Journal of Universal Computer Science, vol. 29, no. 3 (2023), 265-297
submitted: 29/3/2022, accepted: 29/12/2022, appeared: 28/3/2023

CIMLA: A Modular and Modifiable Data Preparation, Organization, and Fusion Infrastructure to Partially Support the Development of Context-aware MMLA Solutions

Shashi Kant Shankar
(School of Digital Technologies, Tallinn University, Tallinn, Estonia
https://orcid.org/0000-0001-8266-3681, shashik@tlu.ee)

Adolfo Ruiz-Calleja
(GSIC-EMIC Research Group, Universidad de Valladolid, Valladolid, Spain
https://orcid.org/0000-0003-1717-6304, adolfo@gsic.uva.es)

Luis P. Prieto
(School of Educational Sciences, Tallinn University, Tallinn, Estonia
https://orcid.org/0000-0002-0057-0682, lprisan@tlu.ee)

María Jesús Rodríguez-Triana
(School of Digital Technologies, Tallinn University, Tallinn, Estonia
https://orcid.org/0000-0001-8639-1257, mjrt@tlu.ee)

Pankaj Chejara
(School of Digital Technologies, Tallinn University, Tallinn, Estonia
https://orcid.org/0000-0002-7630-5789, pankajch@tlu.ee)

Sandesh Tripathi
(Department of Computer Science and Engineering, Birla Institute of Applied Sciences, Bhimtal, India
https://orcid.org/0000-0002-9771-352X, sandeshtripathi@birlainstitute.co.in)

Abstract: Multimodal Learning Analytics (MMLA) solutions aim to provide a more holistic picture of a learning situation by processing multimodal educational data. Considering contextual information of a learning situation is known to help in providing more relevant outputs to educational stakeholders. However, most of the MMLA solutions are still in prototyping phase and dealing with different dimensions of an authentic MMLA situation that involve multiple cross-disciplinary stakeholders like teachers, researchers, and developers. One of the reasons behind still being in prototyping phase of the development lifecycle is related to the challenges that software developers face at different levels in developing context-aware MMLA solutions. In this paper, we identify the requirements and propose a data infrastructure called CIMLA. It includes different data processing components following a standard data processing pipeline and considers contextual information following a data structure. It has been evaluated in three authentic MMLA scenarios involving different cross-disciplinary stakeholders following the Software Architecture Analysis Method. Its fitness was analyzed in each of the three scenarios and developers were interviewed to assess whether it meets functional and non-functional requirements. Results showed that CIMLA supports modularity in developing context-aware MMLA solutions and each of its modules can be reused with required modifications in the development of other solutions. In the future, the current involvement of a developer in customizing the configuration file to consider contextual information can be investigated.

Keywords: ad-hoc development, reusability, Separation of Concerns, SAAM
Categories: H.3.1, H.3.2, H.3.3, H.3.7, H.5.1
DOI: 10.3897/jucs.84558

1 Introduction

The purpose of Learning Analytics (LA) is to understand and improve learning outcomes. LA collects, analyzes, and presents data about learning process and the environment where it occurs [Siemens and Long, 2011]. In a decade of LA, the most common harnessed data source is the digital logs (some authors call it "mono-modal LA") [Bayoudh et al., 2021]. For example, the logs of a Learning Management System (LMS) stores learners' digital traces from their activities performed in the LMS. Results from log analysis lead to decisions that can help to optimize environment and improve learning experience. However, what happens outside the LMS environment is not considered in mono-modal LA [Ochoa et al., 2017]. Hence, mono-modal LA tools present a partial picture of a learning situation by covering only the digital space and only one modality; they do not consider other modalities available in digital and physical spaces of a learning situation [de Silva Joyce and Feez, 2018].

Multimodal Learning Analytics (MMLA) provides a more holistic picture of a learning situation by harnessing heterogeneous datasets from multiple modalities (e.g., audio, video, logs, eye-gaze, etc.) [Blikstein, 2013]. Some MMLA researchers focus on the multimodality of learning (i.e. visual, auditory, reading/writing, and kinesthetic) [Schuller, 2018, Chan et al., 2020, Schneider et al., 2018], while other researchers refer to the multimodality of the data, sources, and its analysis (i.e., different data sources like LMS logs, classroom observations, and audiovisual recordings) [Ochoa et al., 2022, Praharaj et al., 2022, Alwahaby et al., 2022]. In our case, we operationalize modalities as data sources about learning that can be tapped from the digital and physical spaces of a learning situation to collect multimodal educational data [Ochoa et al., 2022]. MMLA processes multimodal educational data and represents the multimodal experience of learning to the learners and teachers.

The current development of MMLA as a field does not allow educational practitioners, like teachers and school principals, to practice and adopt MMLA on their own in their naturalistic educational settings [Patnaik et al., 2021, Hoyos and Velásquez, 2020]. One of the main reasons behind this limited adoption is related to the availability of very few MMLA solutions [Patnaik et al., 2021, Hoyos and Velásquez, 2020]. Since the inception of MMLA, most of the research projects were trying to understand teaching and learning practices in controlled environments and contributed with some tailored solutions [Di Mitri et al., 2018, Prieto et al., 2019]. In the last couple of years, a few MMLA research projects have widened their scope and started exploring authentic MMLA situations by involving cross-disciplinary stakeholders like teachers, students, researchers, and developers [Yan et al., 2022]. However, we are still far away from MMLA solutions that can be adopted in naturalistic educational settings because developers find severe challenges in developing MMLA solutions for authentic MMLA situations.

There are three main reasons that complicate the development lifecycle of MMLA solutions [Schneider et al., 2018]. First, the data-intensive nature of MMLA solutions imposes high technical complexity in their development. Second, to start the development of a solution, cross-disciplinary stakeholders like a teacher, researcher, and software developer need to communicate to specify, and systematically document, *data requirements specification*. Third, solutions need to consider Contextual Information (CI) from the design and enactment of the learning situation to deliver a context-aware analysis. For example, the lesson may start five minutes later than designed, and there may be a lot of noisy audio data as students come in and the classroom is setup, which may be interpreted as part of the learning activity, when it is not. These three reasons lead to the current practice of MMLA development, which is mostly ad-hoc and opportunistic [Schneider et al., 2018]. This kind of software development practice increases the costs and risk of failure in design decisions [Wright, 2012], raising the need for a data infrastructure that can support the development of context-aware MMLA solutions [Ouhaichi et al., 2021]. Moreover, the current opportunistic practice adds burden to the developers where they need to develop each of the components of an MMLA solution from scratch every time [Di Mitri et al., 2019b]. Hence, the needed data infrastructure should also support developers in reusing and modifying existing lines of code.

To support the development of context-aware MMLA solutions and enable more reusability in MMLA development, we propose a data infrastructure called **C**ontext-aware **I**nfrastructure for **M**ultimodal **L**earning **A**nalytics (CIMLA). It includes a software architecture called **P**rocessing **A**rchitecture for **M**ultimodal **E**xperience of **L**earning (PAMEL). Our focus is on two points. First, how to follow a standard data processing pipeline like the Data Value Chain (DVC) [Miller and Mork, 2013] to showcase how multimodal educational data can be prepared, organized, and fused. Second, how to consider CI of a learning situation - involving multiple stakeholders- from classroom to develop context-aware MMLA solutions.

