]. An observed element is said to be "temporally situated", that is, there is an order relation organizing the observed elements of the trace in relation to a time reference (Figure 2.1).

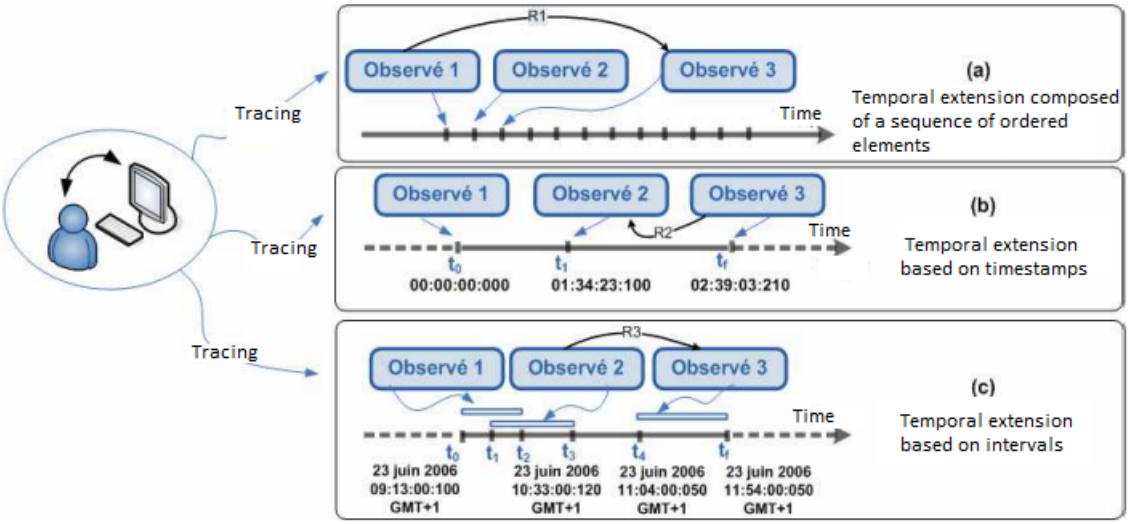

Figure 2.1: Trace, observed element and temporal extension

2.2.3 Trace-Based System

A *trace-based system* (TBS) is a computer system that collects, organizes, and analyzes data about users' actions within a digital environment, such as an online learning platform. These actions, often called "traces," include observable events like sending a message, solving a task, accessing a resource, or interacting with peers. The main goal of a TBS is to use this information to better understand user behavior, support learning processes, and improve the system's response to learners' needs.

Typically, a TBS includes several components: a capture module that records user interactions, a trace model that structures these actions (often using units called *obsels*), and an analysis engine that interprets the data. In some cases, it also provides visualization tools such as dashboards or reports for teachers and learners.

In educational contexts, especially in Computer-Based Human Learning Environments (CBHLE), trace-based systems help track learner engagement, assess collaboration, and provide useful feedback. When traces are structured using an explicit conceptual model, as in model-based trace systems (SBTm), it becomes possible to perform more advanced analysis and to personalize support based on deeper understanding of learning activities.

2.2.4 Limitations of Raw Log Files

In most computer systems, the log file is a method for collecting data. A log file is "an electronic record of the interactions that took place between a system and its users" [72]. These log files can come from different systems such as websites, information systems, blogs, e-journals, etc. Analyzing these files is a research method used to study user behavior [73].

Log file traces are generally raw records of the actions performed by a user (learner, teacher, etc.) in a digital environment. These traces are often collected automatically by the system and stored as text files (log files), usually without a predefined structure that allows for immediate interpretation [74]. They may contain information such as:

- **Timestamp:** Exact time of the action (e.g., date and time).
- **Action:** Type of interaction the user had with the system (e.g., "click," "submit," "login," etc.).
- **User ID:** Who performed the action.
- **Additional parameters:** Some traces may include specific data such as the ID of the viewed item, the time spent on a task, etc.

Example of a trace in a log file:

```
2025-04-08 08:00:00, user1, login
2025-04-08 08:05:00, user1, view_module, module1
2025-04-08 08:10:00, user1, answer_question, question3
2025-04-08 08:15:00, user1, logout
```

However, this raw data is often large, heterogeneous, and difficult to interpret as it is. It is therefore necessary to analyze it, which involves several steps: data collection, processing, structuring, visualization, and finally interpretation of the results [74]. These steps aim to produce meaningful and useful information, in the form of indicators or visual representations, to support pedagogical decision-making.

However, these log files raise several issues that limit their direct use:

- **Lack of structure:** Raw log files are not organized to provide a clear overview of a learner's actions. They are essentially a sequence of independent events.
- **Difficult to interpret:** Data in log files can be hard to use without additional processing, as they often lack detailed context or clear temporal links.
- **Data volume:** Systems generate a large number of events, which can lead to challenges in managing and analyzing the information effectively.

2.2.5 From Raw Log Files to Modeled Trace

Modeling traces is a solution to overcome these challenges. Several studies have proposed approaches to structure and enrich the data collected in log files to facilitate their use and analysis [76]. Trace modeling consists of organizing events, adding context, and integrating temporal relationships.

Modeled traces include:

- **Event organization:** Each trace is categorized by action type, often in a structured format such as a database or XML file. This allows clearer and more precise management of user interactions.
- **Context enrichment:** In addition to basic information such as time, user, and action, modeled traces contain additional details like which module was viewed, what type of question was answered, etc.
- **Temporal structuring:** Events are organized in a chronological sequence that allows tracking the progress of a learner's activity over a given period. This helps analyze the sequence of actions in a temporal context.

These improvements help solve several of the problems related to the use of raw traces.

The works of [75], [17], and [76] have proposed models and techniques to structure these traces and make them easier to analyze.

The transition from non-modeled traces (log files) to modeled traces is a key step to make the collected data more usable. Here are the typical steps of this transition:

1. **Raw trace collection:** Events are collected by the system in the form of log files.
2. **Trace processing:** The raw traces are then processed to add context, structure the data, and organize it chronologically.
3. **Trace modeling:** Once processed, the traces are organized according to a structured model that allows for in-depth analysis.

2.2.6 Modeled Traces and Model-Based Trace Systems (MBTS)

Modeled traces (M-traces) are structured digital representations of user activity that are explicitly associated with a trace model. This model defines the semantics, structure, and temporal organization of the trace by specifying the types of observed elements (obsels), their attributes, and the relations between them. Unlike raw logs, modeled traces enrich the data with contextual, temporal, and semantic dimensions, enabling deeper analysis and interpretation of learner behavior [36]. They serve as instances of a formal model and support advanced applications in learning analytics, such as behavior pattern detection, learner progress tracking, and the generation of pedagogically relevant indicators.

2.2.6.1 Introduction to Modeled Traces

A modeled trace "M-trace" is a digital trace associated with its model, which describes the objects that make it up (observed elements and relations). The trace model can be seen as the vocabulary of the trace, which enables understanding the trace by abstractly describing its elements. The model determines "in intention" all possible traces (that respect this model).

A modeled trace consists of:

- A reference to a m-trace model.
- An origin (start date) and a duration delimiting the temporal extension.
- A list of obsels.
- A set of binary relations between these obsels.

The origin and duration of the m-trace must be consistent with how time is represented as specified by the model. The types and attributes of the obsels, as well as the relations defined between them, must also comply with the model.

Modeled traces are a more structured and enriched form of traces collected from log files. Trace modeling involves organizing and transforming raw data into a structure that allows for better understanding, analysis, and interpretation of learner actions [75]. This typically involves adding context, temporal relationships, and semantic information, thus facilitating the analysis of learner behaviors and the production of meaningful indicators.

A trace model is a formal and structured description of how a trace is represented and organized. It defines not only the structure of the trace but also the content it contains, detailing the properties of the elements that make it up. This includes information about the different types of events or actions recorded, their temporality, the actors involved, and the contexts in which these events occur. Moreover, a trace model specifies the relations between different elements, whether they are dependencies between events, interactions between actors, or how events are linked to specific goals or resources [66].

Figure 2.2 shows an example of an m-trace and its model. Modeled traces are instances of a trace model, where each element corresponds to an instance of a class defined in this model, and each relation is an instantiation of a link or constraint specified in the same model.

2.2.6.2 Example of a Modeled Trace

The following example illustrates a modeled trace corresponding to the typical interactions of a learner in a CBHLE.

User	**Action**	**Context**	**Timestamp**
user1	Login	Platform access	2025-04-08 08:00:00
user1	Viewing	Module 1	2025-04-08 08:05:00
user1	Answering question	Question 3	2025-04-08 08:10:00
user1	Logout	End of session	2025-04-08 08:15:00

Table 2.1: Example of a modeled trace

Figure 2.3 clearly illustrates the trace model, which describes in a structured way the trace corresponding to the example presented above. This model highlights the different observed

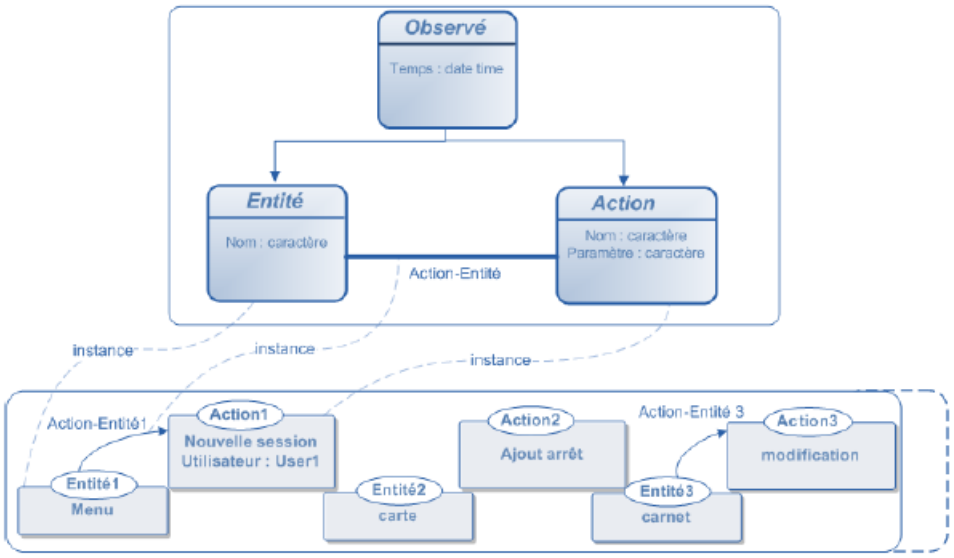

Figure 2.2: Example of an m-trace and its model

elements, their properties, and the relations that link them, thus offering a coherent and complete reading of the dynamics of actions in the studied context.

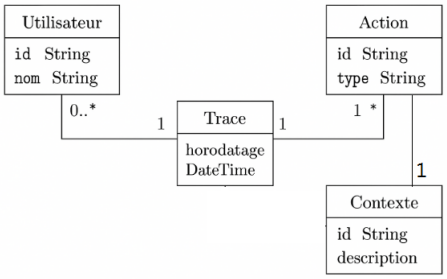

Figure 2.3: Trace model

2.2.6.3 Advantages of Trace Modeling

Modeled traces offer several essential advantages compared to unmodeled traces, especially in learning systems. First, they provide a clear structure that allows the learner's actions to be represented in a coherent and detailed manner, which greatly facilitates their interpretation and processing [77, 81]. Next, the contextualization of recorded events — such as the modules concerned, the resources used, or the types of activities carried out — gives meaning to the observed actions by placing them within their learning environment [79]. Moreover, the temporal structuring of the data makes it possible to perform a detailed analysis of the

sequences of actions over time [77]. Finally, modeled traces provide a solid basis for the automatic generation of relevant indicators. These indicators can then be used to adapt learning paths, assess engagement, or detect difficulties [81, 80].

2.3 Model-Driven Engineering of Interaction Traces

A model is a description of all or part of a system using a clearly defined language. This model is considered correct if its features and behavior evolve over time in the same way as the real system. Any model to be created must comply with a meta-model. A meta-model is the model that defines the language used to express the intended model. Model-driven engineering in CBHLEs, inspired by software engineering, aims to focus on transformations made to the models (depending on their usage contexts), and not on the system's source code, which greatly reduces the effort of designers, teachers, researchers, etc. All the work is based on models using model-driven engineering principles that allow the building and transformation of models [77].

2.3.1 Model-Based Trace System (MBTS)

A Model-Based Trace System (MBTS) is a computer system designed to enable, organize, and facilitate the collection, management, analysis, and exploitation of digital traces resulting from users' interactions with a digital environment. An MBTS generally relies on a modular architecture integrating mechanisms for capturing events (or obsels), modeling traces according to adapted metamodels, as well as visualization or analysis tools. These systems support the implementation of advanced features such as decision support, user feedback, or reflexive support. In learning environments, MBTSs play a crucial role by enabling the monitoring, understanding, and evaluation of learners' behaviors and strategies [79, 77, 80].

The trace base constitutes the informational core of a Model-Based Trace System (MBTS). It gathers all collected traces and their associated models, allowing structured, sustainable, and exploitable storage of data from observed activity. This base ensures continuous access to and efficient querying of traces by the system's different modules, whether for visualization, analysis, or indicator generation. From a technical perspective, the trace base can be implemented in various ways: as a collection of XML files structured according to a given metamodel, or as a temporal database optimized for chronological queries [77, 80, 79]. This component plays a fundamental role in activity traceability, the reconstruction of usage scenarios, and the development of user profiles [81].

As shown in the diagram below (Figure 3.4), such a system includes the general functions of a trace-based system, but now with the ability to handle modeled traces linked to their models.

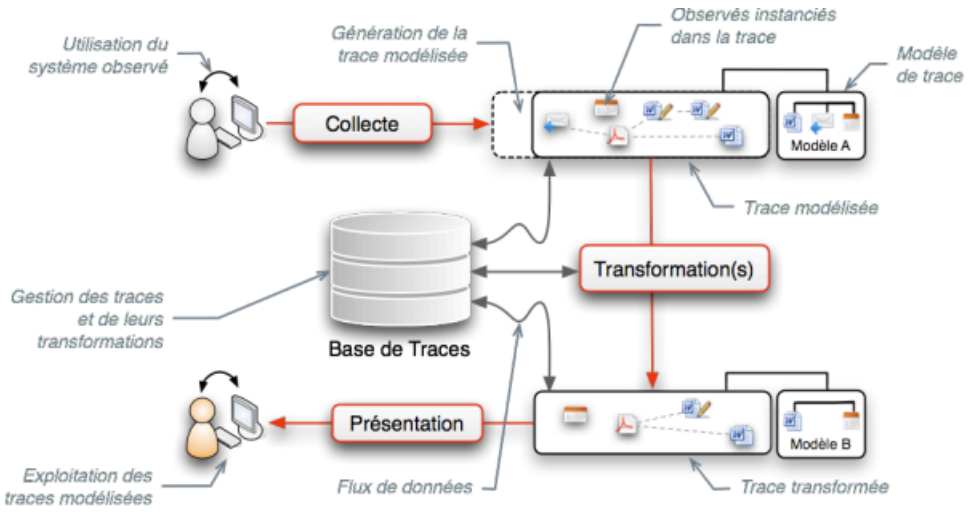

Figure 2.4: General architecture of a modeled trace-based system [77]

2.3.2 Functions Provided by a Modeled Trace-Based System

2.3.2.1 Collection

The collection operation supplies the system with elements that allow the generation—through systematic recording and according to its model—of a modeled trace [77]. The collection must have at least one tracking source, which captures data generated by the learner's interactions with the CBHLE. This tracking source is any structured information flow that can be used to collect traces. It may include audio or video streams or system log data. The collection can (if needed) rely on several tracking sources at the same time, while handling synchronization. The result of the collection is a primary trace associated with its model, which can be used by the system. As the name suggests, this primary trace is the first one obtained at the end of the initial collection process. At the end of this operation, the m-trace is stored in a m-trace database [80].

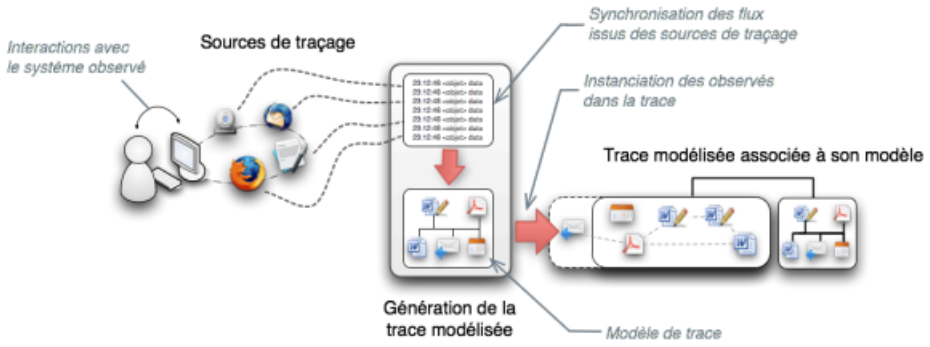

Figure 2.5: Details of the modeled trace collection process in a modeled trace-based system [77]

2.3.2.2 Transformation

The trace obtained after collection is called the primary m-trace. This trace is not always directly usable, and it may need one or more transformations to reach a level of abstraction suitable for the target activity [81]. A transformation of m-traces is any process that converts one m-trace into another. The primary m-traces in a modeled trace database are the only ones not transformed.

Transformation Operators There are two types of transformation operators: those that do not modify the trace model and those that do [77].

Operators that Keep the Trace Model : These operators do not change the model but only the model instances (the traces), such as selection and merging of two instances.

- **Selection Operator**

  This operator selects part of an instance according to one or more criteria such as: time, type of observed element, tool linked to the observed, or actor linked to the observed. The operator is used as follows: $TraceX = Selection(criteria)[TraceY]$ Where TraceX is the resulting trace, criteria is a logical expression based on general attributes of an instance, and TraceY is the source instance.

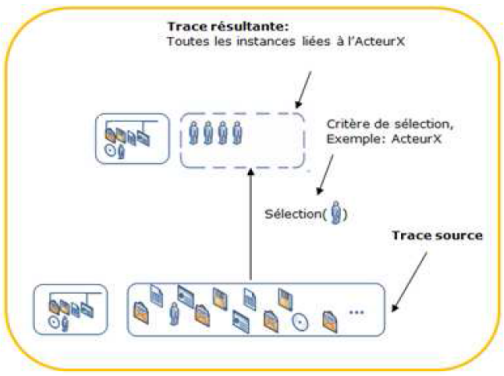

Figure 2.6: Example of a selection [81]

- **Merging Operator of Two Trace Instances**

  This operator combines two source trace instances (TraceA and TraceB) with the same model into a new instance (TraceC). Merging is a union of the observables from both traces. It is used as follows:

  To illustrate this operator in Figure 2.7, we consider the emails sent and emails recorded by ActorX. These two traces can be merged.

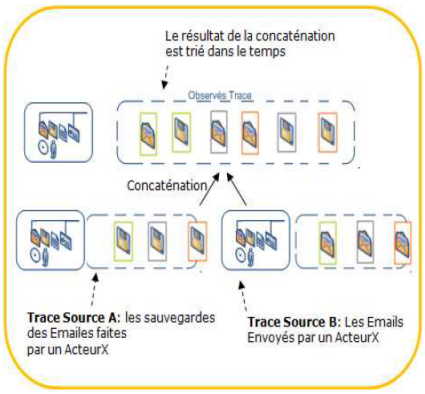

Figure 2.7: Example of a fusion of two traces [81]

Operators that Modify the Trace Model : These operators change the structure of the model. The following operators are proposed: Matching, model rewriting, model pruning, and model insertion.

- **Matching** This operator identifies a sequence of observed elements based on a pattern structure (a task signature). It is used as follows:
  $TraceY = Matching(SignatureA)[TraceX]$
  Where TraceX is the resulting trace, SignatureA is the pattern used to define the search criteria, and TraceY is the source trace. To simplify transformation (and later visualization), the task signature "SignatureA" is made of an input and an output. The input is the pattern to find, made of at least two types of observed elements. The output is the name given to the pattern. For example, an "Effective-Enter" in a chat is a sequence of actions like entering a chat: "ChatEnter", followed by writing a message: "ChatMessage". The signature is: "ChatEnter, ChatMessage, Effective-Enter". Figure 2.8 shows this Matching example. If A=ChatEnter and B=ChatWriteMessage, and if the input sequence is XYXYXYXYXYAYXYXAYXYYXBXYXYX with Matching(A,B), the result is AB.

- **Pruning** This operator is a special case of selection. It deletes classes or class attributes from a model, producing a transformed trace with reduced expressiveness (fine filtering). You give a list of classes to keep in the model. It is used as follows:
  $TraceX = Elagage(traceY$(list of classes or attributes to keep))
  Figure 2.9 shows an example where only two attributes from the source model are kept.

- **Rewriting a Trace** This operator creates a new class name or a new attribute that belongs to a class in a model. It applies to the instances by producing a new trace that matches the updated model. It is used as follows:
  $ModelX.ClassName1 = Réécriture(ModelX, ClassName2)$

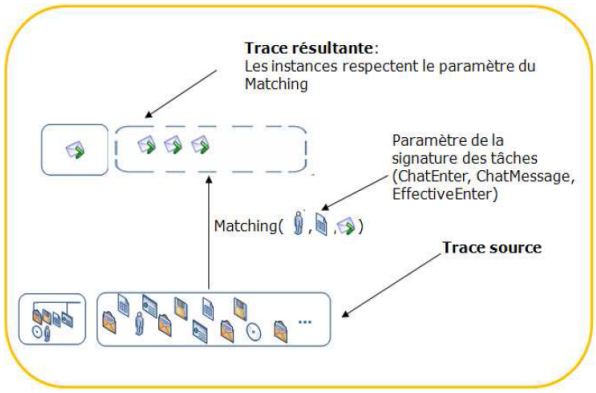

Figure 2.8: Example of a Matching [81]

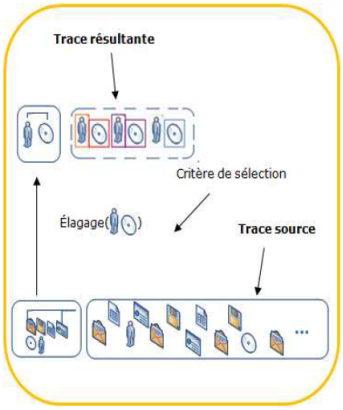

Figure 2.9: Example of pruning [81]

This is illustrated in Figure 2.10, where the class "Enregistrer" in the old trace model is renamed to "Sauvegarder".

- **Insertion** This operator adds a new attribute to a class that didn't have it in the original model structure (Figure 5.1). It is used as follows:
  $TraceX = insertion(TraceY(FieldName)$

2.3.2.3 Presentation

Under presentation, we include visualization and interaction features for modeled traces stored in the m-trace database. Visualization is the graphical and/or textual display of the collection of observed elements in a modeled trace. It can be seen as a specific transformation following a dedicated m-trace model, as shown in Figure 2.12.

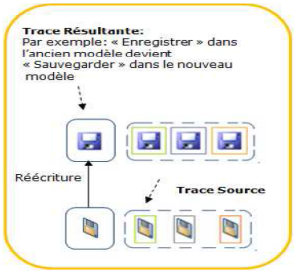

Figure 2.10: Example of rewriting

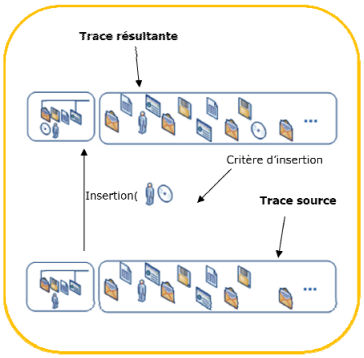

Figure 2.11: Example of inserting a new field into the model [81]

2.4 Interaction Indicators

In the field of CBHLEs (Computer-Based Human Learning Environments), trace analysis is also a very active research area. The data generated by CBHLEs is abundant and includes several meaningful pieces of information to exploit [82]. In recent years, several studies have focused on analyzing traces generated by CBHLEs to achieve different objectives. These studies help teachers or tutors understand the activities done by students during a learning session. The teacher/tutor can then observe, evaluate, and regulate learner activities, and especially improve or adapt the educational scenario they have designed. In fact, trace analysis can be done during or after the learning session using several processes (collection, filtering, transformation, calculation, visualization, etc.) to obtain meaningful and relevant information.

The literature has proposed several definitions for the term "indicator." In our research context, we fully adopt the definition from the ICALTS [83] and DPULS projects [84].

The concept of an Interaction Analysis Indicator is defined as "a variable generally calculated or established using observed data, reflecting the mode, process, or quality of the interaction."

More precisely, Dimitracopoulou and Bruillard (2006) [85] defined an indicator as "a mathematical variable to which a set of characteristics is assigned. It takes on values in numerical or alphanumerical form."

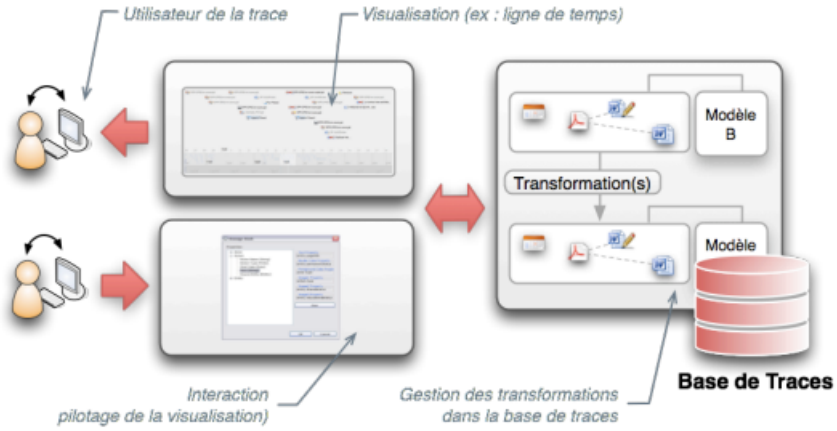

Figure 2.12: Details of the modeled trace visualization process in a modeled trace-based system [75]

Focusing on indicators in the learning field, the DPULS project [84] focuses on the study of usage analysis of a CBHLE to help teachers/designers define pedagogical situations adapted to observed uses. It more broadly defined the concept of an indicator as: "a pedagogically meaningful variable, calculated or established using observed data, and reflecting the quality of interaction, activity, and learning in a CBHLE."

Thus, an indicator is data calculated from traces collected by a CBHLE [86]. According to Djouad [81], each indicator has a name, a textual specification, and a calculation rule.

As shown in Figure 2.13, several steps are involved in obtaining the final value of the indicator: from collecting the necessary traces, to processing them, formalizing the calculation methods, and finally visualizing and interpreting the obtained data. Among all these steps, the central step is the modeling and calculation of indicators.

The value of the indicator allows the construction of more or less detailed feedback for users. According to the categories proposed by [87], this feedback can be a simple visualization of the indicator value ("mirroring"), or the value can be compared with a desired value ("monitoring"), or even used to build more elaborate responses to guide the learner in their learning ("guiding").

2.4.1 Languages for Trace Analysis and Indicator Calculation

In this part, we present languages that can be used in the calculation of indicators.

2.4.1.1 Query Languages

Query languages (SQL, XQuery, SPARQL, etc.) are very often used to perform queries on databases. The two most well-known query languages are SQL (Structured Query Language) and XQuery. They allow different types of operations on a relational database (SQL) or on XML documents (XQuery) [88, 89].

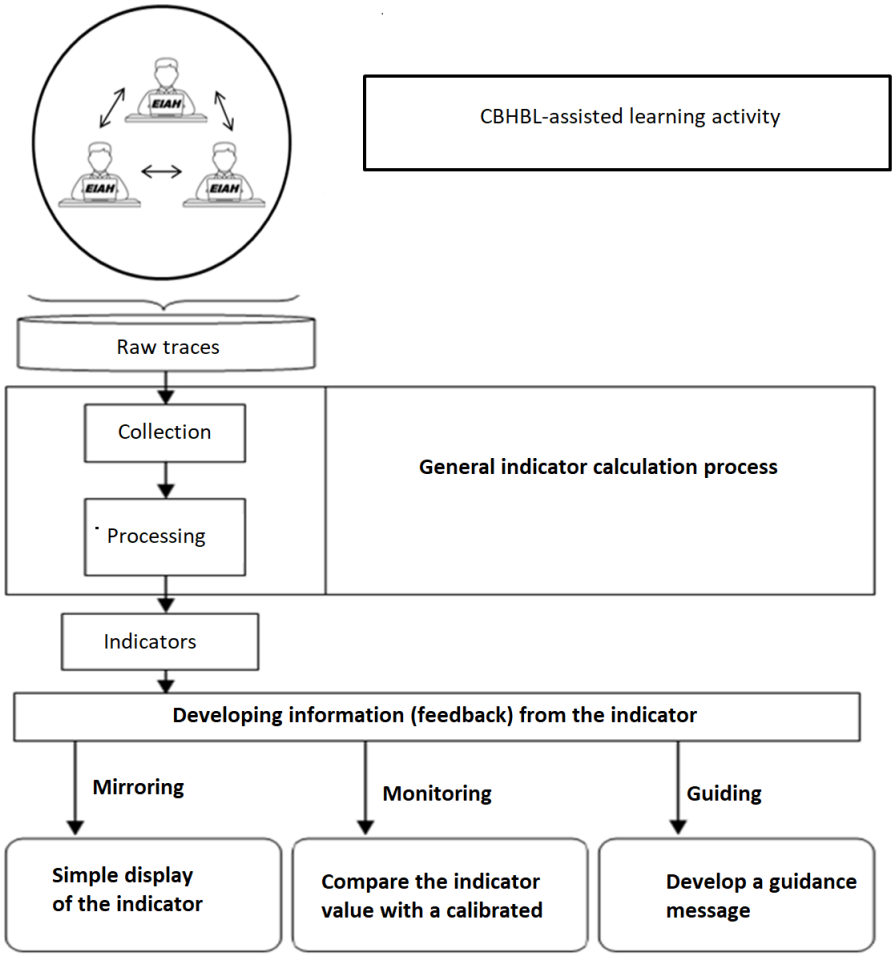

Figure 2.13: Indicators in CBHLEs

SQL

SQL is a declarative language. It consists of three parts: the data definition language, the data manipulation language, and the data control language. The Data Definition Language is used to define elements of a database (tables, columns, keys, indexes, constraints, etc.). Data manipulation is done using the Data Manipulation Language. This language allows retrieving a set of data from those available in the database and updating (inserting, modifying, and deleting) data in the database. Finally, the Data Control Language is used to manage data access rights. SQL queries in the form of SELECT-FROM-WHERE are used to query databases [88].

XQuery Query Languages

XML is increasingly used today to exchange data between various and diverse sources. It was necessary to have a simple language to perform queries on these documents. To read XML data and produce a result in XML format, XQuery is available [90]. XQuery is

compatible with XSLT, the most well-known XML transformation language, and with XPath, a language for extracting nodes in an XML tree. These are W3C specifications. XQuery is a query language that allows extracting information from a document or a collection of XML documents, combining values, operators, and functions to generate a result. It also allows performing complex calculations from the extracted information to produce new XML documents. XQuery operates on XML data and plays a role similar to SQL for relational data. An XQuery query is formalized using five clauses: for, let, where, order by, and return (or FLWOR). A query can contain one or more for and let clauses. The optional let clause allows associating values with one or more variables. The optional where clause allows filtering the tuples produced by the for and let clauses. The optional order by clause specifies whether sorting and the sorting criteria (ascending or descending) are applied. The return clause constructs the result including tuples that satisfy the filter, respecting the order in which the tuples were produced [89, 90].

2.4.1.2 Artificial Intelligence Languages

If traces and indicators are considered as objects, and the indicator calculation method is considered as a set of calculation rules, artificial intelligence languages can be a relevant choice. CLIPS (C Language Integrated Production System) is a declarative and logic-based language specifically designed for the field of artificial intelligence. It is commonly used for building expert systems and particularly for managing production systems. CLIPS is fast and efficient. This language is also designed for logical deduction. Using CLIPS requires defining facts and rules about these facts. Regarding the calculation of indicators from traces, traces and indicators must be represented as facts, and the calculation method must be represented as a set of calculation rules [91].

2.4.1.3 Model Transformation Languages

The calculation of an indicator can be considered as a model transformation [92], from the trace models to the indicator model. In theory, this can be done using the model transformation approach. Model transformation methods are generally based on model-driven engineering (modeled interaction traces), and rely on sequences of trace model transformations and associated instances to collect the observables needed for an explicit calculation of interaction indicators.

2.4.2 Collaboration Indicator Pattern

First, a collaboration indicator in a CBHLE is an interaction indicator that provides information about the level of learners' participation in collaborative work. Several research works focus on the observation and analysis of collaboration. Gendron's work [93] aims to improve collaborative activities through the implementation of indicators. For this, the work

proposed a conceptual framework for modeling collaboration indicators. Each indicator is described through three views: an identity card, a pattern, and a showcase. Figure 2.14 presents these three views.

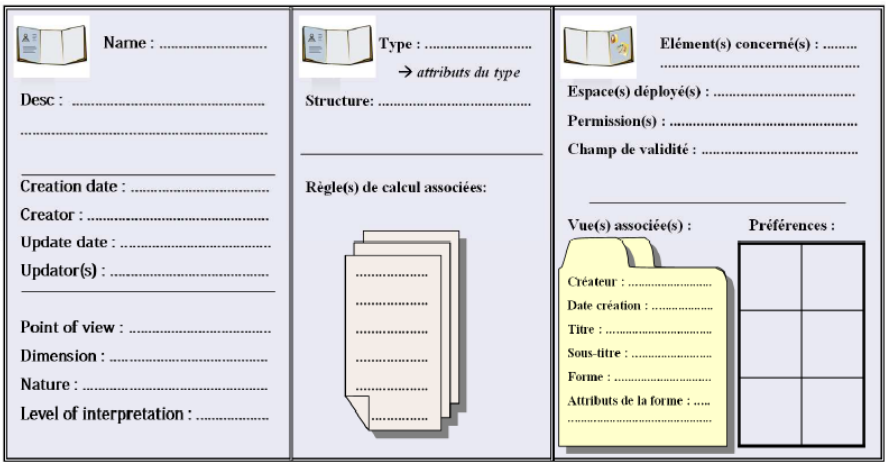

Figure 2.14: The design of an indicator in identity card, indicator pattern, and visualization showcase.

The pattern called Collaboration Indicator Pattern allows specifying the method of calculating indicators. This pattern is composed of a type, a structure, and calculation rules.

- **Type**: this refers to the data type of an indicator. Types can be numeric, alphanumeric, and structured. The numeric and alphanumeric types used include the types Number (e.g.: number of compilations per question) and Threshold (e.g.: each learner's level). Structured types include Table and Discrete Interval types.
- **Structure**: this refers to the production type of an indicator. In this context, two types are considered: elementary indicator and composite indicator. An elementary indicator is calculated from traces, while a composite indicator is calculated from other indicators.
- **Calculation rules**: this refers to applying a set of rules on traces and/or indicators to obtain the indicator values.

The showcase contains information concerning the instantiation of indicators. Concerning these calculation rules, Gendron proposed a calculation language based on the "rule-based system" approach. The language consists of facts and rules. Traces and indicators are the facts. A rule includes facts and actions performed on the facts to obtain the indicator value. All traces contain an action, a resource, an actor, a date, and a value 2.15.

All indicators have the same format which consists of a name, a creation date and a last modification date, a status, one or more creators, an object, a type, and a value (Figure 2.16).

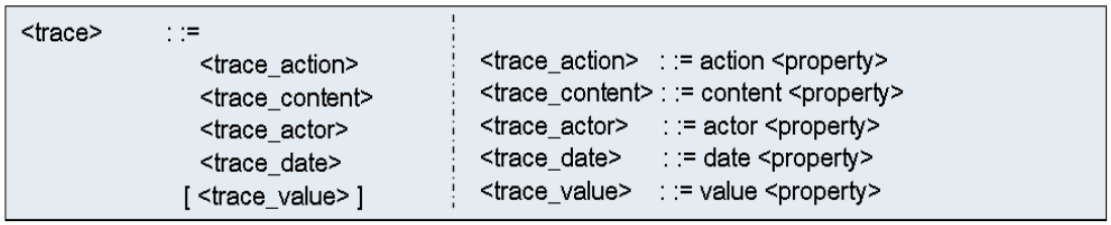

Figure 2.15: The syntax of a trace

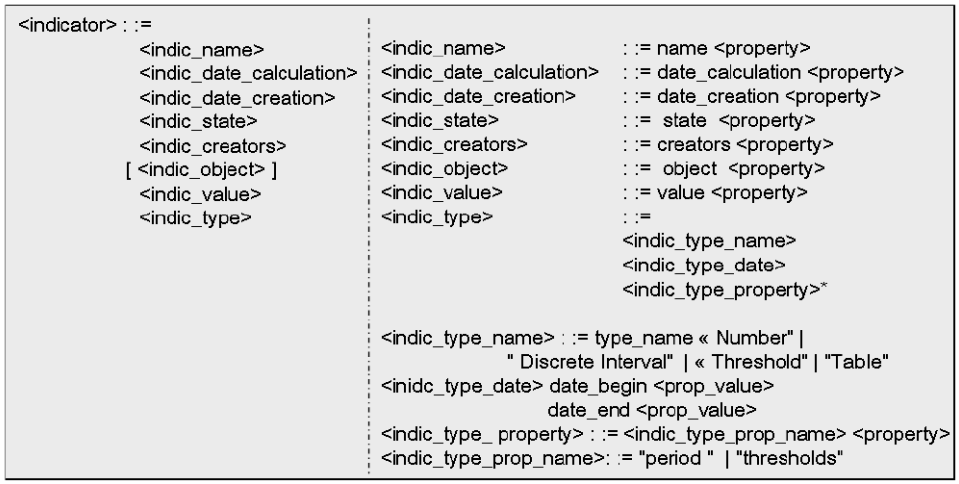

Figure 2.16: The syntax of an indicator

Each rule is identified by a unique name and always includes a <conditions/actions> pair. The "condition" part describes elements that trigger the calculation of the indicator value. These elements can be traces, indicators, and operators such as filtering. The "action" part describes operations (such as display, mathematical calculation, composition) performed on the value of one or more indicators. The "target" part describes indicators modified by the rule. Figure 2.17 presents the syntax of a calculation rule, and Figure 2.18 is an example.

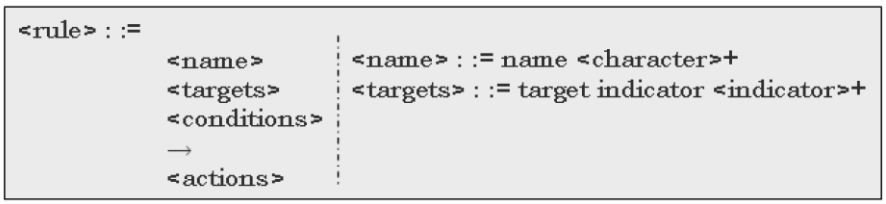

Figure 2.17: The syntax of a calculation rule

```
name ''Construction de l'indicateur Adhésion à un espace''
targets
        name    ''=='' ''Adhésion à un espace''
        state   ''=='' ''actif''
        object  >>     indic_obj
        value   >>     val
filters
        action  ''=='' ''consultation''
                OR
        action  ''=='' ''contribution''
        actor   >>     trace_actor
        date    >>     d
criteria test(indic_obj==trace_actor.groupe)
→
val.incrémenter(d)
```

Figure 2.18: Example of a calculation rule using the rule base

2.5 Related Works

In the field of CBHLEs (Computer-Based Human Learning Environments), several studies focus on the modeling and calculation of indicators. These works can generally be grouped into two types.

2.5.1 First category: Ad hoc calculation of collaboration indicators

The first type concerns the proposal of specific indicators in a particular context.

In [94], the authors proposed a system to monitor learner activity by calculating three indicators from log file traces recorded during a learning activity. The first indicator is used to classify students. It represents a learner's activity rate in a week. It is calculated by taking the number of events performed by the learner during the week compared to the average number of events per learner per week calculated earlier. The second tracking indicator calculates the activity level of each student throughout the course. The calculation of the last indicator gives the participation level, which equals the percentage of learners who completed the tests.

In [95], the authors proposed calculating activity indicators. They are obtained by taking the number of times each learner accessed/used each resource (content pages, forums, wiki, etc.) compared to the activity done by their classmates/group members. These indicators are shown periodically to both learners and tutors so that each learner can see the effort they made compared to the rest of the group, and so that the tutor can evaluate the quality of the activity and the participation of each learner in the learning activity.

In the context of collaborative writing tasks, collaboration indicators are essential to measure the effectiveness of collaboration in collaborative work environments. In [16], these indicators include quantitative measures such as the number of words produced, the ac-

tive participation rate (measured by the number of interactions, modifications, or comments made), as well as qualitative criteria such as the relevance of the content produced by each group member. For example, the relevance of the content can be assessed through manual or automated analysis, based on the clarity of the ideas and their link to the collaborative work goal. Collaborative dynamics are also analyzed through indicators such as the frequency of interactions between participants, the response time between each interaction, and the analysis of communication patterns within the group. A group where exchanges are regular and balanced is generally more effective in completing the collaborative task. Besides individual contributions, it is also necessary to evaluate collaboration at the group level. Indicators such as the group engagement level, which measures the group's overall activity in terms of interactions, and the task completion rate, which reflects the group's ability to achieve the assigned objectives, are used for this. A group that works smoothly, with well-integrated and coherent contributions, produces a homogeneous and high-quality text, which can be evaluated by the document coherence indicator. Finally, the quality of collaboration is measured by indicators such as innovation and creativity in contributions and peer assessment, where participants evaluate each other's contributions. These evaluations make it possible to measure both the quality of the collaborative work and how each participant contributes to the innovation and creation of content.

In [96], the authors analyze the effectiveness of tracking indicators in the architecture of a collaborative system. They define several collaboration indicators from user interaction data, including interaction frequency, response time, relative participation, and the number of contributions per session. The calculation method is based on the automatic collection of activity traces (logs) generated during collaborative sessions. These data are then structured to extract relevant events (messages sent, replies, logins, document edits, etc.). Each indicator is calculated using quantitative functions:

Interaction frequency = total number of collaborative actions / session duration

Response time = average intervals between an action and its associated response

Relative participation = number of actions by a user / total number of group actions

Global engagement = weighted sum of text contributions, comments, and replies

To assess the relevance of these indicators, the authors use metrics from supervised classification (recall, precision, F-measure) by comparing the results automatically produced by the indicators to the expected ones (via manual annotations or expert judgments).

In [97], the authors propose an innovative approach for automatic collaboration analysis based on voice data recorded during group activities. The main goal is to calculate collaboration indicators from oral interactions, using techniques from trace analysis, natural language processing, and machine learning. The indicator calculation method starts with the collection of audio data during collaborative sessions, for example during an educational board game. These recordings are then automatically transcribed into text using speech recognition systems (ASR), making the corpus usable for the next steps. A lexical analysis is then

conducted on the transcripts to identify the most frequently used words and expressions by each group member. This analysis measures each participant's individual contribution to the verbal interaction. Then, the authors apply thematic modeling techniques, such as Latent Dirichlet Allocation (LDA) or Latent Semantic Indexing (LSI), to group words into themes representing the topics discussed. Finally, the relationships between words and their co-occurrences are represented as semantic graphs. In these word networks, centrality measures (such as betweenness or eigenvector centrality) are used to assess word importance in the conversation, revealing dominant themes or major contributions. The entire process aims to produce quantitative and qualitative indicators of oral collaboration, such as individual participation level, contribution diversity, thematic cohesion, or exchange structure. Although the results are promising, the authors highlight technical limitations, especially the need to improve automatic diarization (i.e., identifying speakers in the audio) and linguistic cleaning of transcripts to ensure more reliable indicator extraction.

In the CBHLE field, several studies aim at modeling and calculating indicators. These works can generally be grouped into two types. The first type concerns the proposal of specific indicators in an ad hoc way without focusing on the calculation method. The second type concerns the proposal of generic methods for designing and calculating indicators.

In [14], the authors present the DIAS system (Discussion Interaction Analysis System), designed to monitor and analyze group interactions in asynchronous discussions. The main goal is to provide a framework to evaluate engagement and the quality of collaborative interactions within online learner groups. The indicator calculation method relies on analyzing the traces left by participants during forum discussions. DIAS is based on several participation metrics to quantify and evaluate learner engagement in these asynchronous discussions. These indicators include frequency of contributions, message replies, and interactions between group members. The system also analyzes the quality of contributions using criteria such as topic diversity and relevance of interventions. Statistical techniques are applied to identify communication patterns among participants and detect collaborative dynamics, such as facilitator or observer roles, as well as moments of low or high engagement. The indicator calculation in the DIAS system provides feedback on collaboration quality and helps teachers identify areas for improvement, especially in online learning environments where interactions are often less visible and spontaneous than in synchronous settings.

In [15], the authors explore the evaluation of collaborative learning processes using a method based on systematic measurements. The main goal of their study is to develop and calculate indicators to evaluate collaboration and learner engagement in collaborative learning environments. The indicator calculation method is based on analyzing learners' interaction traces collected by the learning system. Collaboration indicators are defined based on several criteria, including learners' active participation (frequency and quality of contributions) and the distribution of tasks and roles within the group. The authors also use coordination measures to assess how learners support each other and exchange infor-

mation during the collaborative process. Performance indicators are then calculated based on group member interactions, considering aspects such as interaction duration, member responsiveness (response time), and task distribution. These indicators help detect imbalances in member contributions and evaluate collaboration fairness and the impact of interaction on achieving collective goals.