In order to evaluate the modularity and the modifiability of CIMLA, we have identified the requirements for CIMLA from the MMLA literature. For each of the identified requirements, CIMLA components have been designed following Separation-of-Concerns (SoC) architectural design principle [Hürsch and Lopes, 1995]. Further, components have been technically instantiated and a reference implementation has been developed. Finally, CIMLA has been evaluated in three authentic MMLA scenarios for the assessment of functional and non-functional requirements using the Software Architecture Analysis Method (SAAM) [Kazman et al., 1994].

The rest of the paper is structured as follows: section 2 presents the state of the art related to data infrastructures in MMLA; section 3 describes the CIMLA infrastructure. Later, an evaluation study is presented followed by a discussion section. We conclude the paper by summarizing the paper's main contributions to the field and highlighting future work in this line of inquiry.

2 Existing MMLA Infrastructures

In our prior work we reviewed nine existing MMLA data infrastructures in orden to understand current state of the art [Shankar et al., 2018]. We used DVC - a conceptual tool that involves seven standard data processing activities to mine information from raw data [Miller and Mork, 2013] - as a conceptual framework for our review. None of the nine proposals followed all the seven data processing activities defined by the DVC. Moreover, most of them followed ad-hoc and opportunistic development approach to process multimodal educational data rather than following any standard data processing pipeline [Schneider and Blikstein, 2015]. This approach in early design decision leads

MMLA developers to develop each of the solutions from scratch due to their specific selection of data processing activities in the pipeline [Kim et al., 2018].

To the best of our knowledge, nine new data infrastructures have been proposed in MMLA after our review [Schneider et al., 2018, Munoz et al., 2018, Tamura et al., 2019, Ciordas-Hertel et al., 2019, Huertas Celdrán et al., 2020, Camacho et al., 2020, Domínguez et al., 2021, Serrano Iglesias et al., 2021, Slupczynski and Klamma, 2021]. The main focus of eight of these infrastructures is to solve other aspects rather than following a standard data processing pipeline (for example, educational aspects - individual learner's engagement [Camacho et al., 2020], smart learning environments [Serrano Iglesias et al., 2021], inclusive learning [Tamura et al., 2019], oral presentations [Munoz et al., 2018], and customizable learning experience [Schneider et al., 2018]; technical aspects - dynamic deployment of MMLA instances in smart classroom [Huertas Celdrán et al., 2020], using AI and blockchain [Slupczynski and Klamma, 2021], and wearables [Camacho et al., 2020]; or adoption aspects - at institutional level [Domínguez et al., 2021]). Only one infrastructure is focused on following a standard data processing pipeline, but that is from the lens of the 3Vs of Big Data (Volume, Velocity, and Veracity) to optimize the scalability and performance issues of a technical solution [Ciordas-Hertel et al., 2019]. Hence, the limitation highlighted in our previous review remains the same even though nine new infrastructures have been proposed: none of the data infrastructures in MMLA follow a standard data processing pipeline while processing multimodal educational data.

We reviewed 18 data infrastructures in MMLA (nine from our last review [Di Mitri et al., 2017, Muñoz-Cristóbal et al., 2018, Fiaidhi, 2014, Domínguez and Chiluiza, 2016, Ruffaldi et al., 2016, Harrer, 2013, Berg et al., 2016, Wagner et al., 2011, Segal et al., 2017] and nine proposed after our review [Schneider et al., 2018, Munoz et al., 2018, Tamura et al., 2019, Ciordas-Hertel et al., 2019, Huertas Celdrán et al., 2020, Camacho et al., 2020, Domínguez et al., 2021, Serrano Iglesias et al., 2021, Slupczynski and Klamma, 2021]). We started our review by finding out how many learning tools were supported in the proposals. Further, we analyzed which data processing activities of DVC [Miller and Mork, 2013] were included in their data processing pipeline. However, we did not consider the first ***collect and annotate*** and the last ***make decisions*** activities in review because the first activity is performed by default in every proposal whereas the last activity is rarely discussed in any proposal because decision-making is a long-term activity and requires rigorous investigation from multidisciplinary lens before making any claim. Another aim of the review was to understand whether CI (from both design and enactment phases) [Alhadad and Thompson, 2017] was considered while processing MMLA data. The list of the involved stakeholders (teacher, researcher, and developer) who specified the *data requirements specification* were also detected [Shankar et al., 2020]. The summary of the review can be found in Table 3 of Appendix 6.

The most represented activity is ***analysis*** whereas ***visualization*** is the second. ***Preparation*** is used in 10 whereas ***organization*** and ***fusion*** are available in nine proposals. Nine out of 18 proposals did not consider neither design- nor enactment-related CI. We detected an interesting pattern: six out of nine previously reviewed infrastructures [Shankar et al., 2018] did not consider any type of CI whereas there were only three in case of new proposals. This pattern may indicate that the need for context-aware MMLA solutions is starting to be recognized. Finally, we can see that the most involved stakeholder is the researcher while the least involved is the teacher. There was only one proposal where all the three stakeholders were involved [Muñoz-Cristóbal et al., 2018]. One of the reasons behind the limitation of not involving all the stakeholders is because of the cross-disciplinary stakeholders communication [Shankar et al., 2020].

During the communication to specify the *data requirements specification*, each of the stakeholders use their domain-specific vocabularies and do not use any boundary object for communication [Shankar et al., 2020].

From this review we detected the need of data infrastructures in MMLA that cope with authentic MMLA scenarios that include cross-disciplinary stakeholders. Moreover, they should process MMLA educational data by following all the data processing activities of a DVC and consider design- as well as enactment-related CI.

3 CIMLA

This section starts by presenting the requirements for CIMLA. The requirements that define the working components of CIMLA are called "functional requirements", whereas those that define its quality features are called "non-functional requirements". CIMLA is explained in the next subsection. The section ends by presenting the technical instantiations of CIMLA components.

3.1 Requirements for CIMLA

Figure 1 depicts a potential authentic MMLA scenario where multiple cross-disciplinary stakeholders like teachers, researchers, and developers are involved and supported by CIMLA. Teachers and students participate and enact in a learning situation whereas researchers look for evidence that can represent a multimodal experience of learning. They all use different tools in digital and physical spaces. Multimodal educational data is collected from the tools used, obtaining a set of heterogeneous datasets. Further, CI with *data requirements specification* is structured and organized following a contextualized data model called ***CDM4MMLA*** [Shankar et al., 2022]. Structured and organized CI is instantiated by the developer to guide the data processing activities in the data processing engine. Finally, multimodal educational data, collected and stored in heterogeneous datasets, is processed and analyzed results are fed back to the stakeholders.

3.1.1 Functional requirements

The tools used in the learning situation collect multimodal educational data and store them in their underlying data repositories. The first requirement relates to the need of accessing different data repositories to extract heterogeneous datasets (***REQ1*** in Figure 1 and Table 1) [Di Mitri, 2020]. For example, collected students' responses to the questions are saved on *Google Drive* whereas digital traces of students are stored on *LMS server*. Therefore, CIMLA should be able to access the datasets from *Google Drive* and *LMS Server*. Once the heterogeneous datasets are accessed, their file formats need to be unified in a single file format (***REQ2***) [Di Mitri, 2020, Di Mitri et al., 2019a]. For example, students' responses data files are stored in *XLSX*, whereas log data files are stored in *CSV* file format. Hence, students' responses data files need to be unified in CSV format.