In [98], the authors propose a method to detect collaborative skills and calculate indicators in Moodle, an online learning platform. The method relies on using interaction traces collected on the platform, such as forum messages, assignment contributions, and interactions among learners in collaborative activities. The indicator calculation method is based on several collaboration dimensions. Participation indicators are calculated based on the number of contributions made by each learner and the time spent on the platform. Engagement is measured by reply frequency to messages or interaction with other group members, reflecting learner activity and responsiveness. The authors also define contribution equity indicators by measuring the distribution of actions among group members. This helps detect possible imbalances in participant involvement, which is crucial for evaluating group dynamics. Finally, the analysis of collaborative skills includes assessing social skills, such as the ability to support and encourage other learners, and communication skills, such as the clarity of exchanged messages. These indicators provide a detailed profile of learners' collaborative skills and help adapt teaching strategies accordingly.

For the automatic evaluation of collaboration between learners, [23] propose an intelligent multi-agent system called **SMAASA** (Multi-Agent System for Assisted Social Learning), which integrates fuzzy logic to analyze collaborative interactions in an online learning environment. Their goal is to provide personalized support based on the analysis of collaborative behavior.

Their approach is based on extracting a set of collaboration indicators from interaction traces generated by learners, such as forum posts, peer exchanges, or shared resource views. The main indicators used include:

- participation rate (number of messages posted or replies made in collaborative spaces);
- number of social interactions (replies to peer messages, direct exchanges);
- viewing of collective productions (such as wikis or shared documents);
- effective contribution to collaborative productions.

These indicators are often heterogeneous and imprecise. To address this, the authors use a fuzzy logic-based system, which translates raw numerical values into linguistic values (*low*, *medium*, *high*), thus facilitating qualitative evaluation of collaborative behavior. For example, a learner who posts a moderate number of messages may be classified in the *medium* participation category, while a more active learner would be considered *high*.

The combination of these indicators relies on a set of fuzzy inference rules, allowing a global collaboration level to be inferred. These rules are used by the analysis agent to build a dynamic diagnosis of each learner's collaborative profile. Then, the tutor agent uses this analysis to provide personalized feedback and recommendations (e.g., encouraging the learner to participate more in discussions or interact more frequently with peers).

2.5.2 Second category: Methods and tools for the design and calculation of collaboration indicators

A second category of work in the literature focuses mainly on developing methods and tools dedicated to the design and calculation of collaboration indicators. These approaches aim to formalize the processes of defining indicators, automate their extraction from activity traces, and provide means of analysis that help researchers or human tutors interpret them.

2.5.2.1 Main research projects contributing to indicator design and calculation

Several research projects from the European Network of Excellence Kaleidoscope have significantly contributed to the design and calculation of indicators in Computer-Based Human Learning Environments (CBHLEs). Among them, the ICALTS (Interaction and Collaboration Analysis supporting Teachers and Students' Self-regulation) [99], IA (Interaction Analysis) [100], and CAViCoLA (Computer-based Analysis and Visualization of Collaborative Learning Activities) [101] projects focused on observing and analyzing interactions between different actors involved in collaborative learning settings.

The ICALTS project focused on analyzing complex interactions in collaborative learning situations to support self-regulation for learners and teachers. It was extended by the IA project, which aimed to improve the interoperability of analysis tools and to propose standardized methodologies for implementing indicators in a mutual regulation context [100].

The CAViCoLA project, building on the work done in ICALTS and IA, focused on dynamically adapting the visualizations of indicators based on the user's needs. It proposed a common trace exchange format (based on an XML DTD) to allow interoperability between analysis tools from different CBHLEs [101].

From a complementary perspective, the DPULS (Design Patterns for collecting and analysing Usage of Learning Systems) project [84] aimed to capitalize on the experience of designers and teachers in analyzing CBHLE usage. It developed a library of reusable design patterns to support observation, tracking, and interpretation of learner activities through relevant indicators.

Finally, the REDiM project (Model-Driven Reengineering of CBHLEs), developed as part of the work by Choquet [102], focused specifically on analyzing traces from a CBHLE to automatically produce indicators for the teacher. These indicators aim to support pedagogical monitoring by highlighting significant learning behaviors and helping with decision-making.

These various projects share a common goal: to provide standardized methods, tools, and formats to facilitate the design, calculation, and use of collaboration indicators in digital education systems.

2.5.2.2 Model-Driven Bngineering-based method for calculating indicators from modeled traces

The work of Djouad et al. (2009) [18] focuses on engineering activity indicators from modeled traces in a Computer-Based Human Learning Environment (CBHLE). Their structured method allows the extraction and calculation of collaboration indicators based on learner actions and interactions. This approach is crucial for detailed analysis of learner activity dynamics and helps monitor participation throughout the learning process.

The authors stress the importance of having a precise trace model, which enables the calculation of indicators measuring key aspects of collaborative learning, such as engagement, participation, and interaction quality.

In their approach, two main steps are identified for calculating indicators:

- **Retrieval and restructuring of raw traces**: Traces are extracted from trace sources and transformed into modeled traces (primary traces), a key step to clean and prepare the data.
- **Application of transformation operators**: This phase consists of applying transformations on the primary traces to produce usable indicators in a collaborative context.

The calculation of an indicator follows four main steps, illustrated in Figure 2.20:

1. **Defining the trace model for the indicator**: A crucial step that defines the structure and transformations needed to obtain a relevant indicator.
2. **Transformation sequence**: Determining the process of transforming raw traces into the indicator model.
3. **Collecting observed elements**: Extracting specific learner actions or interactions.
4. **Executing transformations**: Applying transformations to produce a final indicator.

Djouad et al. (2009) [18] also proposed two additional operators to refine trace analysis:

- **Count**: This operator allows quantifying specific aspects of learner behavior, such as the number of collaborative interactions.
- **Filter**: The filtering operator selects a subset of observed elements based on specific criteria, such as time or type of tool used, allowing for targeted trace analysis.

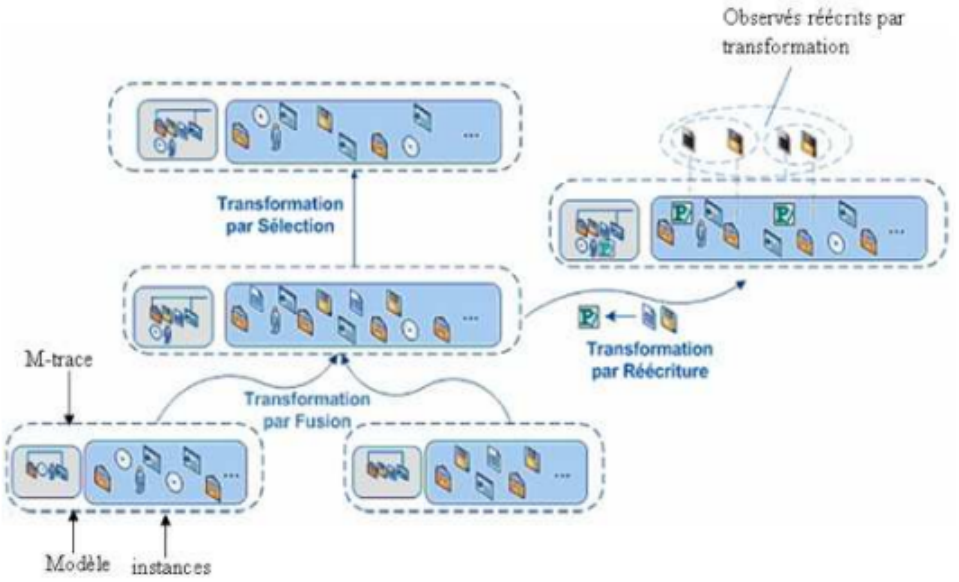

Figure 2.19: Indicators in CBHLEs

2.5.2.3 The GINDIC tool for calculating indicators from modeled traces

The *GINDIC* tool (Générateur d'INDICateurs), developed in [19], makes it possible to calculate collaboration indicators from activity traces left by users in a Computer-Based Human Learning Environment (CBHLE). This tool is designed to support designers and researchers throughout the different stages of creating, managing, and visualizing indicators.

GINDIC is based on a modular architecture that includes several key components for trace management and indicator generation:

- **Indicator base**: It centralizes all the indicators, defining their calculation logic, context, and display methods.
- **Construction system**: This component allows designers to define, in XML format, the logical rules for calculating indicators.
- **Rule-based system**: It applies logical rules to facts (activity traces and intermediate indicators) to produce new indicators.
- **Visualization module**: It provides graphical representation of calculated indicators, with dynamic and customized displays for each type of user (teacher, learner, tutor).

The process of calculating indicators in GINDIC takes place in several steps, as shown in Figure 2.20:

1. **Fact generation**: The *Fact Generator* regularly queries the Trace Base Management System (SGBT) to retrieve new traces generated on the CollaborativeECM platform, which it then transforms into facts.

2. **Rule definition**: Designers define the rules for calculating indicators, expressed in XML files, in the construction system.

3. **Rule application**: The *Rule-based System* uses the facts and rules to generate the indicators. This process includes combining data and applying the necessary logical transformations.

4. **Indicator visualization**: The results are presented in graphs adapted to user needs. The visualization module updates these graphs whenever a new indicator is calculated.

The figure below shows the general organization of the system for calculating indicators, illustrating the interaction between the different components of GINDIC.

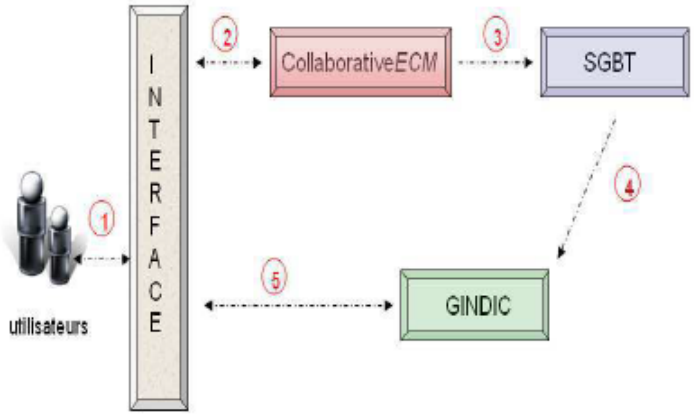

Figure 2.20: General organization of systems for calculating indicators from activity traces of a GED platform such as CollaborativeECM [19]

The GINDIC tool enables the calculation of indicators in a CBHLE by ensuring:

- **Automation and flexibility**: The generation of indicators is fully automated, with great flexibility to adapt to defined rules and available traces.
- **Customization of rules and visualizations**: Designers can customize calculation rules and visualization types based on pedagogical goals and user needs.
- **Accessibility of results**: Indicators are accessible in graphical form and can be viewed on different interfaces, including tablets and mobile platforms.

2.5.2.4 The UTL Language

The *UTL* language (User Trace Language) [20] was designed to formalize the data needed for trace analysis in a learning environment, while remaining independent of scripting languages

and representation formats. Its goal is to enable detailed analysis of learning traces by relying on flexible data structuring.

UTL aims at several essential objectives in trace analysis, including:

- **Capitalizing on know-how**: it helps to capitalize on methods and practices related to trace analysis and observation of learning sessions;
- **Defining observation needs**: with the help of a computer scientist, it allows specifying the technical means needed to acquire relevant data;
- **Structuring data**: UTL helps structure data from learning environments, from raw traces to indicators, independently of the original trace format.

The UTL language is based on two main categories of data from the DPULS project: *primary data* and *derived data*.

- **Primary data**: These data are directly extracted from the environment without prior processing. They are divided into three subtypes:
  - *Raw Datum (RD)*: data directly from learning systems (e.g., log files, videos, questionnaires, forum messages).
  - *Content Datum (CD)*: data voluntarily created by learning participants (e.g., assignments, reports, evaluations).
  - *Additional Datum (AD)*: contextual data related to the learning situation (e.g., metadata, taxonomies).
- **Derived data**: These data are calculated from primary data and/or other derived data. They include:
  - *Intermediate Datum (ID)*: data calculated from other data, with no direct educational meaning but necessary for building indicators.
  - *Indicator (I)*: observable calculated from primary or derived data, carrying educational meaning and reflecting the quality of interaction, activity, or learning.

The conceptual model of the UTL language is shown in Figure 2.21. It illustrates the relationships between the different categories of data and their role in structuring the information needed for trace analysis.

Each type of data in UTL is structured according to three facets:

- **Defining**: This facet models the observation need by specifying information such as the title, cardinality, and description of the data to observe.

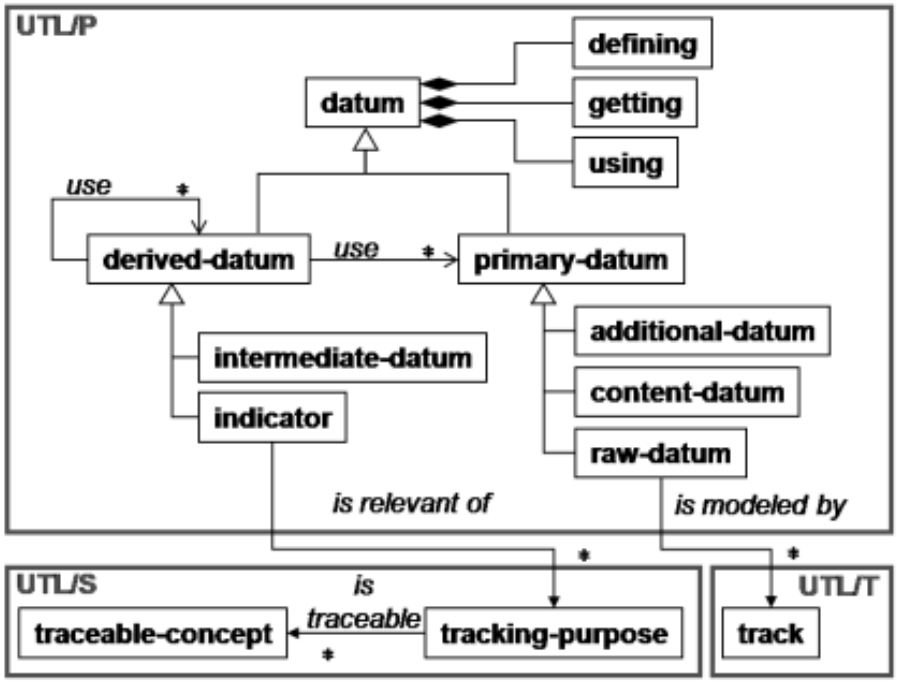

Figure 2.21: Conceptual model of the UTL language

- **Getting**: It describes how to obtain the data, including the source components, collection method, role of the actor (learner, tutor, teacher), tools used, and concrete examples of collected data.
- **Using**: This facet explains how the data are used, including the stored values, their format, and the educational context in which they are used (reengineering, regulation, evaluation, reflection).

2.5.2.5 The DCL4UTL language: an extension of the UTL language

The **DCL4UTL** (*Data Combination Language for UTL*) language was proposed as an extension of the UTL language to formalize the methods for calculating indicators from learning traces. Unlike approaches that provide predefined indicators, this approach is based on the expression of observation needs by the teacher/designer, which are then used to formalize the indicators.

This work follows a dual approach:

- a *language approach*, aiming at the formal description of methods for calculating indicators;
- a *hypothesis-oriented approach*, guiding the analysis of data from traces to produce indicator values.

The structure of the DCL4UTL language is detailed in Figure 2.22.

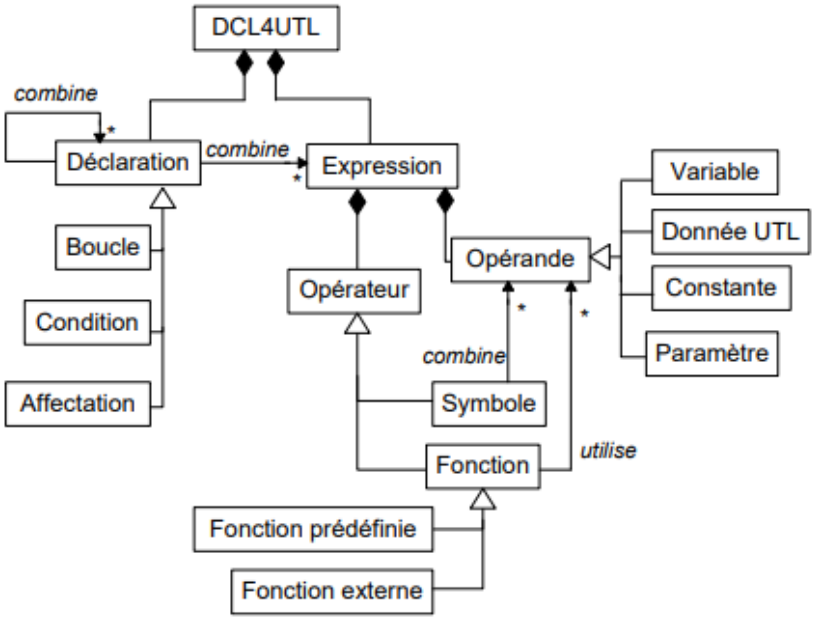

Figure 2.22: The structure of the DCL4UTL language

In this context, DCL4UTL allows:

- the formal description of observation methods using a dedicated grammar;
- the transformation and combination of data from traces;
- the capitalization of calculation methods as well as the results obtained.

An interpreter of the DCL4UTL language was developed and integrated into a trace analysis tool. This tool allows:

- the syntactic and logical validation of the produced descriptions;
- the execution of formalized methods;
- the consultation of the generated indicators and the data used for their calculation.

This language was used in the REDiM project, especially during an experiment conducted with teachers and tutors at the University of Maine. Observation needs were collected, indicators were formalized, then calculated in real time from traces from the *Hop3x* platform, to support tutors in their supervision activity.

Faced with the limits of traditional query languages (such as SQL or XQuery), especially in terms of reusability, integration of external functions, and dependence on the structure

of databases, DCL4UTL is a better-suited alternative for educational analysis of traces in a modeled framework such as that offered by UTL.

The **DCL4UTL** language was designed with the goal of allowing the combination of data from the UTL formalism to produce new derived data, such as indicators, while capitalizing on the calculation methods used. Inspired by traditional query languages (like SQL or XQuery), it adopts some of their data manipulation features while introducing specific extensions, including the possibility to store calculation methods and to integrate external functions.

DCL4UTL is integrated into the theoretical and technical framework of UTL. It specifically uses the *Method.Tool.Description* field of the *Getting* facet, enriched with two new fields:

- **Description.Text**, which describes in a human-readable way the method for calculating the derived data;
- **Description.DCL4UTL-formula**, which contains a formula that can be automatically interpreted for the calculation.

Another added field, **Getting.CalculationPeriod**, allows specifying the moment when the calculation is triggered, which enables real-time use. The calculation can thus be triggered by a system event, a timer, or an explicit user request.

The DCL4UTL language allows an analyst to formalize a method for combining the components of the *Getting.Component* facet, in order to obtain new data. It also acts on the *Using* facet of UTL data, which contains:

- a **Format** field, defining the representation format of the produced data;
- a **Data** field, intended to store the calculated instances according to this format.

2.5.2.6 A Multi-Agent System for the Calculation of Collaboration Indicators

In their work [24], El Mhouti, Erradi, and El Makhfi (2019) propose a multi-agent system dedicated to the calculation of collaboration indicators from modeled traces in an online learning environment (LMS). Their approach is based on a Model-Driven Engineering (MDE) method to formalize and automate the analysis of learner interactions.

The main objective is to extract relevant indicators to evaluate the quality of collaboration in online learning activities.

The system, shown in Figure 2.23, is based on the cooperation of several specialized agents, each with a specific role:

- Trace Modeling Agent: Converts raw logs generated by the LMS into structured traces, according to a specific metamodel.

- Semantic Annotation Agent: Enriches the traces with semantic annotations (for example, type of interaction, intention, or level of collaboration), using a pedagogical ontology.
- Filtering Agent: Selects relevant events by removing passive or insignificant actions (such as simply viewing a resource).
- Indicator Calculation Agent: Applies calculation formulas to the filtered data to produce quantitative indicators.
- Profiling Agent: Interprets the obtained indicators by applying thresholds or fuzzy rules, in order to generate individual collaboration profiles.

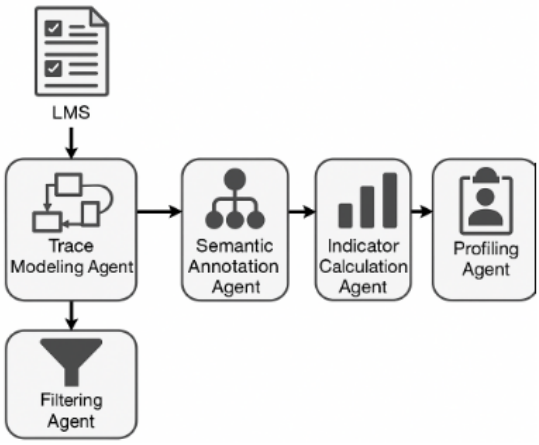

Figure 2.23: Multi-agent architecture for semantic analysis and indicator calculation [24]

Several indicators are extracted from the analyzed traces. Some examples are shown below:

- Collaborative Participation Rate:

$$\text{Participation} = \frac{\text{Number of collaborative actions}}{\text{Total number of actions}}$$

- Social Interaction Density:

$$\text{Density} = \frac{\text{Number of peer replies}}{\text{Total number of exchanges}}$$

- Temporal Regularity of Contributions:

$$\text{Regularity} = \text{Standard deviation of intervals between collaborative actions}$$

- Contribution to Collective Production (e.g., on a wiki or shared document):

$$\text{Involvement} = \frac{\text{Number of significant contributions}}{\text{Number of document sections}}$$

These indicators can be adapted to different types of collaborative activities thanks to the flexibility of the system and the possibility of defining specific models. The collaboration indicators generated by the system make it possible to:

- Categorize learners based on their level of collaborative involvement,
- Identify atypical profiles, such as passive or, on the contrary, dominant learners,
- Provide personalized feedback to tutors and learners to support more effective pedagogical guidance.

2.6 Discussion and Scientific Positioning

Our approach is part of the ongoing efforts initiated by earlier works on the calculation of indicators from modeled traces in CBHLEs. It shares with Djouad et al. (2009) [18] a rigorous modeling of traces, a separation between raw and transformed traces, and the use of processing operators (filtering, counting, selection). However, it differs by introducing a complete process for designing indicators, accessible to teachers through a user interface, aimed at supporting indicator definition rather than just computation. It also relies on an MDA architecture that enhances the reusability of models and proposes a multi-level analysis (individual, group, community), along with a detailed typology of interactions (collaborative/non-collaborative, active/passive).

Similarly, GINDIC (Gendron et al., 2010) [19] provides a modular framework for dynamic indicator calculation using logical rules. While our work reuses the idea of modularity and automation, it goes beyond GINDIC by integrating the upstream phase of indicator design, without the need for technical languages like XML. Our model-driven approach enables the creation of configurable indicators, exportable across various educational contexts, with greater attention to the hierarchical structuring of analyses and the semantics of interactions.