Another requirement emerges from the need to consider structured and organized CI following CDM4MMLA into the data processing pipeline (***REQ3***) so that context-aware data processing can take place [Eradze et al., 2020, Shankar et al., 2020, Shankar et al., 2019, Eradze, 2020]. Further, multimodal educational data stored in heterogeneous datasets should be processed using a standard data processing pipeline (***REQ4***) [Shankar et al., 2020, Di Mitri et al., 2019a, Shankar et al., 2019, Eradze, 2020, Di Mitri et al.,

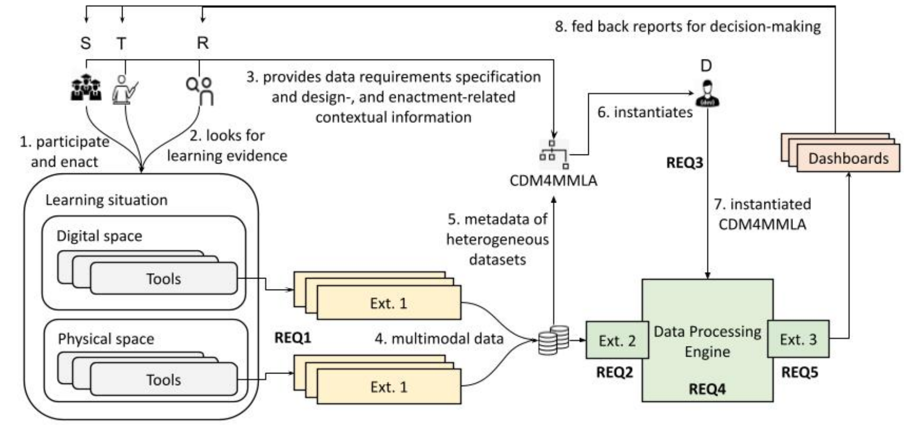

Figure 1: A potential authentic MMLA scenario supported by CIMLA [S- Student, T-Teacher, R- Researcher, Ext- Extension]

2019b, Hassan et al., 2021]. Finally, the output data file generated after processing multimodal educational data should be exported in a set of file formats (***REQ5***) [Di Mitri, 2020, Di Mitri et al., 2019a, Shankar et al., 2019]. For example, the output data file from the processing of students' responses and LMS log would be in *CSV* file format by default because the input data files were in *CSV*. However, if the stakeholders need the file in *XLSX*, then CIMLA should be able to export it in *XLSX*.

3.1.2 Non-functional requirements

In addition to the five functional requirements, there are two non-functional requirements for CIMLA. In the current practice of developing an MMLA solution (ad-hoc and opportunistic approach), the exploitation of multimodal educational data from its raw form to the useful information is treated in one process [Di Mitri et al., 2018, Shankar et al., 2018]. This practice hinders MMLA developers when developing other MMLA solutions with respect to the data-intensive nature of MMLA ecosystems like veracity, velocity, and volume of datasets [Huertas Celdrán et al., 2020, Shankar et al., 2020]. Therefore, it is required to break down the process involved in exploiting multimodal educational data into modules (***REQ6***) [Di Mitri et al., 2019a, Shankar et al., 2019, Hassan et al., 2021, Spikol et al., 2018]. Second, in the current practice of MMLA development, most of the components of an MMLA solution are developed from scratch every time [Shankar et al., 2018, Huertas Celdrán et al., 2020]. Therefore, it is required for CIMLA to support modifiability (***REQ7***). Thus, whenever a new MMLA solution is being developed, all the components do not need to be developed from scratch [Shankar et al., 2020, Di Mitri et al., 2019a, Shankar et al., 2019].

3.2 Design of CIMLA

CIMLA with its three components - ***Fetchers***, ***Configuration File***, and ***PAMEL Manager*** in the context of an authentic MMLA scenario can be seen in Figure 2.

Tag / Category	Requirement	References
[REQ1] [F]	Fetch datasets from different data repositories to access heterogeneous datasets that store multimodal educational data	[Di Mitri, 2020]
[REQ2] [F]	Unify all the datasets into a single file format	[Di Mitri, 2020, Di Mitri et al., 2019a]
[REQ3] [F]	Consider structured and organized CI in the data processing pipeline to support context-aware data processing	[Eradze et al., 2020, Shankar et al., 2020, Shankar et al., 2019, Eradze, 2020]
[REQ4] [F]	Process multimodal educational data using a standard data processing pipeline	[Shankar et al., 2020, Di Mitri et al., 2019a, Shankar et al., 2019, Eradze, 2020, Di Mitri et al., 2019b, Hassan et al., 2021]
[REQ5] [F]	Export output data file in a set of file formats, so that stakeholders can get the data file in their required format	[Di Mitri, 2020, Di Mitri et al., 2019a, Shankar et al., 2019]
[REQ6] [NF]	Support modularity in data processing, so that the complexity of multimodal educational data processing can be broken down	[Di Mitri et al., 2019a, Shankar et al., 2019, Hassan et al., 2021, Spikol et al., 2018]
[REQ7] [NF]	Support modifiability in different components, so that developers do not need to develop them from scratch	[Shankar et al., 2020, Di Mitri et al., 2019a, Shankar et al., 2019]

Table 1: A list of functional and non-functional requirements for CIMLA [F - Functional, NF - Non-Functional]

To start with the architectural design of CIMLA, we followed a ***development architectural viewpoint*** [Woods and Rozanski, 2005] because MMLA developers are the main stakeholders to be supported by CIMLA. Moreover, we followed the ***Separation-of-Concerns (SoC)*** architectural design principle to divide the different interests involved in the challenge of developing a context-aware MMLA solution [Hürsch and Lopes, 1995]. As a result, we designed the candidate architecture called ***PAMEL*** (Processing Architecture for Multimodal Experience of Learning) and evaluated its appropriateness and components following the developer's architectural view in four scenarios [Shankar et al., 2019]. With the lessons learnt from the initial evaluations of fitting PAMEL into the scenarios, we refined the initial PAMEL proposal to its current form (see Figure 3).

The first component of CIMLA is called ***Fetchers***. It is designed to access the datasets from data repositories (*REQ1*) that store the heterogeneous datasets collected through

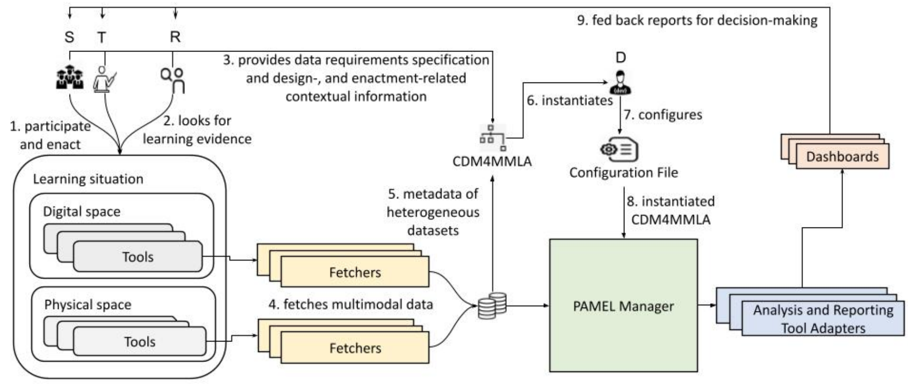

Figure 2: ***C****ontext-aware* ***I****nfrastructure for* ***M****ultimodal* ***L****earning* ***A****nalytics (****CIMLA****)*

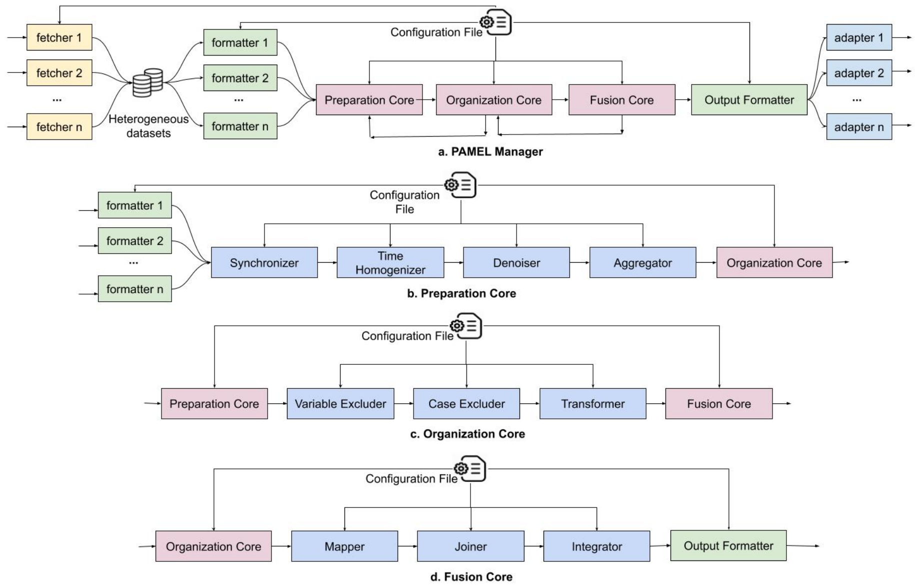

Figure 3: ***P****rocessing* ***A****rchitecture for* ***M****ultimodal* ***E****vidence of* ***L****earning (****PAMEL****)*

different tools used in a learning situation. Second, the PAMEL Manager is the central component of CIMLA and comprises the***Formatter***, ***Preparation Core***, ***Organization Core***, ***Fusion Core***, and the ***Output Formatter*** (see Figure 3a). The *Formatter* is designed to unify all the dataset file formats into a single file format (*REQ2*). The *Output Formatter* is designed to export the final dataset into the stakeholder's required file format (*REQ5*). Other three core components (*Preparation*, *Organization*, and *Fusion*) are designed to partially process multimodal educational data using a standard data processing pipeline, the DVC as per [Miller and Mork, 2013] (*REQ4*). Six data processing components of CIMLA are designed to get the required piece of *CI* from the ***Configuration File*** (refer to Figure 3a) so that context-aware data processing can be done (*REQ3*).

It is noteworthy that standard data processing pipelines like DVC [Miller and Mork, 2013, Shankar et al., 2020] involve six main data processing activities *-Preparation, Organization, Fusion, Analysis, Visualization, and Decision-making*-, out of which CIMLA tackles the first three activities. Fusion and Decision-making are the two key milestone activities highlighted in MMLA [Chango et al., 2022]. The Fusion activity integrates the heterogeneous datasets into one dataset and wipes out the complexity of heterogeneity for the rest of the data processing activities (i.e., analysis, visualization, and decision-making) [Mu et al., 2020]. On the other hand, Decision-making is the second key activity that closes the loop of MMLA for stakeholders. Despite its relevance and importance, its enactment requires rigorous interpretation from the involved stakeholders using a multidisciplinary lens before closing the MMLA loop. Hence, we only considered the first three milestone activities and kept rest of them (Analysis, Visualization, and Decision-making) out of the scope of this paper.

The *Preparation Core* involves four sub-components: ***Synchronizer, Time Homogenizer, Denoiser and Aggregator*** (see Figure 3b). The *Synchronizer* establishes consistency in each of the datasets so that there is no inconsistency in timestamps due to network latency-like issues. The *Time Homogenizer* unifies the timestamps among datasets. For example, *Google server* uses *GMT +2* to store the timestamp in students' responses dataset whereas some *LMS servers* use *GMT +3* to store the timestamp in their logs. For dealing with noises in the datasets, the *Denoiser* is designed. Finally, the *Aggregator* combines the data files of one dataset. For example, if each student uses a separate copy of a *Google Form* to respond to a questionnaire, the number of *XLSX* files would be equal to the number of students who respond. Hence, the *Ággregator* merges all the data files of students' responses in one single dataset.

The *Organization Core* involves three sub-components: ***Variable Excluder, Case Excluder, and Transformer*** (see Figure 3 3c). The *Variable Excluder* removes the variables of a dataset that are not required in the data processing pipeline. Similarly, cases that are not needed in the data processing pipeline can be removed by the *Case Excluder*. Finally, the *Transformer* changes the structure of datasets into the required format. For example, if the rows need to be transformed into columns and vice versa like a matrix skew function, then the *Transformer* is the responsible component.

The *Fusion Core* involves three sub components: ***Mapper, Joiner, and Integrator*** (see Figure 3d). The *Mapper* sets up the relationship among correlated variables across heterogeneous datasets. Once the datasets are related through correlated variables, the *Joiner* connects the heterogeneous datasets, which can be finally combined by *Integrator* component.

3.3 Technical instantiation of CIMLA components

CIMLA components are technically instantiated through a ***User Interface (UI)*** and a ***skeleton codebase*** that are described as follows.

3.3.1 User Interface

On the one hand, the teacher is responsible for providing the required *CI* for the context-aware processing of multimodal educational data [Muslah and Ghoul, 2019, Shankar et al., 2020]. The *Configuration File* of CIMLA accepts the input of *CI* (*REQ3*) that is further stored in a JSON [1] (JavaScript Object Notation) file. On the other hand, the JSON

[1] https://www.json.org/json-en.html

file [2] is developed and customized by the developer that follows a machine-readable file structure. Since we cannot expect teachers to have advanced software engineering skills, there is a need for an interface through which the teacher can provide structured and organized *CI*.