The UTL language [20] and its extension DCL4UTL [22] introduce templates and a formal grammar to structure trace analysis and formalize indicator computation. Our approach aims to be more operational: while UTL proposes conceptual patterns (Defining, Getting, Using) and DCL4UTL an interpreted language, we provide a user-friendly interface and a system for indicator generation. This choice enhances teachers' autonomy and the concrete reusability of indicators, with a specific focus on collaborative phenomena through a hierarchical structure and enriched typology.

Finally, although the multi-agent system by El Mhouti et al. (2019)[24] shares several points with our work — such as the use of a trace metamodel, a model-based approach (MDE), and the goal of generating understandable indicators — our system stands out by its centralized and modular architecture. It bases on a tool that is easy to use, even for users without technical skills. Moreover, it clearly organizes levels of analysis (individual, group, community), refines measurements through an interaction typology, and allows for easy reuse of indicators. While their approach relies on distributed intelligent agents, we focus on simplicity, customization, and pedagogical adaptation.

Therefore, our contribution is distinguished by its goal: making the design of collaboration indicators more accessible, clear, and adaptable. It builds on existing work while offering a practical, user-centered solution focused on multi-level collaboration analysis.

The comparison presented in Table 2.2 is based on four key criteria that distinguish existing approaches to collaboration indicator modeling and computation in e-learning systems:

- **Support for Indicator Design and/or Calculation**: This criterion evaluates whether the proposed method provides tools or mechanisms that assist either in designing collaboration indicators (i.e., defining what to measure and how) or in calculating them based on learning traces. Some approaches focus solely on computation, while others also support the upstream phase of indicator specification.

- **Indicator Type**: This refers to the nature or level of indicators handled by each approach. It may include low-level interaction indicators (e.g., message count), behavioral patterns, socio-cognitive indicators, or composite indicators combining multiple dimensions (e.g., participation, equity, and reactivity).

- **Used Approach**: This column describes the underlying approach adopted for modeling and computing the indicators. It may involve model-driven engineering (MDE/MDA), trace-based analysis, machine learning, rule-based systems, natural language processing (NLP), or hybrid frameworks.

- **Openness of the Implementation System**: This criterion distinguishes between open systems (which allow extension, integration, or user-driven configuration) and closed systems (which are pre-defined and cannot be easily modified or extended by end users). Open systems offer more flexibility, particularly for teachers and researchers who wish to define their own indicators or adapt existing ones.

Work	The suggested method supports the design /calculation of the indicators	Indicator type	Used approach	The implementation system is open or closed
[20]	The design	Indicators in CEHL	• Designing of indicator in a form similar to a design pattern	opened

[18]	The calculation	Indicators in MOODLE platform	• Oriented Model Transformation. • based on MDE. • Using an m-trace-based system(self-developed).	Closed
[19]	The design and calculation	Collabora-tion indicators	• Using a collabora-tion indicator pattern for the design. • Using a Computation method oriented Artificial intelligence and based on rule-logic. • Using a rule-based system(self-developed)	Opened
[22]	The design and calculation	Indicators in CEHL	• Enrich UTL with formal language to formally describe the calculation method. • Indicator computation using a DCL4UTL interpreter (self-developed) integrated into a trace analysis tool.	Opened

[23]	The calculation	Collaboration indicators in e-learning systems	• Oriented Model Transformation. • Basing on MDE. • Using A multi-agent system (self-developed) and an m-trace based system(KTBS).	Closed
Our work	The design and calculation	Collaboration indicators in an e-learning systems	• The design is based on the DCIN Model(self proposal) • The calculation is oriented model-transforma- tion based on MDE. • The sequences of transforma- tions are automatiquelly generated using DCIN-AGSET(self developped).	Opened

Table 2.2: Comparison between related works and our proposition regarding the design and calculation of collaboration indicators in e-learning systems.

In the Table 2.2 above, we compare existing works with our own proposal regarding the design and calculation of collaboration indicators in computer-based human learning environments (CBHLEs). Each study approaches the issue from different angles, using various methodologies and implementation systems.

In the case of [20], the proposed method focuses mainly on the design of indicators in collaborative learning environments (CEHLs), using a design model inspired by design patterns. This approach is open, allowing some flexibility for adaptation. On the other hand, [18] is characterized by its model transformation approach (Model-Driven Engineering, MDE), applied in the MOODLE platform, but the implementation of the system is closed, limiting the possibility of extension or reuse in other contexts.

In [19], the authors propose a combined method for designing and calculating collaboration indicators using a design model based on artificial intelligence rule logic. This approach stands out for its self-developed rule-based system and remains open, which allows for greater adaptability. Conversely, [22] focuses on enriching the UTL (Usage Tracking Language) by integrating a formal language to describe calculation methods. This system is also open, and its indicator calculation relies on a DCL4UTL interpreter integrated into a trace analysis tool.

The methodology adopted in [21] for calculating collaboration indicators remains ad hoc, without a defined structure, and the system implementation is closed, limiting the reusability of the proposed solutions. [23] proposes an approach based on model transformation and MDE, using a multi-agent system to evaluate collaboration indicators. This system, although based on the KTBS platform (Kernel Trace-Based System), remains closed and thus not extensible.

Our own work, presented at the end of the table, is distinguished by an integrated approach to designing and calculating collaboration indicators. We propose a design model based on the DCIN (Design Collaboration Indicators), which we developed, as well as a calculation method based on model transformation following an MDA approach. In addition, to automate the process, we have developed the DCIN-AGSET system, which automatically generates the transformation sequences needed to obtain collaboration indicators. Our system is open, which ensures great flexibility and reusability in various collaborative learning contexts.

In summary, our work is distinguished by a unique combination of modular design, automated calculation, and reusability, which helps overcome the limitations of previous works while offering a flexible and user-friendly system for teachers without requiring advanced computer skills.

Compared to previous works, our proposal focuses specifically on the design and calculation of collaboration indicators in CBHLEs. Thus, our approach can be summarized as follows:

- The first aspect focuses on the calculation of collaboration indicators in CBHLEs, regardless of any specific platform. To this end, we propose to apply a model transformation approach combined with a process based on Model-Driven Architecture (MDA) to derive collaboration indicator models.

- The second aspect is centered on the design of collaboration indicators in CBHLEs. To do this, we introduce a formal model that facilitates the design of valid and meaningful collaboration indicators, according to the teacher's specific observation needs.

- Based on the application development process supported by MDA, the calculation of collaboration indicators can be seen as a model transformation process. In this ap-

proach, the trace model undergoes a sequence of transformations, eventually resulting in the collaboration indicator model. To address the need for automation in the calculation of collaboration indicators and eliminate the need for specific computer skills—usually beyond the reach of non-technical teachers—we propose to automate the generation of the transformation sequences.

- The third aspect concerns how to obtain the transformation sequences. To this end, we propose a system that ensures the automatic generation of these transformation sequences, which will then be applied in the m-trace-based system to produce the final indicator model.

2.7 Conclusion

This chapter has laid the conceptual and technical foundation necessary for understanding our research work. We introduced essential concepts related to traces in CBHLEs, highlighting the challenges of trace collection, the limitations of raw traces, and the value of moving toward formal trace modeling. The approach of Model-Based Trace System (MBTS) was presented as a relevant answer to the need for fine-grained interaction analysis.

We also discussed existing methods and languages for trace analysis and indicator calculation, with a focus on collaboration indicators. Through the review of ad hoc approaches and model-driven engineering methods, we highlighted existing solutions and their limitations.

Finally, by presenting several significant works, such as the DCL4UTL language, the GINDIC tool, and multi-agent systems for indicator calculation, we prepared the ground to present our own scientific contribution.

Part II

Design, Implementation and Experimental Results

Chapter 3

MDA-based Approach for Computing Collaboration Indicators in CBHLEs

3.1 Introduction

In Computer-Based Human Learning Environments (CBHLEs), the evaluation of collaboration between learners increasingly relies on the use of interaction traces. However, the diversity of platforms, learning tools, and modes of interaction generates a strong heterogeneity in the structure and semantics of traces, making systematic and reliable analysis of collaborative behaviors difficult. Faced with this challenge, the Model-Driven Architecture (MDA) approach appears as a powerful solution to standardize and automate the process of computing collaboration indicators.

This chapter proposes a methodological approach based on the MDA paradigm, aiming to define, transform, and use learning traces to produce collaboration indicators independently of platforms. We first introduce the concept of the Trace Platform Independent Model (Trace PIM), which offers an abstract and generic modeling of collected traces. We then address the addition of trace constraints, allowing to ensure the quality and relevance of the data used.

Next, we present the Collaboration Indicator Platform Independent Model (Indicator PIM), which structures the definition of collaboration indicators in a way that is independent of any specific platform. The approach is based on clearly defined transformation modes, which make it possible to move from abstract traces to computable indicators. We also detail the transformation contracts, which ensure the semantic consistency of the model transformations.

3.2 Methodological Approach

Model-Driven Architecture (MDA) is one of the most emblematic approaches of model-driven engineering, widely recognized for its ability to transform software development by focusing mainly on abstract models rather than on the system's source code. This approach significantly reduces the efforts needed for the design, development, and maintenance of systems, by promoting model reuse and allowing flexibility in the face of technological changes. In fact, all the work is structured around the transformations of models that prescribe the structure and behavior of the system, and not on the systems themselves. This leads to reduced complexity and development costs, while ensuring the sustainability of the proposed solutions.

With this in mind, our approach is based on MDA, and particularly on the idea that the computation of collaboration indicators can be seen as a sequence of transformations applied to models. We start from a trace model (T-PSM) and end up with a collaboration indicator model (CI-PSM). This transformation process is designed so that, once the collaboration indicator model is created, it is possible to obtain the final value of the indicator with a simple function call. One of the strengths of this approach is that it provides reliable and reproducible results without the need to write specific source code each time.

According to the principles of MDA, each Platform-Specific Model (PSM) must conform to a Platform-Independent Model (PIM). In other words, the PIM describes a system in an abstract way, independently of any specific technology or platform. In our approach, each trace model intended for a CBHLE (T-PSM) must conform to a generic trace meta-model (T-PIM) that describes the structure of traces independently of any platform. Similarly, each collaboration indicator model (CI-PSM) must conform to a collaboration indicator meta-model (CI-PIM), thus ensuring the universality and portability of the proposed indicators.

MDA makes it possible to standardize and harmonize development steps, which greatly facilitates the management of transformation processes. The first step in this approach is to create platform-independent models (PIM) and enrich them so they match the specific needs of designers. The second step is to choose the implementation platform and generate platform-specific models (PSM), a crucial step for implementing the solutions. In our proposal, we offer a platform-independent trace model (T-PIM) that acts as a meta-model applicable to all kinds of traces, whether collaborative or non-collaborative. This T-PIM model allows the formalization of a structured and consistent approach for analyzing traces generated in a learning context. In parallel, we have developed a platform-independent collaboration indicator model (CI-PIM), which serves as a meta-model for indicators intended to be used in CBHLEs, while ensuring independence from the underlying platform.

Both models (T-PIM and CI-PIM) conform to the Meta Object Facility (MOF) standard of the OMG [23], a widely used standard for ensuring the definition and exchange of metadata between different tools and systems. The transformation between these models is carried out

through a sequence of operations that takes a T-PSM model as input and produces a CI-PSM model as output. This transformation process makes it possible to move from a platform-specific trace model to a collaboration indicator model in an automated way, ensuring the consistency and efficiency of the indicator generation process.

These transformations must be written in a precise transformation language (transformation meta-model) that makes it possible to formalize the different steps of the process. In our case, we chose a simplified version of this language, which will then be translated into the transformation language implemented in the trace-based system. Thus, each transformation step can be carried out transparently, while respecting the rigor and flexibility of the MDA principles. Finally, to ensure that the transformations proceed correctly and that the results meet expectations, our collaboration indicator model (CI-PIM) is associated with OCL (Object Constraint Language) constraints. These constraints make it possible to require certain conditions and specifications that cannot be expressed simply using the UML language, thus ensuring the validity of the transformation process and the reliability of the obtained indicators.

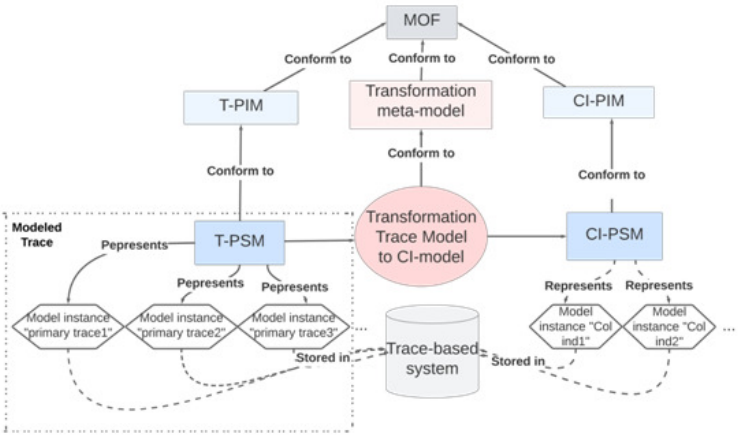

Figure 3.1: Approach based MDA for the computation of collaboration indicators

3.2.1 Trace Platform Independent Model

The proposal of the Trace Platform Independent Model (T-PIM) constitutes a fundamental approach to model collaborative and non-collaborative traces in a general way, independent of any specific learning platform. This model makes it possible to standardize and harmonize traces, thus enabling their use in various contexts and across different platforms without worrying about the technical particularities of each one. In other words, the T-PIM ensures abstraction from the details related to the specificities of each learning environment. Any Platform-Specific Trace Model (T-PSM) proposed must strictly conform to this meta-model to ensure the interoperability and consistency of data between different platforms. As shown in Figure 3.2, a T-PSM represents a primary trace, which is a detailed representation of a

particular CBHLE, while respecting the structures defined by the T-PIM. This relationship between the T-PIM and the T-PSM ensures that the data collected and analyzed across different systems are compatible, thus offering opportunities for comparative analysis and generalization of results.

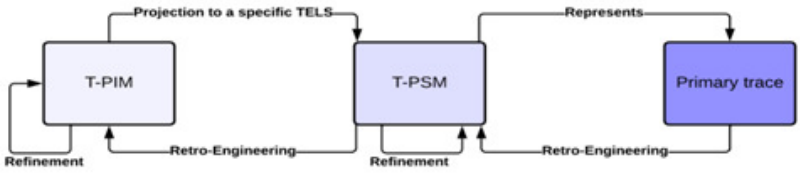

Figure 3.2: The relationship between trace levels.

Figure 3.3 presents the T-PIM model in the form of a UML class diagram. In this model, we distinguish thirteen main classes, each having specific roles in the representation of collaborative and non-collaborative traces, independently of any learning platform used. These classes are designed to structure the information related to traces and allow for coherent and flexible modeling that can adapt to various CBHLEs. Each of these classes interacts with the others to provide a complete and detailed view of the online learning process. The classes are defined and detailed in Table 3.1, where we specify the attributes, relationships, and responsibilities of each class within the framework of trace modeling. This UML class diagram thus makes it possible to visualize the structure and relationships of the elements of the T-PIM, facilitating the integration and extension of the model for different applications in the field of collaborative learning.

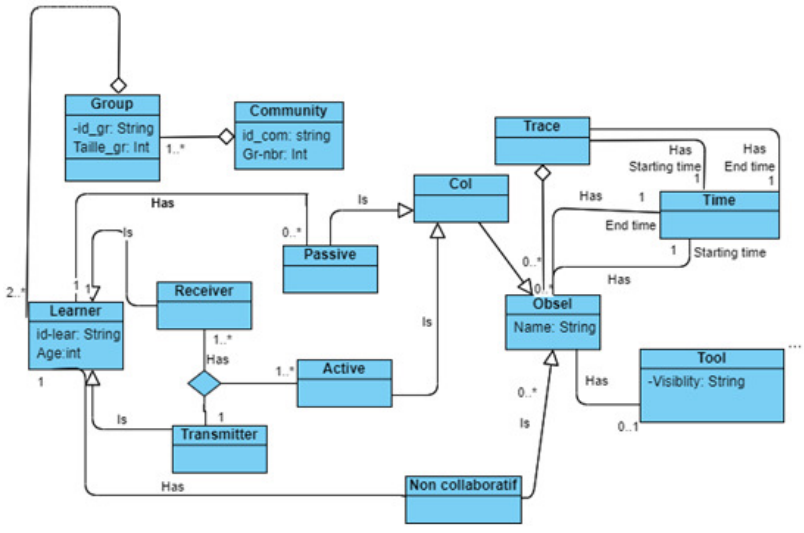

Figure 3.3: Meta model represents the Trace platform independent model

In the T-PIM model, we distinguish a total of 13 classes. The details of these classes and their characteristics are specified in 3.1, thus allowing for a thorough understanding of their function and their contribution to the overall model.

Class	**Description**
Obsel	The class (Observed Element) represents the specific action or event that we want to observe within the system, such as sending a message, viewing a wiki, authentication (login), or carrying out a specific task by a user. It is the starting point of the analysis, allowing the capture and tracking of user behavior within the learning system. These observed elements can be used to generate indicators to measure and analyze learner interaction with the platform, in order to improve the learning experience and guide pedagogical actions.
Col	Represents a collaborative action, such as participation in a poll, where several individuals interact together in order to gather information or make collective decisions. This action involves active collaboration between participants, often in a shared online environment or in a collaborative learning context.
Non-collaboratif	Represents a non-collaborative action, such as viewing a course, where the user interacts independently with the content without needing to exchange or cooperate with other participants. This can include actions like reading documents, watching videos, or browsing through online learning modules in an isolated manner.
Active	Represents a sending action or an active participation, where the user takes the initiative to interact with the system by sending information or actively contributing to a process, such as submitting an answer, posting a message, or interacting with content in real time.
Passive	Represents a passive or viewing action, where the user interacts with the system in a receptive way — for example, by viewing forum posts, consulting a wiki, or accessing a collaborative segment without making any changes or actively intervening.
Learner	Linked to the Obsel class, this class represents the user who performs the observed action. It helps link each action or event to the person responsible for it, giving more context for analyzing interactions in the system.

Transmitter	Represents the actor involved in the action in a context of active collaboration. This is a learner who interacts dynamically and is actively engaged in a group learning activity. This actor plays a key role in carrying out the collaborative action, contributing to the group goal while interacting with others.
Receiver	Represents the non-empty set of receiving learners, meaning those directly concerned by the collaborative observed action. This set includes individuals involved in receiving or interacting with a collaborative action, like viewing a shared document, receiving a message in a forum, or participating in a group activity. These learners are essential in the collaboration process, as their engagement or response can influence how well the collective activity goes.
Time	Closely tied to the observed element, it represents the exact moment the action was performed. It helps place interactions in a timeline and understand the timing dynamics of observed activities.
Tool	Linked to the observed element, it represents the tool that supports and enables interaction within the system. This tool can be public (available to all users) or private (restricted to a specific zone or group). Public tools encourage collaboration and sharing across the whole community, while private tools offer secure and personalized spaces for each user or group, such as confidential chats, custom resources, or targeted activities. These tools are key to shaping interactions and improving the user experience with features tailored to both collaborative and individual contexts.
Group	Represents a set of learners with shared learning goals or a group formed around a specific activity, such as a project, course, or collaborative task. This group can be created based on skill level, learning preferences, or roles in a collaborative learning environment. Group members interact, share resources and knowledge, and work together to reach common learning goals, supporting active and collaborative learning.
Community	Represents a set of interconnected learner groups within the same collaborative learning environment, where each group has its own goals but is also part of a bigger structure. A community can include several groups with shared interests or themes, helping them exchange knowledge, cooperate, and support each other. It encourages collective learning where resources and best practices are shared among groups, boosting engagement and educational innovation.

Trace	Represents a trace in a CBHLE a detailed record of a learner's interaction with the learning platform.

Table 3.1: The T-PIM classes description

In this meta-model, we propose the following rules, which help to better structure and analyze interactions within the e-learning system:

- Each trace has a specific start and end date. It consists of a series of observed elements called "obsel," each representing an action or a significant event in the interaction of learners with the learning platform.
- An obsel, representing a specific action or event, can be collaborative, such as sending a message or participating in a forum, or non-collaborative, such as an individual action like logging out or viewing a course without interacting with other learners.
- Each obsel can be associated with a specific tool that supports the observed action. This tool can be public (accessible to all users) or private (reserved for a specific private area, such as a group of learners).
- Each obsel is associated with specific moments, with a start and end date and time, which allows contextualizing the action in time and tracking the evolution of learner interactions over time.
- A collaborative obsel can be passive if the action consists of simply consulting content or information, such as reading a document or consulting in an online discussion. On the other hand, a collaborative obsel is called active when it involves direct participation in a collective learning process, such as editing a wiki, sending a message, or responding to a questionnaire.
- A passive collaborative obsel is typically associated with a single learner, who acts in a receptive manner and benefits from the information without taking any direct or engaging action. In contrast, an active collaborative obsel is often linked to multiple learners: one sender, who initiates the action, and a set of receivers, who interact or respond to the action dynamically.
- A group is composed of two or more learners who interact and collaborate together in a learning context. The group can be seen as a collective unit of work, and its members may share common learning objectives, as well as participate in various activities depending on their role and interactions.
- A community, on the other hand, represents a set of groups that may work on common projects or interact within a larger platform. A community can be dynamic, with exchanges between groups that promote collective learning and the enrichment of shared

knowledge. This structure facilitates the management of learner groups at different levels and helps better understand interactions and dynamics within the learning platform.

3.2.2 Trace Constraints

To ensure the reliability of the collected data and avoid any erroneous trace collection related to potential incompleteness of the T-PIM (Trace-based Platform Independent Model), we have enriched our T-PIM model by associating it with a set of formal constraints expressed in OCL (Object Constraint Language). These constraints allow us to specify the structural and temporal rules that the elements of the model must adhere to, such as the mandatory presence of obsels, the validity of start and end dates, or the consistency of collaborative categories. By integrating OCL constraints directly into the T-PIM model, we ensure better consistency of the generated traces. This allows us to detect potential errors due to incomplete or incorrect modeling in advance. We thus avoid issues related to erroneous trace collection, ensuring the reliability of the data used in subsequent analyses. The complete set of these constraints is presented in detail in the Table 3.2 below , which associates each rule with its description and formalization in OCL.