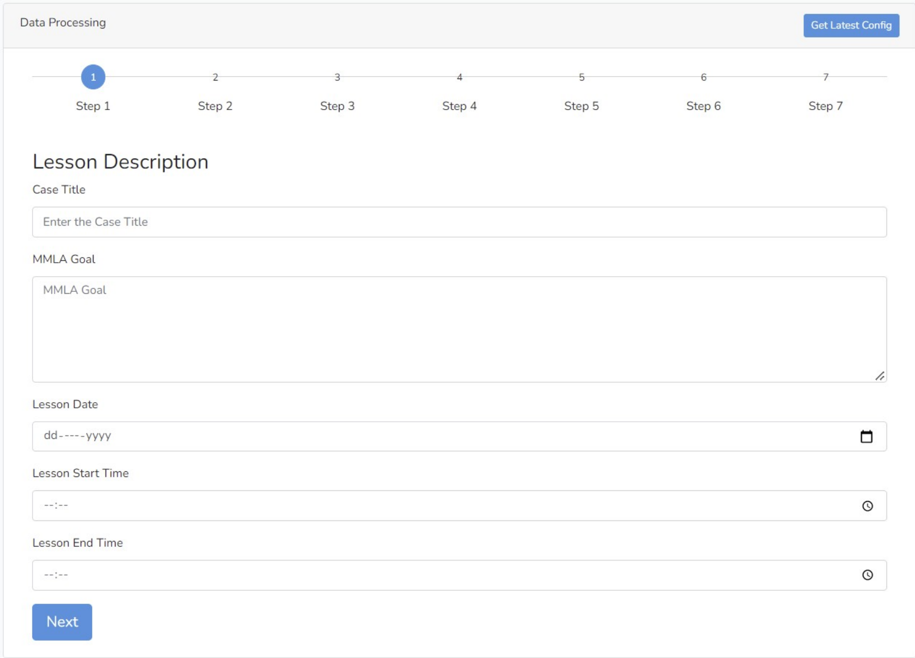

Figure 4: User interface

An UI has been developed to meet this requirement [3]. It includes a seven-step process flow that instantiates the interface representation of structured and organized CI following the CDM4MMLA [Shankar et al., 2022] (see Figure 4). In each of the steps, a set of UI components allows the teacher to enter the specific piece of information. Most of the UI components are based on string variables so that the teacher can use their own way of writing the text and does not have a text length barrier. Further, the provided CI is stored in a JSON file. The reason behind the selection of JSON format is because of its enables the storage of semi-structured data. Finally, the developer intervenes by taking the semi-structured data (i.e., teacher's inputs of CI) from the JSON file and transforms them to structured data (i.e., machine-readable rules). For the transformation process, the developer goes through each of the strings in the JSON, extracts the main rules from the unstructured text, and assigns them back to the JSON file (see 5).

[2] A sample JSON file: https://tinyurl.com/mr27mk4w
[3] UI: http://63.35.249.219/

```
"denoising": {
  "requires": "NO",
  "removeDuplicates": true,
  "rule": "Remove rows from 20 to 25 because they are outside lesson duration"
}
```

a. Provided rule in unstructured format by a teacher through the UI

```
"denoising": {
  "requires": "NO",
  "removeDuplicates": true,
  "rule": {rowsRemoval: "20-25"}
}
```

b. Transformed rule in structured format in configuration file after the developer intervention

Figure 5: Developer's intervention to customize contextual information in Configuration File

3.3.2 Skeleton codebase

Whenever a developer starts the implementation directly from the architectural form, in most of the cases, the implementation code is decoupled from the architecture [Aldrich et al., 2002]. This decoupling leads to inconsistencies, confusion, and violating properties of the architecture, in addition to hindering the evolution of the solution. To remove such decoupling issues and to provide a quick start for developers in technically implementing CIMLA and its components, we developed a template software code. A ***Skeleton Programming*** is a code-based approach that unifies software architecture with implementation and ensures that the implementation conforms to the architectural constraints [González-Vélez and Leyton, 2010]. Hence, we also adopted the *Skeleton Programming* and developed a skeleton codebase [4] to technically instantiate the six data processing components of CIMLA from its architectural form present in PAMEL (refer to Figure 3).

In the skeleton codebase, six components -*Fetchers, Formatter, Preparation Core, Organization Core, Fusion Core, and Output Formatter*- have been implemented at a high level of abstraction in a form of classes. The classes provide the seed to implement the different components. Moreover, different functions are included in the classes for describing the functionality of the components and their sub-components. Further, each function includes a set of parameters in its definition. The functions relate to the functional behavior of the components and their argument values need to be accessed from the *JSON file* generated through the *Configuration File* component of CIMLA. Finally, every function includes the return statement that represents the expected output data file after the execution of the function. For example, Figure 6 illustrates a class called 'FETCHER' that has a function called 'fetchFromGDrive'. The goal of this function is to fetch a data file from *Google Drive*. The parameters are the credentials to access Google Drive, the data source to which the fetched data file belongs, and the sequence number of the data file in the specified data source. The arguments for these parameters need to be accessed from the *JSON file* and passed when calling the function. After the execution, this function should return the data file fetched from *Google Drive*.

[4] Skeleton codebase available at: https://tinyurl.com/2p8caw9b

```
class FETCHER
{
        //Specialized functions
        function   fetchFromGDrive(string   CREDENTIALS,   string   DataSource,   int
DataFile_n)
        {
        …
        return DataSource_DataFile_n.xlsx
        }
        function   fetchFromLocalFileSystem(string   PATH,   string   DataSource,   int
DataFile_n)
        {
        …
        return DataSource_DataFile_n.csv
        }
…
        //Generalized function that needs to be customized
        function    fetchFrom'DataRepository_x'(string    'ACCESS_MECHANISM    /
ADDRESS', string 'DataSource_y', int DataFile_n)
        {
        return DataSource_DataFile_n.'extension'
        }
}
```

Figure 6: Fetcher class in skeleton codebase

4 Evaluation study

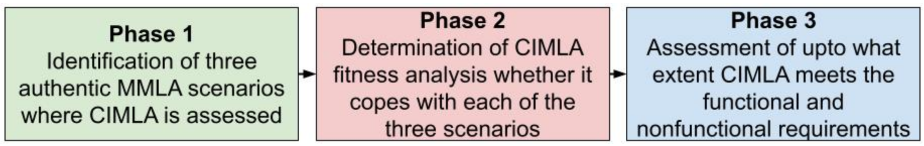

Figure 7: Overall evaluation of CIMLA using SAAM

For the overall evaluation (i.e., to what extent CIMLA and its components meet the requirements detailed in Section 3), we followed the SAAM evaluation method [Kazman et al., 1994]. SAAM is considered a canonical method for the analysis of architectures following the SoC design principle to support the development of computer-based systems. To start with the process of overall evaluation using SAAM, a number of scenarios need to be identified (see Figure 7). Second, for each of the scenarios, it should be determined whether the infrastructure coped with the scenarios. Finally, stakeholders involved in the scenarios are requested to assess to what extent the infrastructure meets the requirements [Kazman et al., 1996].

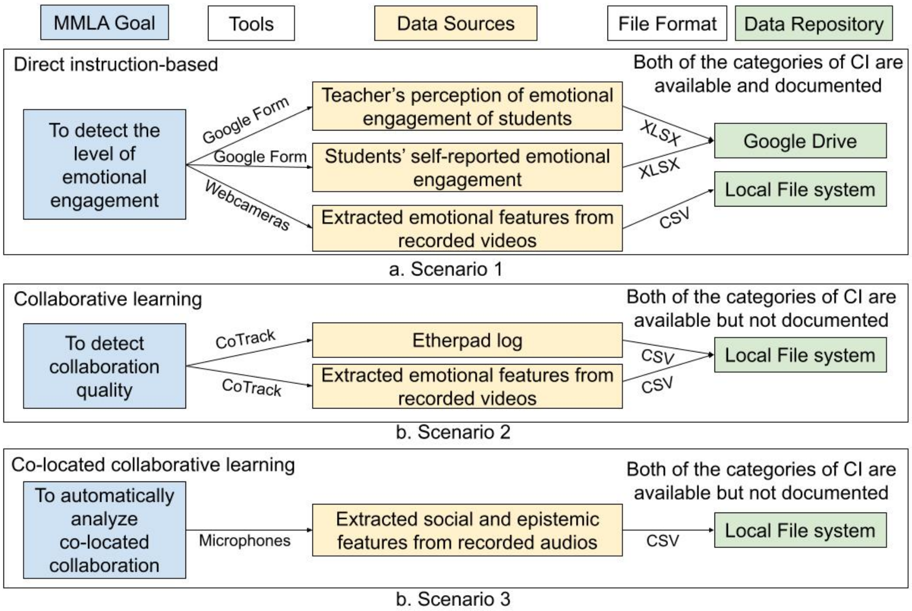

Figure 8: An overview of three scenarios with respect to their learning settings [CI-Contextual Information]

4.1 *SAAM Phase 1:* Scenario identification

Three MMLA scenarios (***S1, S2, and S3***) were identified using a number of selection criteria. These criteria were not selected due to SAAM, but because our goal is to propose an infrastructure that supports authentic MMLA settings. The first selection criteria was that the scenario should include a learning situation that was planned and enacted in a real classroom setting [Schneider et al., 2018]. Second, CI from design as well as enactment phases should be available [Ouhaichi et al., 2021]. Third, multiple cross-disciplinary stakeholders like a teacher, researcher, and developer should be involved in the scenario [Shankar et al., 2020]. And the fourth and last criteria is that at least two stakeholders should be willing to participate in the evaluation study. One stakeholder should be either a teacher or researcher who can provide CI [Schneider et al., 2018]. The other stakeholder should be a developer who can discuss the development approach of the scenario [Ouhaichi et al., 2021]. Figure 8 summarizes the technology and analytics characteristics, and Table 2 some other contextual features of the scenarios.

4.1.1 *S1:* Detecting the level of emotional engagement

This MMLA scenario took place in *Birla Institute of Applied Sciences* (India) to detect the level of emotional engagement of students in a course on *'Computer Organization and Architecture'*. The course was planned in *40 lessons* over a semester (from August to December, 2021) for third-year computer science bachelor students. The lessons were planned to be enacted in a physical classroom setting but due to COVID-19 pandemic and its disruptions, all of the 40 learning situations were enacted in a digital classroom

Parameter	S1	S2	S3
Number of involved stakeholders (T - Teacher, R - Researcher, D - Developer)	(T - 1, R - 1, D - 2)	(T - 1, R - 1, D - 1)	(T - 1, R - 1, D - 1)
Status of the MMLA solution development at the time of introducing CIMLA	Not developed	Developed	Developed
Does the involved developer have MMLA development experience?	No	Yes	Yes

Table 2: A summary of the three identified scenarios

using *Google Meet*. Each of the lessons was planned for 50 to 55 minutes and included three learning activities. Every activity was planned to deliver a concept for 15 minutes with the teacher's direct instruction and a two-minute break at the end of the activity.

Once a few lessons were enacted, the teacher (*S1.T1*) doubted about his own perception of the students' emotional engagement in the digital classroom. He highlighted that a single screen is used to present the slide and also for looking at an average of 30 frames representing students' live videos. He discussed this issue with an MMLA researcher (also one of the authors of this article) and expressed the wish to have other data sources that could complement his observation of emotional engagement. The researcher suggested two data sources. First, students' self-reported emotional engagement [Fuller et al., 2018] and second, emotional features were extracted automatically from recorded videos of students using Microsoft Emotion API [5].

For the self-reported data, students were asked to fill in a Google Form [6] to indicate their level of emotional engagement. The form includes two questions taken from [Fuller et al., 2018] that are designed and evaluated to report emotional engagement. Moreover, the teacher also reported his own observation about the level of students' emotional engagement using the same two questions through a different Google Form [7]. Finally, students were asked to record their videos for the entire lesson through their personal devices' webcams. The informed consent [8] was collected in advance from the students for participating in the study. Later, emotional features (Anger, Contempt, Disgust, Fear, Happiness, Neutral, Sadness, and Surprise) were extracted from those videos. Based on this design, a multimodal educational dataset was collected from three lessons [9]. Finally, two professional developers (***S1:D1 and S1:D2***) who are also fourth-year computer science bachelor students of the same institution were hired to develop the MMLA solution. They did not have any prior experience on MMLA development or any knowledge related to LA at the time of hiring.

4.1.2 *S2:* Detecting collaboration quality

This MMLA scenario is a part of a larger research project that aims to support teachers in monitoring and feedback in *Computer-Supported Collaborative Learning (CSCL)*

[5] https://azure.microsoft.com/en-us/services/cognitive-services/
[6] Student's Form available at: https://forms.gle/odzxCtVqJdMUVzpm7
[7] Teacher's Form available at: https://forms.gle/2H4ETy7D7LXceqUB8
[8] Informed consent available at: https://forms.gle/Ck4BVQvfDsHEzrRT8
[9] Access the datasets here: https://tinyurl.com/ytm5925v

activities [Chejara et al., 2020]. As one of the outcomes, researchers plan to support teachers by showing them a dashboard that graphically represents the collaboration quality of participating groups. In this project, the scenario S2 was planned by a teacher and a researcher at *Tallinn University* (Estonia). Multimodal educational data was collected from two data sources: Etherpad logs and video recordings. Etherpad [10] is a collaborative text editor and its logs were processed to extract the number of characters added or deleted by an individual member of a collaborative group. Similarly, features that could be indicative of ongoing collaboration quality like action units, emotion, head pose, and mouth region were extracted from the recorded videos.

Four CSCL sessions were enacted in Estonian schools. The participating students who were from 9th, 10th, and 12th grades, signed an informed consent before participating. The language of the sessions were either in *English* or *Estonian* language. The developer (***S2.D1***, also one of the authors of this paper and involved researcher in this scenario) had already developed a prototype MMLA solution following his own an ad-hoc development approach prior to introducing CIMLA [Chejara et al., 2021]. The developer highlighted a few issues in his development approach. First, all the components of the solution were developed in a single program file that took more time than expected in modifying any component of the code. Second, the software code was organized in a very tightly-coupled manner that lead to complexity for any other developer to understand and reuse. As CIMLA was designed with modularity principles at the center, *S2.D1* was suggested to use it as an infrastructure in the context of *S2*.

4.1.3 *S3:* Automatically analyzing co-located collaboration

This MMLA scenario took place at the *Open University of the Netherlands*. It is also part of a larger research project that aims to automate the *analysis of co-located collaboration* [Praharaj et al., 2022]. This scenario was designed by a teacher and a researcher, and was further enacted in *14 learning situations*. Each of the situations included only one group that involved five participants (different university staff members and students) and the communication language was *Dutch*. The participants had to design a learning activity using the *Fellowship of Learning Activity and Analytics (FoLA2) framework*[11].

In co-located collaboration, the participants collaborate *face-to-face*, thus sharing a common social and epistemic space. The social space involves non-verbal indicators of collaboration like turn-taking and speaking time. The epistemic space involves verbal indicators of collaboration like content discussion and log data about the content of discussion [Praharaj et al., 2022]. To cover both of these spaces, the audio data was collected from the aforementioned 14 learning situations. Each of the participants signed the informed consent and were wearing clip-on microphones with their own audio recorders. At the end of the situation, audio files from all the recorders were collected, stored, aggregated, and processed for collaboration analytics. A prototype MMLA solution for this scenario had already been implemented by the developer (***S3.D1***, who is also the main researcher involved in *S3*) before getting introduced to CIMLA. The development approach was also ad-hoc like *S2*. Hence, the highlighted issues were similar to those of *S2*, and CIMLA was also suggested to *S3.D1* for using and assessing as an infrastructure in the context of *S3*.

[10] https://etherpad.org/

[11] https://edutec.science/fellowship-of-the-learning-activity-learning-analytics-fola2/

4.2 *SAAM Phase 2:* Fitness Analysis of CIMLA

This and the following subsection presents the evaluation approach of CIMLA using a mixed-methods approach (see Figure 9).