Description	**OCL Constraint**
A Trace cannot be empty (it must contain at least one Obsel).	`context Trace inv mustHaveAtLeastOneObsel: self.obsels->size() >= 1`
An Obsel must have a non-empty name.	`context Obsel inv nameNotEmpty: not self.name.oclIsUndefined() and self.name.size() > 0`
A trace can contain at most a finite number of obsels, denoted by m.	`context Trace inv maxObsels: self.obsels->size() <=` m
A Trace must have a start and end date.	`context Trace inv mustHaveTime: not self.time.oclIsUndefined()`
The start date must be earlier than the end date.	`context Time inv startBeforeEnd: self.startingDate < self.endingDate`
A Trace cannot start in the future.	`context Time inv startNotInFuture: self.startingDate <= OclDateTime::now()`

A Trace cannot be modified after its end date.	`context Trace inv cannotModifyAfterEnd: self.time.endingDate >= OclDateTime::now()`
An Obsel must have a start and end date.	`context Obsel inv mustHaveTime: not self.time.oclIsUndefined()`
The start date of an Obsel must be earlier than its end date.	`context Obsel inv obselStartBeforeEnd: self.time.startingDate < self.time.endingDate`
An Obsel cannot start before the Trace that contains it.	`context Obsel inv obselWithinTrace: self.time.startingDate >= self.trace.time.startingDate and self.time.endingDate <= self.trace.time.endingDate`
An Obsel must be categorized as Active, Passive, or Non-Collaborative.	`context Obsel inv MustHaveCategoryConstraint: self.oclIsTypeOf(Active) or self.oclIsTypeOf(Passive) or self.oclIsTypeOf(NonCollaborative)`
A Collaborative Obsel must be either Active or Passive.	`context Collaborative inv collaborativeMustBeActiveOrPassive: self.oclIsTypeOf(Active) or self.oclIsTypeOf(Passive)`
A Trace cannot be deleted after its end date.	`context Trace inv cannotBeDeletedAfterEnd: self.time.endingDate >= OclDateTime::now()`
An Obsel can have 0 or 1 Tool.	`context Obsel inv obselHasAtMostOneTool: self.tool->size() <= 1`
A Tool must be either 'public' or 'private'.	`context Tool inv VisibilityConstraint: self.visibilité = 'public' or self.visibilité = 'private'`
An Obsel must be either Collaborative or Non-Collaborative, but not both.	`context Obsel inv mustBeEitherCollaborativeOrNonCollaborative: (self.oclIsTypeOf(Collaborative) or self.oclIsTypeOf(NonCollaborative)) and not (self.oclIsTypeOf(Collaborative) and self.oclIsTypeOf(NonCollaborative))`

If an Obsel is Collaborative, it must be Active or Passive.	`context Obsel inv collaborativeMustBeActiveOrPassive: self.oclIsTypeOf(Collaborative) implies (self.oclIsTypeOf(Active) or self.oclIsTypeOf(Passive))`
An Obsel can also be Non-Collaborative, but not just Collaborative without being Active or Passive.	`context Obsel inv collaborativeOrNonCollaborative: (self.oclIsTypeOf(Collaborative) and not self.oclIsTypeOf(NonCollaborative)) or (self.oclIsTypeOf(NonCollaborative) and not self.oclIsTypeOf(Collaborative))`
A Learner can be a Receiver, Transmitter, or both.	`context Receiver inv inheritsFromLearner: self.oclIsTypeOf(Learner) context Transmetter inv inheritsFromLearner: self.oclIsTypeOf(Learner)`
A Collaborative Active Obsel must have a Transmitter and at least one Receiver.	`context Obsel inv CollaborativeActiveObselConstraint: self.oclIsTypeOf(Collaborative) and self.oclIsTypeOf(Active) implies (self.transmetter->size() = 1 and self.receiver->size() >= 1)`
A Non-Collaborative Obsel has a Learner who is neither a Transmitter nor a Receiver.	`context Obsel inv NonCollaborativeLearnerConstraint: self.oclIsTypeOf(NonCollaborative) implies (self.learner.oclIsTypeOf(Learner) and not self.learner.oclIsTypeOf(Transmetter) and not self.learner.oclIsTypeOf(Receiver))`
A group consists of at least two Learners.	`context Groupe inv GroupLearnerConstraint: self.learner->size() >= 2 and self.learner->forAll(l | l.oclIsTypeOf(Learner))`
Each group must belong to exactly one community.	`context Groupe inv CommunityConstraint: self.community->size() = 1`

Table 3.2: OCL Trace Constraints

3.2.3 Collaboration Indicator Platform Independent Model

The proposed Collaboration Indicator Platform Independent Model (CI-PIM) aims to provide a flexible and universal approach to modeling collaboration indicators, regardless of the specific characteristics of the platform used. This model thus enables better analysis and tracking of collaborative interactions in an online learning context, while ensuring interoperability between different platforms. Any Platform Specific Model for Collaboration Indicators (CI-PSM) to be obtained must strictly comply with the rules and structures defined by this meta-model to ensure the consistency of analyses across various learning environments. As illustrated in Figure 3.4, a CI-PSM concretely represents collaboration indicators in a specific CBHLE. It is a detailed projection that adapts the principles of the CI-PIM to the particular context of the CBHLE in question.

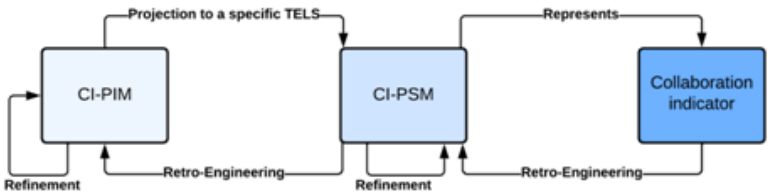

Figure 3.4: The relationship between collaboration indicator levels.

Figure 3.5 presents our CI-PIM in the form of a UML class diagram. This diagram illustrates the different classes and their relationships within the collaboration indicator meta-model(CI-PIM), thus providing a clear visual representation of the structure and key elements of the model. Each class in the diagram is designed to capture specific aspects of collaborative interactions, whether they concern learners, the tools used, the observed actions, or the collaboration groups.

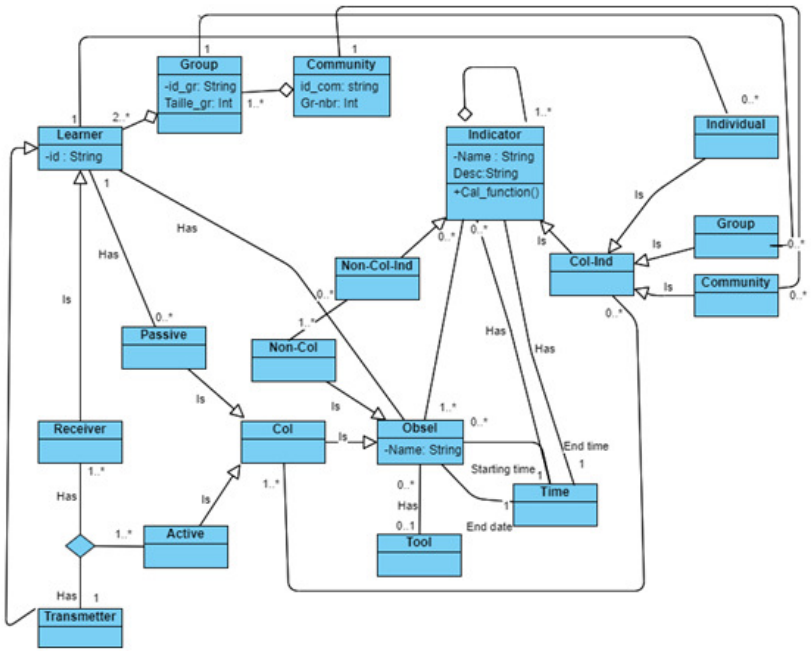

Figure 3.5: Meta model represents the Collaboration Indicators Platform Independent Model

For clarity and to avoid repetition, only the classes not mentioned in the T-PIM will be defined in the Table 3.3. These classes, specific to the collaboration indicators model, complete those already present in the T-PIM in order to define what a collaboration indicator is in a collaborative learning system.

Class	Description
Indicator	An *Indicator* represents a measurable element or variable used within a CBHLE to observe, evaluate, or monitor aspects of the learning process. It can provide insights into user behavior, cognitive engagement, learning progress, participation, or interaction patterns. Indicators serve as essential tools for diagnosis, feedback, and decision-making, helping educators and systems adapt to learners' needs and improve the effectiveness of learning activities.
Col-Ind	Represents a collaboration indicator in a CBHLE. This indicator measures the intensity and quality of collaboration among learners in a collaborative environment. It can assess elements such as the frequency of interactions, active participation in group discussions, resource sharing, or contribution to group projects. Collaboration indicators are essential to track learners' engagement, identify collaborative or individual behaviors, and adjust pedagogical approaches. They also allow teachers to analyze group dynamics and intervene in a targeted manner if needed.
Non-Col-Ind	Represents other types of indicators in a CBHLE. These indicators relate to non-collaborative aspects such as individual performance, personal engagement, progress in modules, or success in assessments. They assess criteria like time spent on the platform, login frequency, or test results. These indicators provide an overall view of learners' behavior and help identify those who need more support. They can be combined with collaboration indicators for a more complete analysis of learning activities.
Individual	Represents a collaboration indicator specific to a learner. It measures their involvement and interaction in collaborative activities, considering their participation in discussions, contributions to projects, and engagement in using shared resources. It allows for the analysis of individual behavior in a collaborative context, identifying strengths and areas for improvement, and adapting pedagogical strategies to promote more effective collaboration.
Group	Represents a collaboration indicator specific to a group of learners. It evaluates the group's collaborative dynamics through elements such as the distribution of contributions, the frequency and quality of exchanges, resource sharing, and coordination in carrying out joint tasks. It helps identify imbalances, analyze cohesion and the effectiveness of teamwork, and adjust teaching practices to strengthen collaboration.

Community	*Community* represents a collaboration indicator within a community of learners in a CBHLE. This indicator measures the engagement, interaction, and collaboration among members of a community, which may include several learner groups or individuals with common learning goals. It assesses how members interact through discussions, resource exchanges, project collaborations, and participation in collective events. The community collaboration indicator considers elements such as the frequency of interactions, the diversity of contributions, and the impact of exchanges on collective progress. It can also reflect the community's ability to support each other, share ideas, and solve complex problems together. This information not only helps analyze the overall effectiveness of the community but also strengthens bonds and creates a more dynamic, inclusive, and productive collaborative learning environment.

Table 3.3: Description of the specific classes in CI-PIM

In this meta-model, we propose the following rules to structure and refine the evaluation of collaborative activities in CBHLEs:

• An indicator in a CBHLE can be elementary, representing a simple measure, or composite, combining several indicators to provide a more complete analysis of learners' actions or performances.

• An indicator is calculated within a well-defined time interval to ensure its relevance and accuracy. Indicators can be:

Non-collaborative, associated with non-collaborative obsels. These indicators measure actions that do not require interaction with other learners, for example, the R-time indicator, which represents the average time a learner takes to complete a course or an individual activity, without direct interaction with other members of the platform.

Collaborative, associated with collaborative obsels. These indicators measure collaboration actions, where learners actively interact to carry out a common task. For example, the indicator of a learner's participation rate in collaborative activities within their group, or the contribution rate in group projects, thus evaluating the actual involvement of the individual in group work.

• A collaboration indicator can be calculated at different levels of analysis, depending on the object of study:

Individual: This makes it possible to measure the involvement of a single learner in collaborative activities. For example, the communication indicator (such as the frequency of participation in group discussions) can be measured for a single learner to assess their contribution to social interaction in a collaborative learning environment.

Group: The indicator can also be used to analyze the dynamics of a group of learners. For example, the collaboration rate indicator within a group measures the effectiveness of interactions among group members to accomplish a collective project.

Community: This level makes it possible to evaluate collaboration on a larger scale,

for example, the communication indicator of a learning community, which could reflect the collective activity in a forum or an exchange platform. This allows for an overall view of participation within a learning community, identifying general collaboration trends.

3.2.4 Transformation mode

Our model transformation process is a unidirectional, exogenous, horizontal transformation of the M2M (Model-to-Model) type, where:

- It is carried out in a single direction, from T-PSM to CI-PSM, thus allowing a transition from one model abstraction to another, while respecting the requirements of the different representation levels of a collaborative learning system.
- T-PSM and CI-PSM conform to distinct meta-models, respectively T-PIM and CI-PIM, ensuring a clear separation between the general concepts of learning platforms and the specific collaboration indicators. This distinction allows better adaptation of indicators to platforms while maintaining a common structure throughout the transformation process.
- T-PSM and CI-PSM are at the same level of abstraction, which means they operate at a similar level of conceptual abstraction. This ensures consistency while allowing a direct and efficient transformation between models, without the need to adapt highly technical aspects specific to the underlying learning systems.
- It is a Model-to-Model (M2M) transformation, which means that the source and target models are abstract representations of the same domain of study. This facilitates a more systematic representation and flexible adaptation of data or processes without focusing on specific implementation details, thus allowing an easier transfer of collaboration indicators between different learning systems.

3.2.5 From T-PSM to CI-PSM Using Trace Transformation Operators

The transformation sequences applied to the trace model T-PSM in order to derive the collaboration indicator model CI-PSM must rely on a well-defined set of transformation operators. These transformations aim to produce new models or model instances from existing ones, according to formal transformation rules.

As previously described, these transformation operators fall into two main categories:

Operators that do not modify the trace model These operators act solely on the *instances* of the model (the m-traces), without altering the model structure itself. Notable operators in this category include:

- **Selection**: This operator extracts a sub-part of a trace based on specific criteria such as time period, observed action type, actor involved, or tool used.

- **Merging**: This operator combines two trace instances that share the same model into a single aggregated trace, unifying their observed elements.

Operators that modify the trace model These operators directly affect the *structure* of the trace model by adding, removing, or renaming elements. They are essential to adapt the trace model towards a more abstract or targeted form, suitable for indicator computation. The main operators in this category are:

- **Matching**: Identifies specific sequences of observed elements based on a predefined task pattern (task signature) and groups them under an abstract label.
- **Pruning**: Deletes certain classes or attributes from the model, performing fine filtering and reducing the model's complexity.
- **Rewriting**: Renames a class or attribute in the model, generating a new structure that better aligns with the analytical objective.
- **Insertion**: Adds a new attribute to an existing class, thereby enriching the model to introduce new observation dimensions.

Together, these operators form the foundation of the progressive transformation from T-PSM to CI-PSM, ensuring both structural adaptation of the model and semantic shaping of the traces required for computing meaningful collaboration indicators.

3.2.6 Transformation contracts

To express complex requirements that cannot be represented only using traditional UML diagrams, and to ensure both the validity and compliance of the transformation process, we propose the use of constraints in OCL (Object Constraint Language). These constraints allow the formalization of specific rules, often difficult to capture with UML graphical elements, and play a crucial role in the verification and validation of the transformation process. They provide a rigorous framework to ensure that the initial specifications are correctly translated and that the generated models meet the defined expectations. We propose the following constraints in OCL (Object Constraint Language), as illustrated in Table 3.4.

A collaboration indicator must be calculated based on at least one collaborative obsel, meaning that the indicator reflects real collaborative actions or events performed by learners during their interaction.	`context col_ind inv AtLeastOneCollaborativeObsel: self.obsel→exists(o | o.type.oclIsKindOf(collaborative))`
A non-collaborative indicator is computed only from non-collaborative obsels, ensuring that the indicator exclusively reflects individual or passive actions without collaboration.	`context non_col_ind inv OnlyNonCollaborativeObsels: self.obsel→forAll(o | o.type.oclIsKindOf(non_collaborative))`
An individual collaboration indicator must be linked to a single learner, indicating that it measures collaboration at the individual level only.	`context col_individuel inv IndividualTargetConstraint: self.targetEntity.oclIsTypeOf(learner)`
A group collaboration indicator is associated exclusively with a group entity, meaning the indicator assesses collaboration at the group level.	`context col_group inv GroupTargetConstraint: self.targetEntity.oclIsTypeOf(groupe)`
A community collaboration indicator must refer only to a community entity, capturing collaboration metrics at the community-wide scale.	`context col_community inv CommunityEntityConstraint: self.targetEntity.type = 'community'`
A composite indicator is formed only from existing sub-indicators, meaning it aggregates several simpler indicators into a combined measure.	`context indicateur inv CompositeIndicatorConstraint: self.isComposite implies self.subIndicators→forAll(i | i.oclIsKindOf(indicateur))`

A composite indicator must contain at least one sub-indicator to be valid and meaningful.	`context indicateur inv CompositeHasAtLeastOneSubIndicator: self.isComposite implies self.subIndicators→size() >= 1`
An indicator cannot simultaneously be classified as individual, group, and community level; it must belong to exactly one of these categories to avoid ambiguity.	`context indicateur inv ExclusiveTypeConstraint: (self.oclIsTypeOf(col_individuel) implies not (self.oclIsTypeOf(col_group) or self.oclIsTypeOf(col_community))) and (self.oclIsTypeOf(col_group) implies not self.oclIsTypeOf(col_community))`
Every indicator must specify the type of calculation it uses to compute its value, ensuring clarity in its measurement process.	`context indicateur inv CalculationTypeDefined: not self.calculationType.oclIsUndefined()`
All obsels linked to an indicator must belong to the same trace to guarantee consistency and coherence in the data analyzed.	`context indicateur inv ConsistentTraceConstraint: self.obsel→collect(o | o.trace)→ asSet()→size() <= 1`
The start time of an indicator must be less than or equal to its end time, ensuring that the time interval for measurement is valid.	`context indicateur inv NonEmptyTimeRange: self.startTime <= self.endTime`
Each obsel must have a start time less than or equal to its end time, guaranteeing that each recorded event has a valid time span.	`context obsel inv ObselTimeRange: self.startTime <= self.endTime`
The time interval of each obsel must be fully included within the time interval of its associated indicator, ensuring proper temporal alignment.	`context indicateur inv ObselWithinIndicatorTime: self.startTime <= self.obsel.startTime and self.endTime >= self.obsel.endTime`
In an active collaboration, the set of receivers must not be empty and must be different from the transmitter to ensure genuine interaction.	`context active inv ReceiversNotEmptyAndDifferentFromEmitter: self.receiver→notEmpty() and self.receiver→forAll(r | r <> self.transmitter)`
Each learner in the system must have a unique identifier to avoid confusion and maintain user distinction.	`context learner inv UniqueLearnerId: Learner.allInstances()→forAll(l1,l2 | l1<>l2 implies l1.id<>l2.id)`

Every learner can belong to only one group at a time, reflecting clear group membership.	`context learner inv SingleGroupMembership: self.group→size() = 1`
Similarly, each learner can belong to only one community to ensure membership clarity.	`context learner inv SingleCommunityMembership: self.group.community→size() = 1`
Each group must have a unique identifier to distinguish it from other groups.	`context group inv UniqueGroupId: Group.allInstances()→forAll(g1,g2 | g1<>g2 implies g1.id<>g2.id)`
Groups should contain only unique learners to avoid duplication in membership.	`context group inv UniqueLearnersInGroup: self.learners→forAll(l1,l2 | l1<>l2)`
Communities must contain unique groups to maintain proper organization and avoid overlap.	`context community inv UniqueGroupsInCommunity: self.groups→forAll(g1,g2 | g1<>g2)`

Table 3.4: Transformation contracts (OCL constraints)

Conclusion

In this chapter, we presented a methodological approach based on Model-Driven Architecture (MDA) for the computation of collaboration indicators in Computer-Based Human Learning Environments (CBHLEs). This approach aims to address the challenges related to the heterogeneity of learning platforms and the diversity of interaction traces.

We first introduced the platform-independent model of traces (Trace PIM), allowing a common abstraction of data from different environments. The addition of trace constraints was proposed to ensure the quality and consistency of the data being handled. We then defined the platform-independent model of collaboration indicators (Collaboration Indicator PIM), structuring the design and computation of indicators in a generic and reusable way.

The approach relies on transformation modes and precise transformation contracts, ensuring a rigorous transition between trace models and collaboration indicator models.

Chapter 4

Formal system for the design of collaboration indicators and the automatic generation of transformation sequences

Introduction

The evaluation of collaboration in Computer-Based Human Learning Environments (CBHLEs) requires indicators capable of representing the dynamics of interactions between learners. However, the design of such indicators is often complex due to the diversity of learning contexts and the variability of collaborative behaviors. To address these challenges, we propose a formal system to structure, design, and compute collaboration indicators in a rigorous and systematic way.

In this chapter, we introduce a formal model for the design of collaboration indicators.

We then describe the implementation of this model and its operation through a dedicated architecture, allowing users to design and compute collaboration indicators in a semi-automated way. The process of calculating the indicators is presented, highlighting the mechanisms for transforming trace models into valid collaboration indicator models.

4.1 Formal Model for the Design of Collaboration Indicators

4.1.1 The DCIN Model for the Design of Collaboration Indicators

Our proposed DCIN model is inspired by previous work on collaboration indicators. This model is structured around a name and an informal description [19], and according to [11], it can be computed at different levels, whether at the individual, group, or community level, while allowing a distinction between elementary and composite indicators [19]. An elemen-

tary indicator is built from traces, on which a set of transformations and calculations are applied to obtain the indicator value. In contrast, a composite indicator consists of other combined elementary indicators. Thus, based on the mentioned definitions of a collaboration indicator, and as presented in Figure 4.1, our DCIN model allows a flexible and rigorous design of collaboration indicators by identifying its name, its informal description, and its time interval. This model is designed to simplify the creation of collaboration indicators at the individual, group, and community levels. The indicators created using this model can be either elementary or composite, depending on the needs of the situation.

Our timed automata for the design of collaboration indicators facilitate the modeling process by guiding the designer through the crucial steps. The designer is first asked to identify the name of the indicator, its informal description, and its time interval. Then, they must choose the desired type of indicator, whether elementary or composite, depending on the complexity and the intended objectives. After this initial choice, the designer must determine the set of obsels involved in the creation of this indicator. These obsels represent the observation units or significant events that will feed into the indicator's calculation. Finally, the designer must determine whether the indicator will be focused on an individual, a group, or a community, depending on the evaluation objectives and the required level of analysis.

Our model also proposes predefined paths, which can be considered as ready-to-use templates or patterns for designing collaboration indicators in CBHLEs. These paths facilitate the creation of indicators in two main forms: "number" and "rate", which makes it easy to adapt the model to different types of measurements. A path represents the execution of an automaton, as defined by [13], where each step is guided by a series of predefined rules and transformations. These predefined paths, integrated into our model, enable partial automation of the indicator creation process while ensuring the validity of the created indicators.