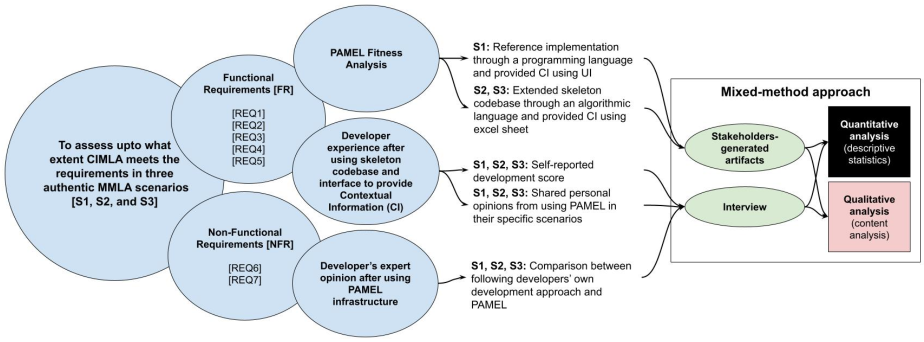

Figure 9: The mixed-methods approach used for evaluation

4.2.1 Method

To start with phase 2 of SAAM, the CIMLA components and technically instantiated objects (i.e., UI and Skeleton codebase) were introduced to the MMLA developers involved in the three MMLA scenarios described above (*S1:D1, S1:D2, S2:D1, and S3:D1*). For this purpose, a commentary document file[12] was prepared that explains each of the CIMLA components, sub-components, and their workings. As explained above, in the case of S1, no MMLA solution existed prior to the introduction of CIMLA to the developers. Hence, the teacher (*S1.T1*) was asked to use a *UI* to provide the contextual information (CI). As the teacher was a first-time user of the UI, the commentary file was also provided to him. Further, both developers in S1 were asked to develop a context-aware MMLA solution by extending the skeleton codebase of PAMEL and considering the CI provided through the UI. The commentary file was also provided to them. *S1.D1* and *S1.D2* developed the solution by extending the skeleton codebase, linking the UI (front-end) with the codebase as back-end, and enabling the functionality to fetch rules from the JSON file (see Section 3). The end product was a working MMLA solution and a reference implementation [Martens and Vogten, 2005] of CIMLA[13]. Hence, in the case of S1, the fitness analysis was done *after* such reference implementation.

In S2 and S3, MMLA solutions had already been developed prior to the introduction of CIMLA. Developers had experience in developing MMLA solutions for that context and they had the CI of authentic learning situations. Although the CI in these two scenarios was not structured and organized in the way CIMLA prescribes (i.e., using the CDM4MMLA), they used it at their convenience. Therefore, unlike S1, teachers were not available to provide the CI in the UI. Similarly, developers were not asked to extend the skeleton codebase and link it as back-end code to a UI for the actual development

[12] Commentary file available at: https://tinyurl.com/2p8caw9b
[13] Reference implementation available at: https://github.com/shashi2y22/PAMEL-

of the MMLA solutions. However, the developers were asked to extend the skeleton codebase quickly from their past experience by following algorithmic language [Bauer and Wössner, 2012] rather than actual lines of code[14],[15]. Algorithmic language is a problem-oriented language, unlike programming languages, that follows fixed notations, structures, and sequences. Using an algorithmic language is faster and does not require specific programming language knowledge to use it [Bauer and Wössner, 2012]. We prepared a ***Google Sheet***[16] as an alternative to the UI, with the same input fields as S1's UI components, for the developers (S2.D1 and S3.D1) to enter the CI of their respective situations. These two strategic changes with respect to S1 were adopted due to the high expected cost (in terms of working hours of developers and teachers) of following the original S1 approach.

Once CIMLA was used by the stakeholders (*S1.D1, S1.D2, S1.T1, S2.D1, S3.D1*) in all the three authentic MMLA scenarios, the evaluation of CIMLA's fitness was done by analyzing the extended skeleton codebase files and the provided CI. Stakeholders were involved in a discussion with the researcher to analyze the fitness of the generated artifacts. The fitness of each of the CIMLA components was classified and tagged with four labels ***L1: Direct***, ***L2: Direct but with an intervention***, ***L3: Indirect***, and ***L4: Not Applicable*** [Kazman et al., 2003]. The label *L1* was assigned in cases when the stakeholders were able to use the specific component in a scenario on their own and without any required modification in the component. However, if the stakeholder needed an intervention even after going through the commentary file, either when using any CIMLA component or when providing the CI, then the assigned label was *L2*. The label *L3* was assigned in cases when the stakeholders could not use the technically instantiated objects in their provided form and required necessary modifications. Finally, the label *L4* was assigned if a specific CIMLA component was not required in a particular scenario.

4.2.2 Results

Table 4 in Appendix 6 summarizes the results of the fitness analysis. *L1* was assigned *29* times whereas *L4* was used *15* times out of a total of *57*. The count for *L2* is *12* and only one occurrence of *L3* was registered. The practical aspect of passing the arguments from the *JSON file* generated through the *Configuration File* component to the *Preparation, Organization, and Fusion Cores* in S2 and S3 required an intervention. S2:D1 and S3:D1 did not understand how the unstructured rules in an excel sheet could be used as arguments to the parameters of different functions. This issue was not faced in S1 because the unstructured rules provided through UI components were processed through the developer intervention to make them structured rules. Finally, the structured rules were saved in the *JSON file* that were further accessed for argument passing for different function calls (please refer to Figure 5). Other *L2* cases occurred mainly because of further required explanations in addition to the commentary file, to clarify CIMLA components and their workings.

There was only one case of *L3*. This occurred with the *Integrator* component in S2. Initially, the skeleton codebase used the term *Aggregator* instead of *Integrator* which was confused with the term *Aggregator* in the *Preparation Core*. Hence, S3:D1 suggested renaming the *Aggregator* component of *Fusion Core* to *Integrator*.

[14] Access to S2's Codebase: https://tinyurl.com/mr3d7p34
[15] Access to S3's Codebase: https://tinyurl.com/mwnpn9mw
[16] Shared spreadsheet: https://tinyurl.com/28wxnuet

4.3 *SAAM Phase 3:* Assessment of CIMLA

This subsection presents the assessment of the extent to which CIMLA meets the functional and non-functional requirements (please refer to Section 3).

4.3.1 Method

Developers of the three scenarios (*S1:D1, S1:D2, S2:D1, S3:D1*) were interviewed in a semi-structured manner to comment, from their development experience, on the functionality of the CIMLA components in terms of how well they could cope with the MMLA functionalities needed in each scenario. Three interview sessions post-development were conducted to interview each of the scenarios separately. The sessions lasted for an average of 90 minutes. Each of the interview sessions was structured in five sections to assess the five functional requirements of CIMLA (please refer to Section 3).

The first part of the interview was structured in five sections, in which a researcher initiated each section by explaining the requirement and the underlying CIMLA component that addresses it. Further, a question (*Did you experience the working of the specific component as explained?*) was posed to understand whether the component was working in the specific scenario after developing the component. Once remarks were made on the functionality of a specific component, we also wanted to understand how successful were the developers in developing the component through a development score reported on a five-point Likert scale (*1- Very Unsuccessful, 2- Unsuccessful, 3- Neither Successful Nor Unsuccessful, 4- Successful, 5- Very Successful*). Finally, at the end of each of the sections in the interview, we also asked an open-ended question to know the developer's reasoning behind the score. In cases where the components were not applicable, the participants were asked to score from the perspective of their development expertise.