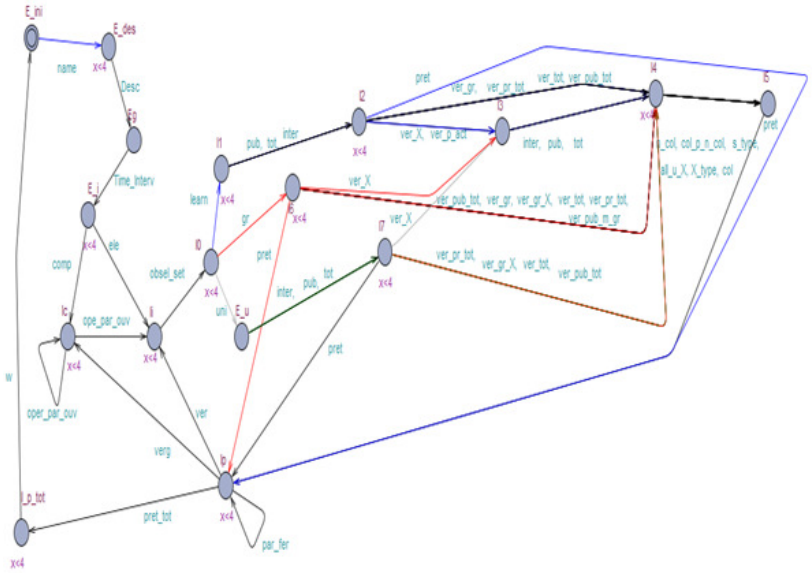

Figure 4.1: DCIN Model

4.1.1.1 Formal Definition of the DCIN Model

The model takes the form of a timed automaton, such that:

- E: Represents a finite set of transition labels;
- S: Represents a finite set of places;
- $X = (x)$ a finite set of variables having a value in R^+ (clocks).
- $I : S \rightarrow C(X)$ is a function that associates to a place s the set of conditions on X called Invariants.
- $T \subset S \times E \times C(X) \times 2^x \times S$ is the set of transitions. A transition $e =\prec s, a, k, m, s^{'} \succ \in T$ represents the transition from place s to place s', guarded by the constraint k, labeled by a, and resets the variables $m \in X$.
- $E - init$ belongs to S, initial place.

Where the elements of the sets E and S are:

$S = (E - ini, E - des, Ei, Eg, Ii, I0, I1, I2, I3, I4, I5, I6, I7, Ec, Ip, I - p - tot)$.

$E = (ver - gr, ver - pub - m - gr, col - p - n - col, Desc, Time - interv, ver - pub - tot, obsel - set, ver - gr - X, ver - X, ver - pr - tot, ver - p - act, all - u - X, ver - tot, n-col, Name, Uni, Inter, X-type, Col, s_type, oper-by-ouv, ope-par-ouv, Ele, pret-tot, pret, Learn, ver, verg, gr, pub, par - fer, tot)$.

The elements of set E are defined in 4.1.

Transition Label	Description
Ver-gr	Related to private activities of the members of the group.
Ver-pub-m-gr	Related to public activities of the members of the group.
Col-p-n-col	Collaborative and non-collaborative activities.
Desc	Textual description of the indicator to design.
Time-interv	Time interval of the indicator to design.
Ver-pub-tot	Compared to total public activities.
Obsel-set	The set of relevant observed elements.
Ver-gr-X	Related to activities of a group X.
Ver-X	Related to activities of a learner X.
Ver-pr-tot	Related to total private activities.
Ver-p-act	Related to the activities of the learner.
All-u-X	All types of obsel except initial types.
Ver-tot	Related to total activities.
n-col	Non-collaborative activities.
Name	Name of the indicator.
Uni	Universal (community-level).
Inter	Private within the group.
X-type	All obsels of type X.
comp	Composite indicator.
Col	Collaborative activities.
type	Same initial type.
oper-par-ouv	Operator (
Ope-par-ouv	Operator (
ele	Elementary indicator.
Pret-tot	Total ready.
pret	Partially ready.
learn	Related to a learner.

ver	Comma ",".
gr	Related to a group.
pub	Public.
Par-fer	) .
tot	Total.

Table 4.1: Elements of the set E

4.1.2 Implementation and Operation of the Proposed Approach

4.1.2.1 DCIN-AGST Formal System

We have implemented the DCIN model in a real-time system modeled as a timed automaton network in a formal system (DCIN-AGST system) using UPPAAL [29]. This formal DCIN-AGST system facilitates the design of collaboration indicators and allows the automatic generation of transformation sequences. The UPPAAL simulator is used to design collaboration indicators. The simulation control panel, presented in Figure 4.2, offers possible transitions to be refreshed at each step. The designer must choose one of the transitions. By refreshing the 'pret-tot' transition, the designer completes the design of their collaboration indicator. Then, they can retrieve the transformation sequence and the details of their indicator via the message sequence graph panel, shown in Figure 4.3.

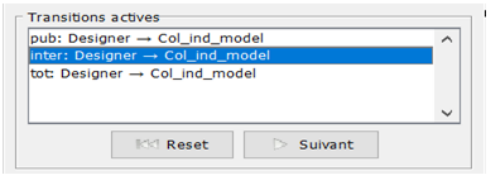

Figure 4.2: The simulation control panel of the UPPAAL simulator

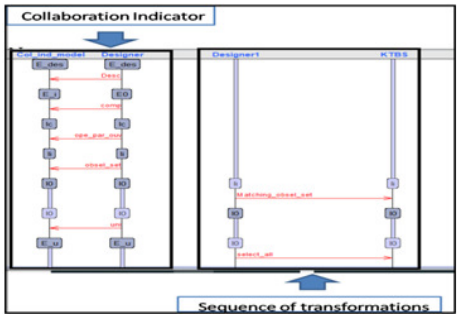

Figure 4.3: The message sequence graph panel of the UPPAAL simulator

In this framework, the designer can visualize the different stages of indicator creation, make choices on appropriate transitions, and get immediate feedback on the sequence of applied transformations. This process ensures not only an accurate and structured modeling of indicators but also automates the design process, making the creation of collaboration indicators faster and more reliable.

4.1.2.2 DCIN-AGST Formal System composition

Figure 4.4 shows the interface of the UPPAAL tool, displaying the "System Composition" tab where the process instantiations and their integration into a global system are defined. In this example, the `Col_ind_model` and `Aut_Gen` processes play a central role: `Col_ind_model` corresponds to the design model of collaboration indicators, while `Aut_Gen` is responsible for the automatic generation of transformation sequences. The `Designer` and `M_trace_based_sys` processes play a supporting role. The instruction `Process = Col_ind_model();` is used to instantiate the main model, while the command `system Col_ind_model, Designer, Aut_Gen, M_trace_based_sys ;` declares the full composition of the system being analyzed.

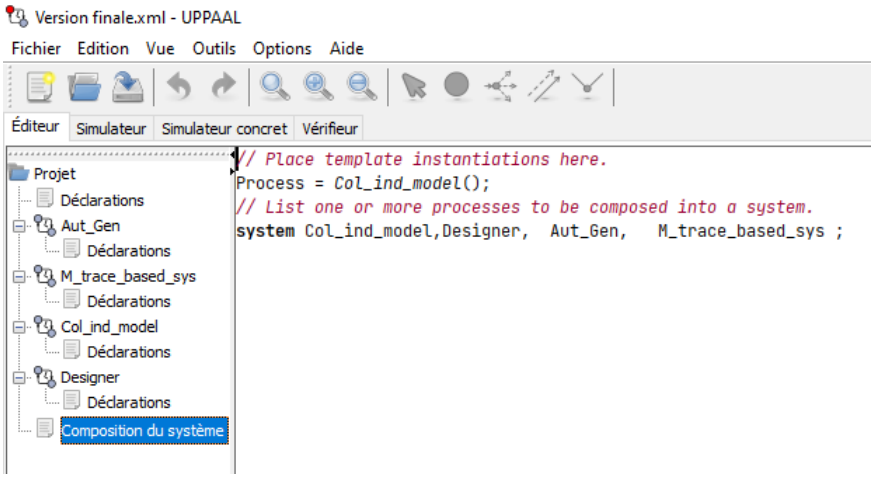

Figure 4.4: System composition in UPPAAL with instantiation and integration of main and supporting processes

As previously mentioned, the `Col_ind_model` process, illustrated in Figure 4.5, implements the DCIN (Design of Collaboration Indicators) model, designed to guide the modeling and design of collaboration indicators. This process is closely synchronized with the `Aut_Gen` process, shown in Figure 4.6, in order to automatically transmit the choices made by the designer (type of indicator, involved obsels, level of analysis, etc.).

Based on these choices, the `Aut_Gen` process automatically generates the appropriate sequence of transformations, enabling the transition from the trace model to the collaboration indicator model.

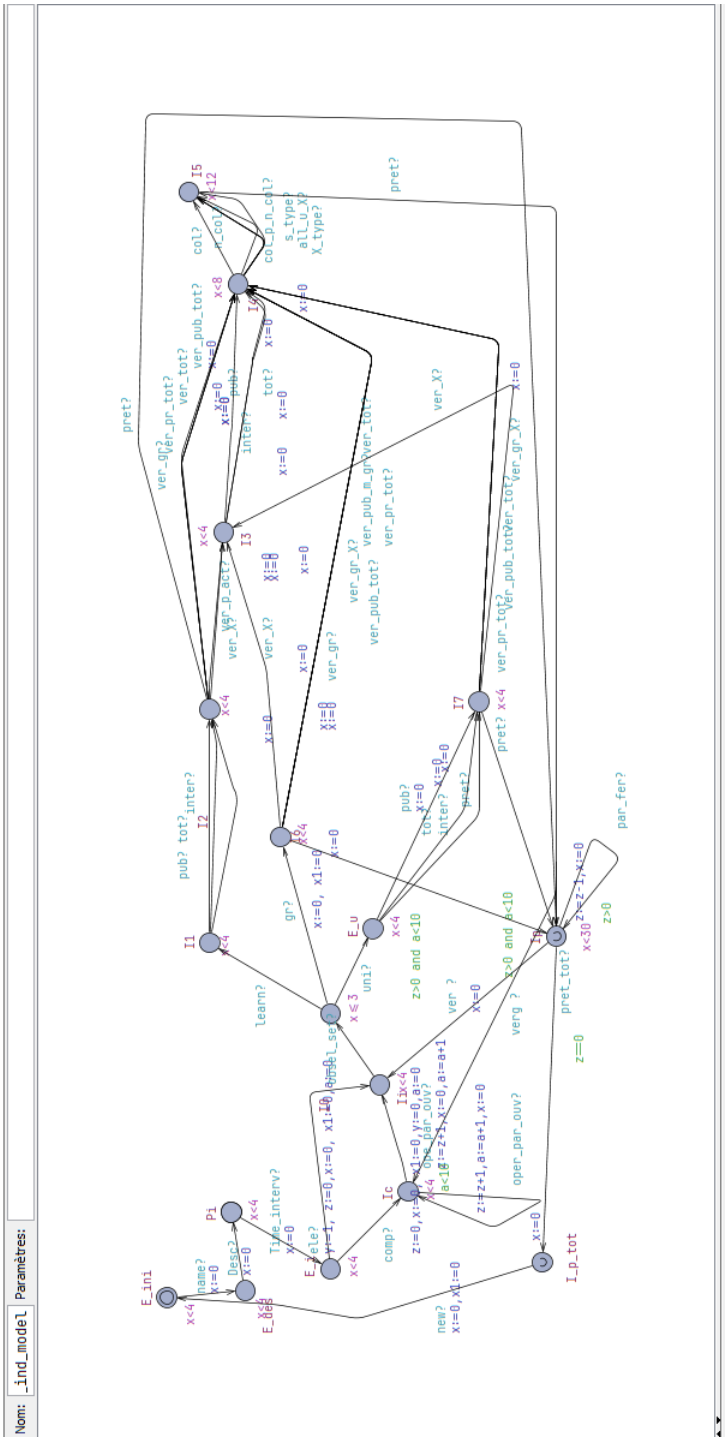

Figure 4.5: The Col_ind_model process based on the DCIN Model

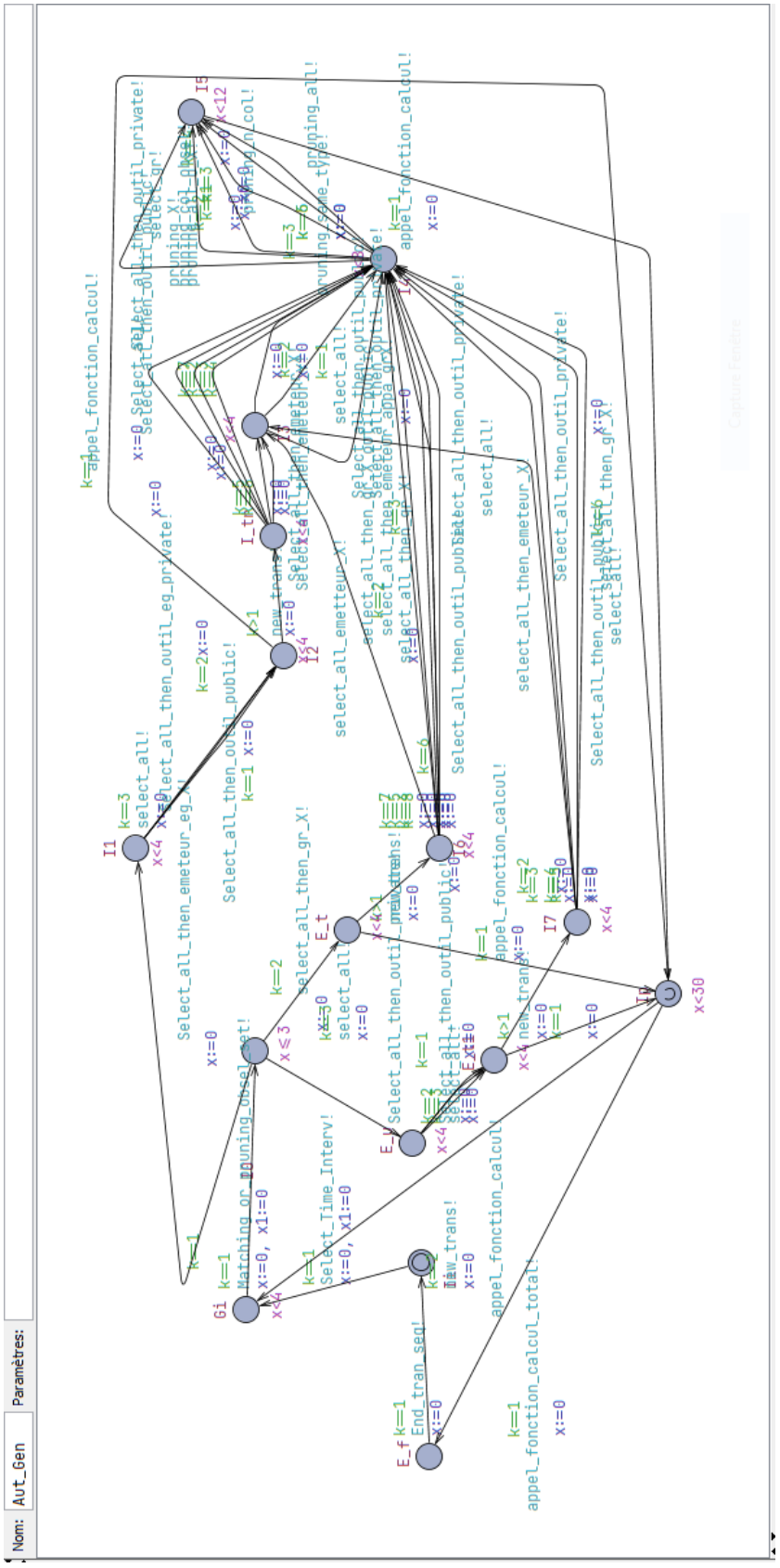

Figure 4.6: The Aut_Gen process

Figure 4.7 presents the global declaration of communication channels used in our formal system, as implemented in the UPPAAL environment. This declaration includes all synchronous channels (`chan`) enabling coordination between the different processes:

`Col_ind_model`, `Aut_Gen`, `Designer`, and `M_trace_based_sys`. These channels facilitate the sequencing of actions, in particular the selection of obsels, the definition of indicator parameters (type, period, level of analysis), as well as the automatic generation of the transformation sequence from the trace model to the collaboration indicator model.

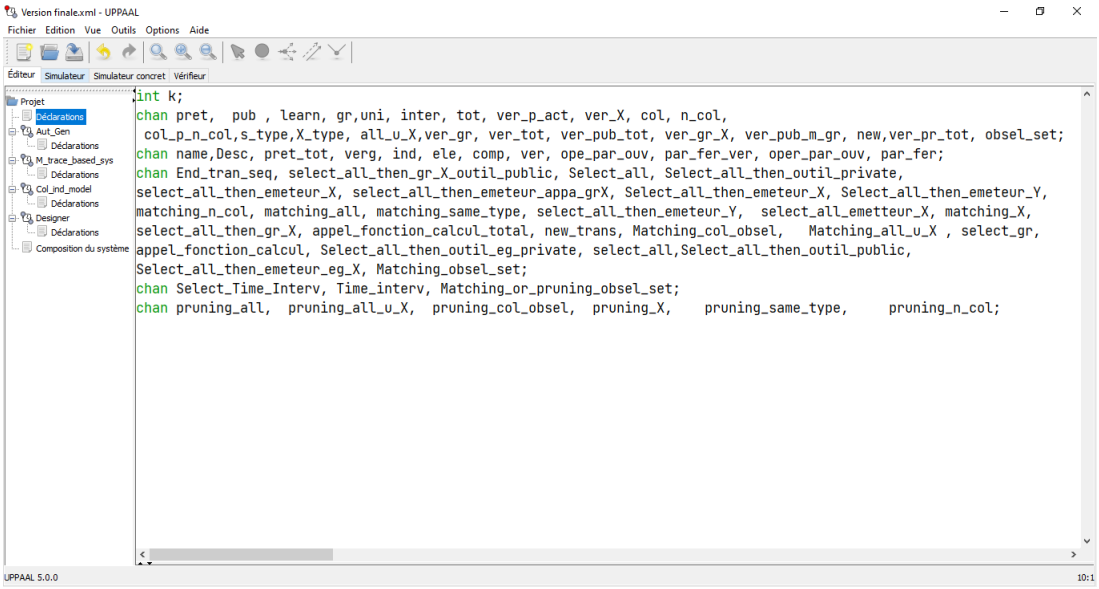

Figure 4.7: The global declaration of the DCIN-AGST Formal System

The local declaration of the `Col_ind_model` process, as shown in Figure 4.8, includes a clock variable used to control and monitor the timing of events during the indicator modeling process. This clock allows the system to manage temporal constraints, such as verifying whether a specific action or selection occurs within a defined time interval. In the context of the DCIN model, the use of a clock is essential because it is a real-time system.

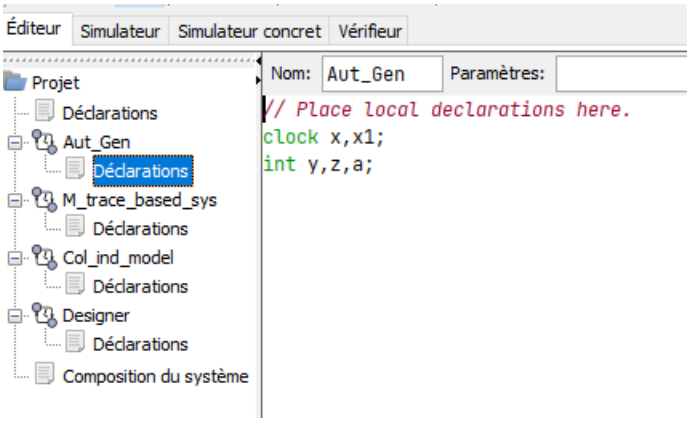

Figure 4.8: The local declaration of the DCIN-AGST Formal System

4.1.3 Calculation of Collaboration Indicator

In this subsection, we present in detail the method for calculating a collaboration indicator using our approach and the proposed system. As shown in the sequence diagram of Figure 4.9, the calculation of a collaboration indicator follows a series of structured steps that ensure both its pedagogical relevance and analytical validity.

First, the designer develops a model of collaborative and non-collaborative traces (T-PSM) specific to the chosen CBHLE. This model must comply with the T-PIM specification, thus ensuring consistency with system standards. This step forms the foundation of the approach, as it defines the categories of data to be analyzed and the criteria on which the indicator will be based. Then, the designer proceeds to collect the necessary data to build the trace base in the *m-trace base* system. This system uses the proposed model to organize and structure the raw traces from collaborative learning sessions. These raw traces are then transformed into exploitable data, ready for analysis.

Once this m-trace base is constructed, the designer uses the formal DCIN-AGST system to design a valid collaboration indicator. This indicator can be either elementary or composite, and it can be defined at different levels of analysis: individual, group, or community. This step is guided by the observation needs defined by the pedagogical designer or may follow one of the predefined paths in the proposed model, providing flexibility and adaptability. The DCIN-AGST system then automatically generates a transformation sequence, a set of precise steps that allows the transition from the trace model to the collaboration indicator model.

The obtained transformation sequence is then applied to the *m-trace base* system. This application allows transforming the data from the initial traces into a structured collaboration indicator model, while simultaneously collecting the relevant observed elements (*obsels*) related to this indicator. This process ensures the transparency and traceability of each step in the indicator construction, thus increasing trust in the obtained results.

Finally, the designer calls the calculation functions to determine the final value of the indicator. This value quantitatively and qualitatively reflects the observed collaborative interactions in the CBHLE. This rigorous process allows generating reliable indicators, which can be used by teachers and designers to evaluate, monitor, and improve collaborative dynamics within educational systems. By combining flexibility, automation, and precision, this methodology contributes to optimizing collaborative learning environments.

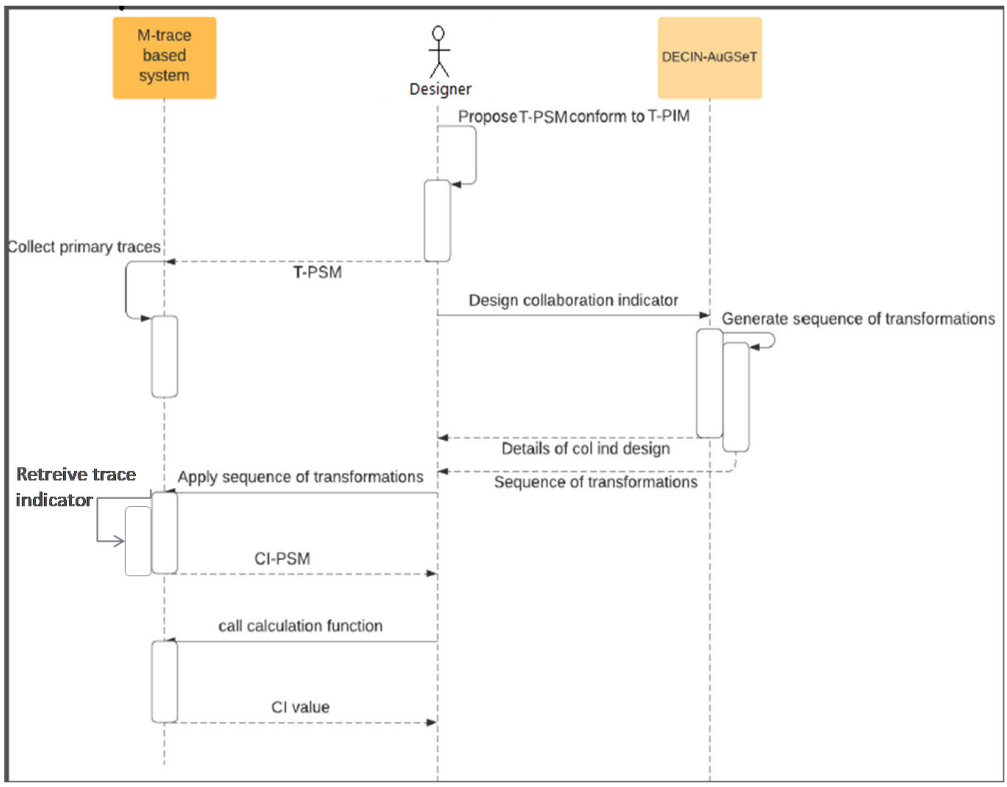

Figure 4.9: Sequence diagram showing the steps required to calculate a collaboration indicator

4.2 Conclusion

In this chapter, we presented a formal system for the creation of collaboration indicators, designed to allow teachers and tutors to create their own collaboration indicators without being limited by predefined ones. Thanks to the DCIN model, we proposed a system that allows educators to adapt collaboration indicators according to their specific needs and the learning context. At the end of the design process, the DCIN-AGST system generates a sequence of transformations to be applied to the trace model, thus producing the collaboration indicator model.

Finnally, this chapter summarized the key steps for computing a collaboration indicator using our model-driven approach. From trace modeling to indicator calculation, each phase ensures consistency, validity, and traceability, supporting the evaluation of collaboration in CBHLEs.

Chapter 5

Testing Phase

5.1 Introduction

After proposing a formal approach for the design and computation of collaboration indicators, it is necessary to evaluate its validity and robustness through a testing phase. This chapter is dedicated to the practical application and validation of our system based on the DCIN model, through various use cases and formal verification.

We first present concrete collaboration indicators designed and computed using the DECIN model, in order to illustrate the modeling of collaboration indicator basing on our proposed approach. Then, a use case of the DCIN-AGST system is detailed, showing how the approach can be applied in a real context of collaboration analysis.

We then proceed with a formal verification of the system, evaluating the internal consistency of the models, the conformity of the transformations, and the fulfillment of the specified requirements. Finally, we present the formally verified properties, attesting to the soundness and reliability of the design and computation process for the indicators.

5.2 Collaboration Indicators Designed Using the DECIN Model

The following examples illustrate three types of collaboration indicators that can be designed using our DCIN model, allowing effective formalization and structuring of indicator design in collaborative learning environments:

The first example concerns the elementary indicator 'Nbr-chat-gr', which represents the number of messages sent by a learner in a chat to members of their group. This indicator measures the learner's participation in online discussions, a fundamental aspect of collaborative engagement in a learning environment. To design this indicator, the designer will follow the path: $(E_{init}, 0) \rightarrow_{name} (E_{des}, 0) \rightarrow_{Des} (E_g, 0) \rightarrow_{Time-interv} (E_i, 0) \rightarrow_{ele} (I_i, 0) \rightarrow_{obsel-set} (I_0, 0) \rightarrow_{learn} (I_1, 0) \rightarrow_{inter} (I_2, 0) \rightarrow_{pret} (I_p, 0) \rightarrow_{pret-tot} (I_{p-tot}, 0)$.

This example highlights how an elementary indicator is designed, starting from its definition (name and description) and its time interval, up to the identification of the obsels used to compute its value. The indicator here is based on a learner's interactions in a specific environment, allowing tracking of their engagement within the group.

The second example concerns the composite indicator 'group participation rate', which evaluates the number of collaborative interactions performed within a group compared to the total number of collaborative interactions in the whole CBHLE. This composite indicator is essential to measure the effectiveness and impact of group dynamics on online collaboration. The design of this indicator follows the path: $(E_{init}, 0) \rightarrow_{name} (E_{des}, 0) \rightarrow_{Des} (E_g, 0) \rightarrow_{Time-interv} (E_i, 0) \rightarrow_{compo} (I_c, 0) \rightarrow_{ope-par-ouv} (I_i, 0) \rightarrow_{obsel-set} (I_0, 0) \rightarrow_{gr} (I_6, 0)$ $\rightarrow_{pret} (I_p, 0) \rightarrow_{ver} (I_i, 0) \rightarrow_{obsel-set} (I_0, 0) \rightarrow_{uni} (E_c, 0) \rightarrow_{tot} (I_7, 0) \rightarrow_{pret} (I_p, 0) \rightarrow_{par-fer} (I_p, 0) \rightarrow_{pret-tot} (I_{p-tot}, 0)$.

This example shows how a composite indicator can be designed from other elementary indicators. Here, the 'group participation rate' combines collaborative interactions within the group with those at the system level, allowing comparison between group participation and that of all learners. This process allows defining broader and more detailed indicators by integrating data at different levels.

The third example highlights the use of a predefined path to design an indicator of the type "rate of idea expression in a CBHLE." This indicator measures learner engagement through their contributions in forums, comparing them to overall collaborative interactions in the system. The design follows the path: $(E_{init}, 0) \rightarrow_{name} (E_{des}, 0) \rightarrow_{Des} (E_g, 0) \rightarrow_{Time-interv} (E_i, 0) \rightarrow_{ele} (I_i, 0) \rightarrow_{obsel-set} (I_0, 0) \rightarrow_{uni} (E_c, 0) \rightarrow_{tot} (I_7, 0) \rightarrow_{ver-tot} (I_4, 0) \rightarrow_{col} (I_5, 0) \rightarrow_{pret} (I_p, 0) \rightarrow_{pret-tot} (I_{p-tot}, 0)$.

This pre-existing indicator in our model shows the importance of learner contributions to idea generation in a collaborative learning platform. By using a predefined path, the designer benefits from a structured framework that simplifies indicator creation while ensuring their alignment with learning objectives.

Regarding the design of these collaboration indicators, two crucial points should be noted:

- Temporal transitions have been ignored here for simplicity, under the assumption that the designer makes instant decisions, allowing focus on the structure of the indicators rather than on the temporal details of each transition.
- The operators in 'composite' type indicators are written in prefix notation, which is a representation choice that simplifies the logical and mathematical expressions involved in the calculation of complex indicators.

5.3 Use Case of the Formal System DCIN-AGST

Let us take as an example the indicator Ic, defined as the number of chat messages sent by a learner to the members of their group. This example illustrates the power and flexibility

of the formal system DCIN-AGST in the design of precise and contextualized collaboration indicators. As shown in Figure 5.1, the design process of this collaboration indicator follows a series of structured steps guided by the system.

5.3.1 Detailed Design Steps

1. **Definition of the indicator**: As mentionned in the Figure **??**, the designer starts by defining the name of the collaboration indicator ($Ic1$), giving a clear description (for example, "Number of chat messages sent by a learner to their group"), and specifying the relevant time interval. This information forms the foundation of the indicator and ensures consistent interpretation of its results.

2. **Classification of the indicator**: Since $Ic1$ is an elementary indicator, the designer activates the `ele` transition in the UPPAAL simulator, indicating that this indicator is built directly from traces without using other indicators.

3. **Selection of obsels**: The designer then identifies the set of observed elements (*obsels*) concerned by this indicator. In this case, the selected obsels are the actions "send a message" carried out by learners.

4. **Identification of the actor**: The `learn` transition is activated to specify that the indicator focuses on the actions of a specific learner, distinguishing the message sender.

5. **Level of analysis**: By activating the `inter` transition, the designer specifies that this indicator is calculated at the group level, focusing on interactions within a specific group.

6. **Finalization**: By successively refreshing the `pret` and `pret-tot` transitions, the designer finalizes the design of the indicator $Ic1$. At this stage, all specifications are integrated, and the indicator is ready for use.

5.3.2 Generation and Visualization of Transformations

Thanks to the message sequence graph panel in the UPPAAL simulator, the designer can access a clear and detailed visualization of the design steps.

- **Left segment**: This segment(Figure 5.1) displays detailed information about the $Ic1$ indicator, including its name, description, selected obsels, and analysis level. This allows for quick and efficient validation by the designer.

- **Right segment**: This segment shows (Figure 5.2) the sequence of transformations to execute in order to move from the trace model to the collaboration indicator model $Ic1$. This visualization helps check the logic and integrity of the steps followed. The

automatic transformation sequence generated by the DCIN-AGST system allows moving from the T-PSM trace model to the CI-PSM collaboration indicator model. As shown in Figure 5.2, the process begins by selecting a time interval, followed by filtering to keep only the relevant obsels ("send a message"). Then, the specific actions of the learners are isolated, and finally, only interactions carried out within the learner's group are retained.

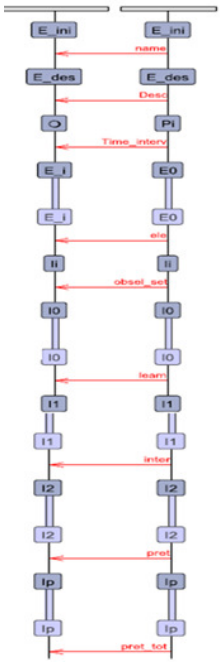

Figure 5.1: Execution to the design of the Ic1 indicator

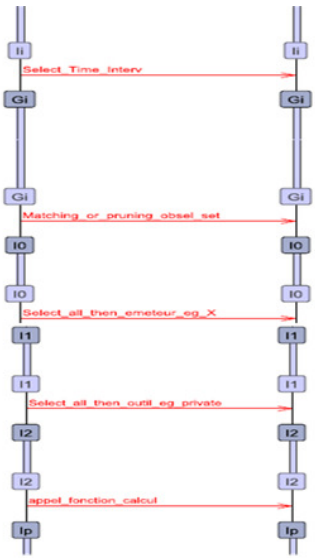

Figure 5.2: The automatically generated sequence of transformations for indicator Ic1

5.3.3 Interpretation of Results

5.3.4 Detailed Design and Transformation Sequence Generation

The indicator $Ic1$ represents the number of messages a learner sends to their group through the chat. It is defined using the structured approach of the DCIN-AGST system.

To define $Ic1$, the process follows several steps. First, the designer gives the indicator a name, a clear description (such as "Number of messages sent by a learner to their group"), and sets a time period for observation. This helps place the indicator in a meaningful and understandable context.

Next, the classification step uses the `ele` transition to mark $Ic1$ as an elementary indicator, meaning it is built directly from observable data, without combining it with other indicators. The designer then selects the relevant obsels, which in this case are actions where a message is sent in the learning environment.

To focus on a specific learner, the `learn` transition is used to filter the obsels so that only the messages sent by that learner are considered. The level of analysis is then set with the `inter` transition, indicating that the indicator is interpreted at the group level, taking into account interactions within the group.

Finally, the `pret` and `pret-tot` transitions are used to finalize and save the indicator in the system.

Once all these steps are completed, the DCIN-AGST system generates a transformation sequence that converts the trace model (T-PSM) into the indicator model (CI-PSM). This sequence is displayed in the UPPAAL simulation tool as a graph divided into two segments. The left segment contains descriptive elements of the indicator such as its name, the selected obsels, the type of actor, and the level of analysis. The right segment shows the transformation steps needed to produce the final indicator model $Ic1$ from the collected trace data.

The generated transformations include:

- **Time filtering**: limits obsels to those within the specified time interval.
- **Action type filtering**: selects obsels of type "message sent".
- **Actor filtering**: keeps obsels associated with a specific learner.
- **Contextual filtering**: restricts to messages sent to members of the learner's group.
- **Aggregation**: counts the obsels that meet the previous criteria, resulting in the value of the indicator $Ic1$.

5.3.5 Interpretation of $Ic1$ values to obtain

The collaboration indicator $Ic1$ corresponds to the number of messages sent by a learner to their group through the chat. It provides information about a learner's communication activity in a group work context. This indicator is built following the steps defined in the DCIN-AGST system, based on recorded traces.

Once calculated, this indicator can be interpreted in different ways depending on its value:

- A high value means the learner has sent a large number of messages. This may indicate a strong presence in group exchanges.
- A medium value may reflect regular participation, neither too low nor too high. This can match the expected level of involvement for the task or the instructions given.
- A low value may show limited participation. This may be due to several reasons: technical problems, lack of interest, absence, or unclear roles. This may require a check or intervention by the teacher.

Analyzing this indicator over a specific period or comparing it between different learners or groups can help better understand how each person is participating. It can also be used with other indicators to get a broader view of group work or collaborative activity.

5.4 Formal Verification of the System

The formal system **DCIN-AGST** has been rigorously verified using the formal verification method *model checking* via the verification tool **UPPAAL** [29]. This approach allows us to ensure the validity and safety of the DECIN-AGST System by testing essential properties such as safety, reachability, non-blocking/deadlock, and liveness. The verification of system properties is carried out using a simplified version of Computation Tree Logic (CTL) [33] and Timed Computation Tree Logic (TCTL) [34], which are used to model the behavior of real-time systems.

The interface of the **UPPAAL** verifier is shown in Figure 5.3. This interface allows real-time visualization of the model state and verifies whether the specified properties are satisfied. The verified properties are indicated by color codes: those that are satisfied appear in green, while those that are violated are in red. This feature ensures full transparency when validating the model and helps quickly identify possible errors.

Verified Properties

- **Non-blocking/deadlock property:** A state is considered a deadlock if no outgoing action transition is possible, nor any time successor. This property is expressed by the

formula:

$$A[] \text{ not deadlock.}$$

This ensures that the system will not remain in a state where no progress is possible. In our system, this property is fully satisfied, ensuring that no deadlock occurs during the indicator design.

- **Liveness property:** This property states that a desired event will inevitably occur. In other words, the system is designed so that a desired event will eventually happen. For our model, this means that a valid indicator will always be generated from the initial state to the final ready state:

$$A <> \text{Col-ind-model.}E_{ini} \text{ imply Col-ind-model.I-p-tot.}$$

This liveness property is satisfied, confirming that the system is dynamic and that the indicator design process is guaranteed to complete.

- **Safety property:** An undesirable event never occurred during the design process. At some point, the designer finalizes the indicator design, but the model reaches a different state than `I-p-tot` (missing synchronization):

$$E[]\text{Designer.I-p-tot and not Col-ind-model.I-p-tot.}$$

This property *is not satisfied* in our system. It ensures that no undesirable event occurred during the design process. It is crucial that indicator design is done without synchronization errors or rule violations. Such an undesirable event appears as a synchronization failure between the steps of the indicator design.

- **Reachability property:** "A certain situation must be reached." If the designer has chosen to design an indicator for a learner, they must reach one of the final states `I5` or `I2`:

$$A <> \text{Designer.I1 imply (Col-ind-model.I5 or Col-ind-model.I2).}$$

This property is satisfied in our system, meaning that the system can reach the specified final states according to the designer's choice. This reachability property ensures that a specific situation, defined by the model's final state, can be reached during the design process.

- **Safety property (opening and closing brackets):** The model allows the design of an invalid indicator containing more opening than closing brackets:

$$A <> \text{Col-ind-model.I-p-tot and Col-ind-model.z} > 0.$$

This property ensures that the model does not allow the design of an indicator with unbalanced brackets, which would be an error in the structure. The property *is not satisfied* in our system, meaning that the model does not allow the design of invalid indicators in any case.

- **Safety property (compositions in a composite indicator):** For complexity reasons, we must limit the number of compositions in a composite indicator to avoid an infinite loop. The system allows a finite number of compositions:

$$A[]\text{Col-ind-model.a} < 12.$$

The model imposes a limit on the number of compositions in a composite indicator to avoid infinite loops, which is essential to ensure system stability. This property is satisfied in our system, confirming that the system prevents overly complex compositions and efficiently manages composite indicators.

- **Safety property (invalid indicators):** There is at least one path that allows designing a valid indicator without reaching one of the final states of the model:

$$E[](\text{Col-ind-model.Ip and not Col-ind-model.I2 and not Col-ind-model.I5}$$
$$\text{and not Col-ind-model.I7 and not Col-ind-model.I6}).$$

This property ensures that there is always a path to design a valid indicator without passing through incorrect or invalid final states. The property *is not satisfied* in our system, preventing the design of indicators such as "number of chat messages" or "number of wiki edits," which are considered invalid collaboration indicators in this context.

- **Safety property (limited time for design):** The design of the indicator must be done within a limited time (critical resource issue for a single instance case):

$$A[](\text{Col-ind-model.x1} < 80 \text{ and Designer.x1} < 80).$$

This property ensures that the design of the indicator respects a time constraint, considered a critical resource. The system must guarantee that a valid indicator is designed within a limited time. The property is satisfied in our system, meaning that the time required to design an indicator is controlled and limited.

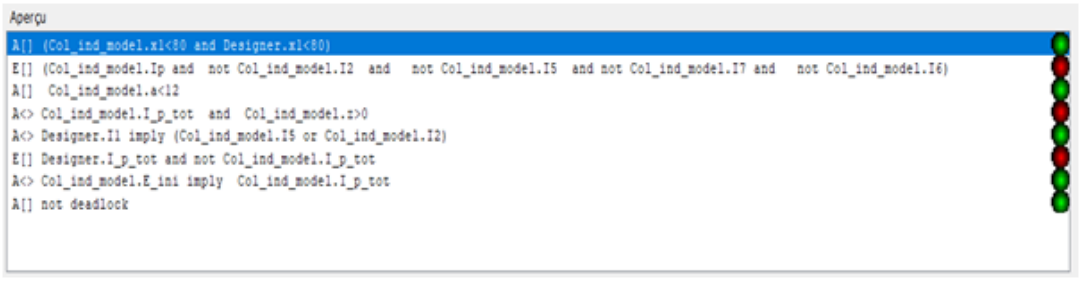

Figure 5.3: UPPAAL Verifier

5.5 Conclusion

This chapter has made it possible to experimentally and formally validate the proposed approach for the design and computation of collaboration indicators based on the DCIN model. Through the creation of concrete indicators, we have demonstrated the system's ability to model different types of collaborative behaviors in a structured and rigorous manner.

The use case of the DCIN-AGST system has shown the applicability of our approach in a real context, highlighting its flexibility and ability to adapt to the specificities of collaborative learning environments. The formal verification carried out confirmed the consistency of the transformations performed, the satisfaction of the specified constraints, and the validity of the system's critical properties.

General Conclusion

Summary of the Work

In the context of Computer-Based Human Learning Environments (CBHLEs), the evaluation of collaboration relies on indicators derived from the analysis of learning traces. The development and use of these indicators to support collaboration and collaborative learning represent a particularly dynamic research area.

In this thesis, our main objective is to propose a platform-independent method, accessible to teachers and tutors, that enables the design, customization, and calculation of collaboration indicators in CBHLEs. This method aims to ensure automation, reusability, and adaptability of the indicators to various collaborative learning contexts. This approach allows teachers, including those without specific technical skills, to benefit from appropriate tools to observe, analyze, and evaluate learner interactions, thus enabling more precise and personalized monitoring of collaborative dynamics. Moreover, these tools offer teachers a detailed view of learner behavior, facilitating the identification of difficulties as well as the adaptation of activities and teaching strategies in real time.

Our contributions fall within the evaluation and calculation of collaboration indicators in CBHLEs. The first contribution concerns the calculation of these indicators in a platform-independent manner, by adopting a model transformation-based approach and relying on the Model-Driven Architecture (MDA). The second contribution concerns the design of collaboration indicators, by proposing a formal model that ensures their validity and relevance, while addressing teachers' observation needs. Finally, the third contribution is based on the application development process specific to MDA, by considering the calculation of indicators as a model transformation process. The trace model thus undergoes a series of successive transformations leading to the collaboration indicator model. To ensure automation of this calculation, we propose a solution that automatically generates transformation sequences, which are then applied in the m-trace-based system to produce the final model of collaboration indicators.

We have defined an approach based on the Model-Driven Architecture (MDA) for the

calculation of collaboration indicators in CBHLEs, regardless of the platform used. In this context, we proposed a generic trace model (T-PIM), serving as a metamodel for collaborative and non-collaborative traces, as well as a platform-independent model for collaborative indicators (CI-PIM). In order to ensure the reliability and consistency of the traces used for indicator calculation, the T-PIM model was enriched with a set of formal constraints expressed in OCL (Object Constraint Language). These constraints formalize the structural, temporal, and semantic rules that model elements must follow. Their integration plays a key role in preventing inconsistencies and modeling or collection errors, thus strengthening the robustness of our approach and the relevance of the indicators calculated, regardless of the context or observation platform used.

The calculation of indicators is then based on a sequence of transformations applied to a specific trace model (T-PSM), conforming to T-PIM, to produce a collaboration indicator model (CI-PSM), conforming to CI-PIM. These transformations, written in a dedicated language, allow each step of the process to be formalized while ensuring their validity thanks to the associated OCL constraints.

To simplify and structure the design of collaboration indicators, we developed a formal model based on timed automata, called DECIN model(DEsigning Collaboration INdicator Model). This model guides the designer through the key modeling steps to ensure the validity and suitability of the indicators for observation needs. To automate this process, the DECIN model was implemented in a dedicated system (DECIN-AGST system), allowing the automatic generation of transformation sequences, which are then applied to trace models to obtain the corresponding collaboration indicators.

To ensure the reliability and safety of the DECIN-AGST system, we used formal verification by model checking with the UPPAAL tool. This validation ensured compliance with essential properties such as safety, reachability, absence of deadlock, and liveness of the system.

In conclusion, the contributions of this thesis to collaborative learning are both numerous and significant. It proposes an innovative and platform-independent method for the design and calculation of collaboration indicators, facilitating their integration into various CBHLEs. This approach allows teachers, including non-technical ones, to benefit from appropriate tools to observe, analyze, and evaluate learner interactions, thus enabling more precise and personalized monitoring of collaborative dynamics.

From a scientific perspective, this thesis makes an essential contribution to research on the evaluation of collaboration in CBHLE. By proposing an approach based on the Model-Driven Architecture (MDA), we have demonstrated the feasibility of a structured, reusable, and adaptable methodological framework for the design and calculation of collaboration indicators. The use of a formal model based on timed automata (DECIN model) not only ensures the validity of the indicators but also opens new perspectives in the formalization and validation of collaborative interaction evaluation processes.

The integration of formal verification by model checking using the UPPAAL tool is another major scientific contribution, ensuring the robustness and reliability of the proposed system. By combining this formal validation with the OCL constraints of the T-PIM and CI-PIM models, our work contributes to strengthening the methodological rigor in the design of collaborative learning indicators.

This research also contributes to the improvement of teaching practices by providing indicators that facilitate decision-making and the adaptation of teaching strategies based on learners' needs. By optimizing the evaluation of collaborative interactions, this thesis actively contributes to enriching learning environments, encouraging more effective, equitable, and learner-centered practices. Thus, it opens new perspectives for the development of innovative teaching methods adapted to the contemporary realities of education.

These results confirm the robustness of our approach and open the way for future improvements, notably by integrating artificial intelligence techniques to refine interaction analysis and improve the adaptability of indicators to various learning contexts. Although this work has made significant contributions to the formalization and design of collaboration indicators for CBHLE, some limitations remain and deserve to be highlighted.

First, the partial automation of the indicator design process is a notable limitation. Currently, certain steps, such as defining the T-PSM trace model conforming to T-PIM, retrieving transformation sequences, and optimizing calculation functions, still require human intervention. This dependency reduces the efficiency of the process and may limit its large-scale adoption. Full automation is envisioned to overcome this constraint.

Moreover, the demonstration of the completeness of the set of verified constraints, particularly those associated with the T-PIM and CI-PIM models through OCL constraints and the constraints verified for the DECIN-AGST system, is not yet exhaustive. Although preliminary results are encouraging, this demonstration requires further work to ensure the robustness and reliability of the approach in varied contexts.

Furthermore, the adaptability of the T-PIM and CI-PIM models to the specific needs of teachers and indicator designers could be improved. Currently, the refinement of the models is done semi-automatically, which may not fully meet the flexibility requirements of end users. The integration of artificial intelligence and machine learning techniques to dynamically adapt the models according to learning contexts represents a promising but still unexplored direction.

Finally, although experiments in a controlled environment have been carried out, our approach has not yet been extensively tested in real collaborative learning conditions. The impact of the indicators on the quality of interactions, learners' progress, and teaching effectiveness remains to be assessed in varied and authentic contexts. These experiments are crucial for refining our models, identifying areas for improvement, and evaluating the relevance of our method on a large scale.

Perspectives of work

As part of future perspectives, we aim to automate all steps of the process of designing and calculating collaboration indicators in CBHLEs. This includes the definition and customization of indicators, the retrieval and application of transformation sequences, and the optimization of calculation functions.

In addition, we will seek to demonstrate the completeness of all verified constraints, based on the formal validation of our system and the evaluation of the OCL constraints associated with the T-PIM and CI-PIM models.

As part of a continuous improvement approach and in accordance with the MDA approach, we also plan the progressive refinement of the T-PIM and CI-PIM models to better meet the specific needs of teachers and indicator designers. In parallel, we will explore the integration of artificial intelligence and machine learning techniques to improve the detection of collaboration patterns and dynamically adapt the indicators according to learning contexts.

Finally, we plan to validate our approach through experiments in real collaborative learning conditions. These experiments will make it possible to analyze the impact of the indicators on the quality of interactions and learners' progress, refine our models, identify possible areas for improvement, and evaluate the relevance of our approach in various educational contexts.

In conclusion, the contributions of this thesis to collaborative learning are both multiple and significant. It proposes an innovative and platform-independent method for the design and calculation of collaboration indicators, facilitating their integration into various CBHLEs. This approach allows teachers, including those who are not technicians, to benefit from suitable tools to observe, analyze, and evaluate interactions between learners, thus promoting more precise and personalized monitoring of collaborative dynamics.

In addition, the introduction of a formal model for the design of indicators ensures the validity and relevance of the designed collaboration indicators, meeting the specific needs of teachers and educational contexts. The automation of the indicator calculation process not only saves time but also ensures greater reliability in the analysis of collected data.

This research also contributes to the improvement of teaching practices by providing indicators that facilitate decision-making and the adaptation of teaching strategies according to learners' needs. By optimizing the evaluation of collaborative interactions, this thesis actively contributes to enriching learning environments, encouraging more effective, equitable, and learner-centered collaborative practices. Thus, it opens new perspectives for the development of innovative pedagogical methods adapted to the contemporary realities of education.

In short, the work carried out in this thesis paves the way for a flexible, automated, and platform-independent approach to the calculation of collaboration indicators in CBHLEs. By pursuing the outlined perspectives, we hope to contribute to the improvement of collaborative learning environments, by facilitating the analysis of interactions and providing teachers with

better tools to support and evaluate learners' collaborative work.

Bibliography

[1] M. Berthelot et E. Garrot. *Les rôles du tuteur à distance dans une formation universitaire hybride: le cas du dispositif AGIRE.* Environnements Informatiques pour l'Apprentissage Humain (EIAH), 2011.

[2] Jiang, Y., Wang, S., Yu, Z., Li, Y., and Miao, C. (2021). A survey on multi-modal collaborative learning. *ACM Computing Surveys (CSUR)*, 54(10), 1–38.

[3] Dillenbourg, P. (2009). Orchestration graphs: Modeling scalable education. In *European Conference on Technology Enhanced Learning* (pp. 489–502). Springer.

[4] L. Pei, C. Poortman, K. Schildkamp, et al., « Teachers' and students' perceptions of a sense of community in blended education », *Education and Information Technologies*, vol. 29, pp. 2117–2155, 2024.

[5] G. N. Doğuer and D. Öner, « Examining sense of community in the pandemic: A case of an online course », *Journal of Educational Technology and Online Learning*, vol. 6, no. 3, pp. 602–624, 2023.

[6] T. Samuel-Azran, S. Goldberg, T. Z. Hayat, and Y. Amichai-Hamburger, « The Social Side of Online Learning: How Social Capital Can Enhance Online Learning », *SAGE Open*, vol. 15, no. 1, 2025.

[7] France Henri et Karin Lundgren-Cayrol. *Apprentissage collaboratif à distance.* PUQ, 2001.

[8] Begonya Gros. *Collaborative systems and multi-agent systems: The missing link?* Educational Technology, 37(2), pp. 45–50, 1997.

[9] Gangloff-Ziegler, C. *The obstacles of collaborative work, Steps and organizations.* 2009, N-10(3), 95–112.

[10] Arnaud, M. *Les limites actuelles de l'apprentissage collaboratif en ligne [Current Limitations of Online Collaborative Learning]* (in French), STICEF – Sciences et Technologies de l'Information et de la Communication pour l'Éducation et la Formation, vol. 10, 2003, 7 pages. [Online]. Available: https://shs.hal.science/hal-00696421/.

[11] Dimitrakopoulou, A.: State of the art on Interaction and Collaboration Analysis. Deliverable D31.1.1 of the Kaleidoscope Network of Excellence, 2004.

[12] Fessakis, G., Petrou, A., and Dimitracopoulou, A. (2004). *Collaboration Activity Function: An Interaction Analysis Tool for Computer Supported Collaborative Learning Activities.* In *Proceedings of the 4th IEEE International Conference on Advanced Learning Technologies (ICALT 2004)* (pp. 196–200). Joensuu, Finland: IEEE.

[13] R. Alur, Timed Automata , in N. Halbwachs and D. Peled (Eds.), *Computer Aided Verification*, Springer, pp. 8–22, 1999.

[14] Bratitsis, T., Dimitracopoulou, A. *Monitoring and Analyzing Group Interactions in Asynchronous Discussions with the DIAS System,* 2006. In Y. A. Dimitriadis, I. Zigurs, and E. Gómez-Sánchez (Eds.), Groupware: Design, Implementation, and Use (Vol. 4154, pp. 54–61).

[15] Collazos, C. A., Guerrero, L. A., Pino, J. A., Renzi, S., Klobas, J., Ortega, M., Redondo, M. A., Bravo, C. *Evaluating Collaborative Learning Processes using System-based Measurement,* 2007. 18.

[16] Djelil, F., Hoffmann, C., Haddouche, A., Nadine, M., d'Ham, C. C. *Proposal of Collaborative Writing Indicators and Their Evaluation,* 2023 In: Computer Based Environments for Human Learning. (in French), Brest, France. Available: https://hal.archives-ouvertes.fr/hal-04213363f.

[17] Choquet, C., Iksal, S. *Modeling and Constructing usage Traces of a Learning Activity: A Language Approach for Reengineering a CEHL* (In French), 2007.

[18] Djouad, T., Mille, A., Reffay, C., Benmohamed, M. *Ingénierie des indicateurs d'activités à partir de traces modélisées pour un Environnement Informatique d'Apprentissage Humain [Engineering of Activity Indicators from Modeled Traces in a Computer Environment for Human Learning]* (in French), STICEF, vol. 16, no. 1, 2009, pp. 103–139.

[19] Gendron, É. *Cadre conceptuel pour l'élaboration d'indicateurs de collaboration à partir de traces d'activités [A Conceptual Framework for the Development of Collaboration Indicators from Activity Traces]* (in French), PhD thesis, Université Claude Bernard Lyon 1, 2010.

[20] Randriamalaka, N., Iksal, S., and Choquet, C. *Elicitation des indicateurs pour la réingénierie des scénarios pédagogiques : Approche à base de traces utilisant UTL [Elicitation of Indicators for the Re-engineering of Educational Scenarios: A Trace-Based Approach Using UTL]* (in French), 2008.

[21] C. Acosta, H. Derouin, R. Nodenot, et D. Guillot, « A Computer-Supported Approach for the Specification and Calculation of Collaboration Indicators in CSCL Contexts », *International Journal of Computer-Supported Collaborative Learning*, vol. 19, 2024.

[22] Ngoc, D.P.T.: *Spécification et conception de services d'analyse pour l'usage d'un Environnement Informatique pour l'Apprentissage Humain* [Specification and Design of Analysis Services for the Use of a Computer Environment for Human Learning]. Thèse de doctorat, 2004. [En ligne]. Disponible : https://theses.hal.science/tel-00689025v1

[23] Matazi, I., Bennane, A., Messoussi, R., Touahni, R., Oumaira, I., Korchiyne, R. *Multi-Agent System Based on Fuzzy Logic for E-Learning Collaborative System,* 2018, International Symposium on Advanced Electrical and Communication Technologies (ISAECT), 1–7.

[24] El Mhouti, A., Erradi, M., El Makhfi, N. *A Multi-Agent System of Semantic Analysis and Filtering of Modeled Traces to Calculate Interaction Indicators Favoring Collaboration in LMS,* 2019, International Conference on Intelligent Systems and Advanced Computing Sciences (ISACS), 1–7.

[25] Baron, M., Vivet, M.: Modélisation de connaissance pour des environnements interactifs d'apprentissage avec ordinateur. In: *Actes des 5èmes Journées Nationales PRC-GDR-IA*, Teknea, Nancy, 1995, pp. 239–262.

[26] Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge*. Morgan Kaufmann.

[27] A. Jézégou, « La présence à distance en formation à distance : un cadre conceptuel pour comprendre les modalités pédagogiques mises en œuvre dans un dispositif de formation médiatisée », *Distances et Savoirs*, vol. 8, n° 4, pp. 469–496, 2010.

[28] Sleeman, D. H., and Brown, J. S. (1982). *Intelligent Tutoring Systems*. Academic Press.

[29] G. Behrmann, A. David, and K. G. Larsen, « A Tutorial on Uppaal », in M. Bernardo and F. Corradini (Eds.), *Formal Methods for the Design of Real-Time Systems*, Lecture Notes in Computer Science, vol. 3185, Springer, Berlin, Heidelberg, pp. 200–236, 2004.

[30] Carbonell, J. R. (1970). AI in CAI: An Overview. *IEEE Transactions on Man-Machine Systems*, *11*(1), 1–10.

[31] Vygotsky, L. S. (1978). *Mind in society: The development of higher psychological processes.* Cambridge, MA: Harvard University Press.

[32] Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas.* New York: Basic Books.

[33] M. Reynolds, « An axiomatization of full Computation Tree Logic », *The Journal of Symbolic Logic*, vol. 66, no. 3, pp. 1011–1057, 2001.

[34] P. Bouyer, « Model-Checking Timed Temporal Logics », *Electronic Notes in Theoretical Computer Science*, vol. 231, pp. 323–341, 2009.

[35] De Jonge, J., & Towne, D. M. (1992). *Simulation: A complementary learning tool in intelligent tutoring systems.* In D. H. Jonassen (Ed.), *Handbook of Research for Educational Communications and Technology* (pp. 987–1013). New York: Macmillan.

[36] Dillenbourg, P. (Ed.): Collaborative Learning: Cognitive and Computational Approaches. Elsevier, 1999.

[37] Paquette, G.: Instructional Engineering in Networked Environments. Pfeiffer, 2004.

[38] Anderson, T.: Toward a Theory of Online Learning. In: T. Anderson and F. Elloumi (Eds.), *Theory and Practice of Online Learning*, Athabasca University Press, 2004.

[39] Brusilovsky, P.: Adaptive Hypermedia. User Modeling and User-Adapted Interaction, 11(1-2), 87–110, 2001.

[40] Mohamed, B.: Learning Management Systems and E-learning Standards. *International Journal of Emerging Technologies in Learning (iJET)*, 7(1), 2012.

[41] Garrison, D.R., Anderson, T., Archer, W.: Critical Inquiry in a Text-Based Environment: Computer Conferencing in Higher Education. The Internet and Higher Education, 2(2–3), 87–105, 2000.

[42] Vygotsky, L. S.: Mind in Society: The Development of Higher Psychological Processes. Harvard University Press, 1978.

[43] Johnson, D.W., Johnson, R.T.: Cooperative Learning and Social Interdependence Theory. In: R. S. Tindale et al. (Eds.), *Theory and Research on Small Groups*, Springer, 1998.

[44] Slavin, R.E.: Cooperative Learning: Theory, Research, and Practice. Allyn and Bacon, 1995.

[45] Peraya, D.: Distance education and presence. *Revue Distances et Savoirs*, 2002.

[46] Guichon, N.: Richness and complexity of the learning environment: Limits of instrumental mediation. In: *Apprentissage des langues et interactions en ligne*, ENS Éditions, 2009.

[47] Paquette, G.: Designing Learning Objects for Virtual Learning Environments. *Educational Technology and Society*, 5(1), 2002.

[48] Delestre, P.: Collaborative Learning in Online Training Systems. In: *Proceedings of the TICE Conference*, 2013.

[49] Berger, C.: Virtuality and Collaboration: Sociotechnical Models. *Revue des Sciences de l'Éducation*, 2007.

[50] Dejean, C.: Presence and Social Interaction in Virtual Environments. *Réseaux*, 20(115), 2002.

[51] Wenk, A.: The Feeling of Presence in Online Collaborative Learning. *Education and Information Technologies*, 2010.

[52] Johnson, D.W., Johnson, R.T.: An Educational Psychology Success Story: Social Interdependence Theory and Cooperative Learning. *Educational Researcher*, 38(5), 365–379, 2009.

[53] Roschelle, J., Teasley, S.D.: The Construction of Shared Knowledge in Collaborative Problem Solving. In: O'Malley, C. (Ed.), *Computer-Supported Collaborative Learning*, Springer, 1995.

[54] Bruner, J.: The Culture of Education. Harvard University Press, 1996.

[55] Rebaiaia, M.-L., Mille, A.: Mesure et visualisation des interactions collaboratives à partir de traces modélisées. In: *Environnements Informatiques pour l'Apprentissage Humain (EIAH)*, 2021.

[56] Berger, C.: Le rôle du tuteur à distance dans les dispositifs d'enseignement médiatisés. *Distances et Médiations des Savoirs*, 2009.

[57] Guichon, N.: Le tuteur dans les dispositifs d'apprentissage en ligne: entre guidage et médiation. *Revue Française de Pédagogie*, 154, 2006.

[58] Panitz, T. (1996). A definition of collaborative vs cooperative learning. [Online] Available: https://files.eric.ed.gov/fulltext/ED448443.pdf

[59] Johnson, D. W., Johnson, R. T., and Holubec, E. J. (1999). *Cooperation in the classroom* (8th ed.). Interaction Book Company.

[60] Bruffee, K. A. (1995). Sharing our toys: Cooperative learning versus collaborative learning. *Change: The Magazine of Higher Learning*, 27(1), 12–18.

[61] Dam, G., and Volman, M. (1995). Collaborative learning and the role of the teacher. *Educational Studies*, 21(1), 13–27.

[62] Strauss, R., & Goetz, T. (2001). Group work: The influence of group size and composition on performance. *Journal of Educational Psychology*, 93(3), 564-576.

[63] Webb, N. M. (2009). Teachers' interventions in peer interactions. *International Journal of Educational Research*, 48(2), 112-124.

[64] Roschelle, J., & Teasley, S. D. (1995). The construction of shared knowledge in collaborative problem solving. *Cognitive Science*, 19(3), 319-349.

[65] Barkley, E. F., Cross, P. K., & Major, C. H. (2005). Collaborative learning techniques: A handbook for college faculty. *Jossey-Bass*.

[66] Kreijns, K., Kirschner, P. A., & Jochems, W. (2003). Identifying the pitfalls for social interaction in computer-supported collaborative learning environments: A review of the research. *Computers in Human Behavior*, 19(3), 335-353.

[67] Webb, N. M. (1989). Peer interaction and learning in small groups. *International Journal of Educational Research*, 13(1), 21-38.

[68] Fjuk, A. (1998). Collaborative learning and group dynamics. *Scandinavian Journal of Educational Research*, 42(4), 377-392.

[69] De Wever, B., Schellens, T., Valcke, M., and Van Keer, H. (2008). A review of research on online discussion groups: Design, participation, and assessment. *Computers in Human Behavior*, 24(5), 778-795.

[70] Michaelsen, L. K., Knight, A. B., and Fink, L. D. (2004). Team-based learning: A transformative use of small groups in college teaching. *Jossey-Bass*.

[71] Tchounikine, P. (2011). Computer-supported collaborative learning: An overview of research perspectives. *Educational Technology and Society*, 14(3), 151-163.

[72] Jansen, B., *The Effect of User Interaction on Online Behavior*, Journal of Online Behavior, 2025.

[73] Meister, A., *On the Evolution of Educational Technology*, Educational Technology Journal, 15(3), pp. 45-55, 1967.

[74] Belgroun, M., *Criteria for Trace Data Interpretation in Collaborative Learning Environments*, Journal of Learning Analytics, 9(4), pp. 102-110, 2020.

[75] Settouti, A., *Modeling Learner Interaction in Collaborative Environments*, International Journal of Educational Technology, 18(1), pp. 34-40, 2006.

[76] Bratitsis, T., *Analyzing User Interactions through Modeled Traces*, Journal of Educational Computing, 22(2), pp. 78-89, 2005.

[77] Settouti, L. S., Prié, Y., Marty, J.-C. (2009). Modèle de traces pour une observation indépendante des outils dans les EIAH. *Revue des Nouvelles Technologies de l'Information*, 16(1), 103–139.

[78] K. A. Bruffee, *Collaborative Learning: Higher Education, Interdependence, and the Authority of Knowledge*, Johns Hopkins University Press, Baltimore, 1999.

[79] George, S., Garrot, E., Prieur, M. (2011). Traces et apprentissage: quels apports mutuels? *Sciences et Technologies de l'Information et de la Communication pour l'Éducation et la Formation (STICEF)*, 18.

[80] Prié, Y., Marty, J.-C., Mille, A., and Lardon, J. (2005). Traces et systèmes à base de traces: un cadre général pour la recherche sur la gestion des traces. In *Environnements Informatiques pour l'Apprentissage Humain* (EIAH 2005).

[81] Djouad, T., Mille, A., Reffay, C., and Benmohamed, M. (2009). Engineering of activity indicators from modeled traces in a Computer Environment for Human Learning. *Revue des Nouvelles Technologies de l'Information*, 16(1), 103–139.

[82] Mostow, J., & Beck, J. (2006). Evaluation of an interactive tutoring system for reading that listens. In *Proceedings of the 21st National Conference on Artificial Intelligence.*

[83] Dimitracopoulou, A. (2004). Designing collaborative learning systems: Current trends and future research agenda. In *Proceedings of the International Conference on Advanced Learning Technologies* (pp. 185–189).

[84] Choquet, C. (2005). DPULS: Vers un dispositif d'observation et d'analyse des usages des environnements informatiques pour l'apprentissage humain. In *Actes de la conférence EIAH 2005.*

[85] Dimitracopoulou, A., & Bruillard, É. (2006). Indicators: From research to usage analysis. *Sciences et Technologies de l'Information et de la Communication pour l'Éducation et la Formation (STICEF)*, 13, 65–84.

[86] Ngoc, D. P. T., Reffay, C., & Labat, J.-M. (2021). A framework for the design of interaction indicators in collaborative learning environments. *International Journal of Artificial Intelligence in Education.*

[87] Soller, A., Martinez, A., Jermann, P., & Muehlenbrock, M. (2005). Designing collaborative learning conversation tool feedback for learners and groups. In *Proceedings of the Conference on Computer Supported Collaborative Learning (CSCL)* (pp. 514–523).

[88] Jim Melton and Alan R. Simon. *Understanding the New SQL: A Complete Guide*. Morgan Kaufmann, 2002.

[89] Donald Chamberlin. XQuery: A query language for XML. *Computer Networks*, 39(5):557–589, 2003.

[90] W3C. XQuery 1.0: An XML Query Language. 2007. https://www.w3.org/TR/xquery/

[91] Gary Riley. *CLIPS Reference Manual*, 2002. http://www.clipsrules.net/

[92] Ivan Kurtev, Jean Bézivin, and Mehmet Aksit. Technological spaces: an initial appraisal. In *International Conference on Software and Systems Modeling (MoDELS)*, 2006.

[93] Sébastien Gendron. *Indicateurs pour le suivi d'activités collaboratives en environnement informatique pour l'apprentissage humain*. PhD thesis, Université du Maine, 2010.

[94] A. Juan, J. Cano, J. Faulin, and P. Vila, "A control charts-based monitoring system for learning management systems," in *Proceedings of the 2009 International Conference on Education and Information Technology*, IEEE, 2009, pp. 102–106.

[95] M. Zorrilla, D. García, and E. Álvarez, "A monitoring tool for a virtual learning environment: Design and implementation," *Computers and Education*, vol. 54, no. 4, pp. 1121–1131, 2010.

[96] M. Vásquez-Bermúdez, C. Alario-Hoyos, and C. Delgado-Kloos, "Defining and evaluating collaboration indicators in online learning environments," *IEEE Transactions on Learning Technologies*, vol. 15, no. 1, pp. 77–88, 2022.

[97] S. Praharaj, T. Dutta, and A. Mitra, "Voice-based collaboration analytics using natural language processing and semantic analysis," in *Proceedings of the 14th International Conference on Educational Data Mining (EDM 2021)*, 2021, pp. 562–567.

[98] S. García-Sastre, A. Pardo, and C. Delgado Kloos, "Monitoring collaborative learning processes using Moodle logs," in *Proceedings of the 9th International Conference on Computer Supported Education (CSEDU 2017)*, vol. 1, 2017, pp. 60–70.

[99] Dimitracopoulou, A., Komis, G., Tselios, V., & Pombortsis, I. (2004). Designing Interaction Analysis Tools for Learners and Teachers: Supporting "Learning to Learn" in Collaborative Learning Activities. In *Proceedings of the International Conference on Networked Learning 2004* (pp. 385–392). Lancaster University.

[100] Dimitracopoulou, A., & Komis, G. (2005). Design principles for tools supporting regulation in collaborative learning settings: The role of the Interaction Analysis (IA) tools. In *Proceedings of the 12th Euro-CSCL Conference* (pp. 293–302).

[101] Dimitracopoulou, A., Gillet, D., Dillenbourg, P., & Komis, G. (2006). CAViCoLA: A Tool for Visualization and Analysis of Collaborative Learning Activities. In *Proceedings of the International Conference on Advanced Learning Technologies (ICALT)* (pp. 1101–1105). IEEE.

[102] Choquet, C. (2007). *Une approche dirigée par les modèles pour l'ingénierie des EIAH: application à l'observation et à l'analyse d'usages*. Thèse de doctorat, Université du Maine.