To assess the non-functional requirements of CIMLA, the methodology was different in S1, compared to S2 and S3. In S1, the development approach was different because the MMLA solution had not been developed prior to the introduction of CIMLA. *S1.D1* and *S1.D2* developed anew the reference implementation of CIMLA, whereas in S2 and S3, developers had developed MMLA solutions previously, following their own approach. Therefore, a single methodology could not be followed to assess the modularity and modifiability features to support the developers. In S1, *S1.D1* and *S1.D2* were interviewed prior to the development of the solution *and* the introduction of CIMLA. In this *190-minute* semi-structured pre-development interview, the developers were presented with the scenario and the requirement to develop an MMLA solution. Further, they were given 30 minutes to think-aloud through its possible development approach if they had to develop the solution. Once both developers reached an agreement, both of them were asked to sketch their development approach for the next 30 minutes. Finally, they were asked open-ended questions to describe their development approach and the rationale behind it.

Post-development of the reference implementation in S1, developers were interviewed in the same interview session where they assessed the functional requirements. The interviewer shifted from functional to non-functional requirements assessment, by asking the developers to compare their development approach (selected at the time of the pre-development interview) with the CIMLA approach, to see if the modularity aspect came up in the discussion. In case modularity came up, further discussion was carried on to explore such modularity aspect. Further, they were presented with an ongoing issue in MMLA development: in most of the cases, developers need to develop most components of an MMLA solution from scratch. They were asked to express their expert opinion (through open-ended questions) about whether CIMLA dealt with this issue, i.e., to assess the modifiability aspect.

In S2 and S3, there were no pre-development interviews because the solutions were already developed prior to the introduction of CIMLA. Hence, in the same post-development interview session for the assessment of functional requirements, the developers were asked to compare and discuss the two development approaches: the developer's own ad-hoc approach through which they had developed the existing MMLA solutions for S2 and S3, and CIMLA's development approach. However, as already highlighted above, the developers did not re-develop the solutions following CIMLA - rather, they experienced it by extending the skeleton codebase and providing a situation's CI. The rest of the interview session was planned as described above for S1.

The interviews were conducted via *Skype* and all the interview sessions were recorded using the record feature of *Skype*. The recorded files were downloaded from Skype and stored in the local machine of the researcher which is a password-protected device. Moreover, a copy of the video files was also uploaded to *Google Drive* using the institutional *Tallinn University* account of the researcher. Further, the content analysis of the videos was performed following a bottom-up approach, by one researcher. This included inspecting the three recorded files to extract *quantitative scores* and *qualitative quotes* with respect to the seven requirements. The findings were tagged with the requirements and are presented in the following subsections.

4.3.2 Results of the assessment of functional requirements

Table 5 of Appendix 6 summarizes the quotes and the development score from the participants of three scenarios in the context of assessing the five functional requirements of CIMLA.

[REQ1]: The *Fetcher* component was designed for *REQ1*. The fitness analysis of CIMLA and developers' opinions after using *Fetchers* suggests that it has achieved its functionality. The development score is also either in the category of *successful or very successful* for all three scenarios. In the current reference implementation of CIMLA in S1, the *FETCHER* class has two working functions to access datasets from *Google Drive and Local File System*. These two functions within the class are independent of the rest of CIMLA components. Hence, they can be reused in other MMLA scenarios where datasets need to be fetched from these two kinds of data repositories.

[REQ2]: The developer scores for the *Formatter* component is in the category of *successful* in all the three scenarios. This suggests that the developers did not experience any problems in developing using this component. Two working functions already exist in the reference implementation to convert *XLSX and JSON* file formats to *CSV*. From the discussions on the functionality assessment, the quotes from developers also suggest that it has achieved its functionality. One developer questioned whether the *Formatter* is needed as a separate component or not. The logic given by the developer was that there are different libraries in data science languages that support the processing of datasets in a set of file formats; then, why do developers need to unify datasets in one format. This requirement of unification goes deeper into data mining [Barkhordari and Niamanesh, 2017] and has also been highlighted in MMLA literature [Di Mitri, 2020, Di Mitri et al., 2019a]. One of the main reasons behind the requirement of unification is that some libraries can support some of the data processing activities on a set of datasets but may not support all of them. Hence, different libraries need to be chosen in different data processing activities. However, if the datasets are unified beforehand then the selection

of libraries for data processing should be carefully planned just once [Barkhordari and Niamanesh, 2017].

[REQ3]: The developer scores and most of the qualitative quotes suggest that the requirement of considering the structured and organized CI in the data processing pipeline was achieved and successfully programmed. However, in its existing form of CIMLA and its reference implementation in S1, the *JSON file* was configured by the developer. This aspect was the most discussed throughout the interview sessions in S2 and S3. S2.D1 and S3.D1 highlighted the use of some kind of Artificial Intelligence (AI) techniques like Natural Language Processing (NLP), which could automate this process and remove the intervention of human developers. Another point that has been highlighted is the decoupling of the UI and back-end code to process multimodal educational data. In its current form, the UI is tightly coupled with the back-end code. This form has the limitation that inputs to the UI should be mirrored by extensions of the data processing functions.

[REQ4]: Aside from certain confusion regarding the terminology, the development score and quotes suggest that the *PAMEL Manager* achieved its functionality and was *very successfully* developed with working *Preparation, Organization, and Fusion* classes and their functions. The developers throughout the interviews mentioned the modular structure of *PAMEL Manager* that breaks down the data processing task into smaller modules. S1:D1 and S1:D2 highlighted that *the estimated complexity was brought down to easier programmable modules*. The developers in S2 and S3 wished that they had known and used this kind of data processing pipeline and modular structure, which would have led to *faster and more efficient* development of their MMLA solutions. They also emphasized that using this kind of approach would have led them to reuse their codes across development cases. The only developer suggestion was to develop a help file. The commentary file briefly presented the description of each of the components in the *UI and skeleton codebase*, but did not include examples. A help file with a thorough description and examples with respect to each of the components was suggested as really helpful for developers. One concern raised by S1.D1 and S1.D2 is that they faced time delays in data processing due to the heavy size of their datasets. They suggested using some of the technical measures that can help to reduce such delays like parallel computing. Moreover, they also pointed out that decoupling between UI and back-end is required so that the user can continue using the UI and does not need to wait until the back-end code gets executed, which would lead to delays in the current form of the *PAMEL Manager*.

[REQ5]: The developer scores and quotes suggest that the *Output Formatter* achieves its functionality. The reference implementation of CIMLA in S1 has one working developed function to export the fused dataset into *XLSX* format. Although this component was not required in S2 and S3, both of the developers confirmed its functionality and usefulness in the development of an MMLA solution. S1:D2 developed this component for the reference implementation in S1 and suggested that *Formatter's* lines of code with minor changes could be reused in the *Output Formatter*. He also highlighted that one of the advantages of using CIMLA is that a developer can make the modifications across components but that does not affect the workings of other components.

4.4 Results of the assessment of non-functional requirements

We extracted quotes that included terms like 'module', 'reuse', and 'modify' from the interview sections where developers were giving their expert opinion about the functional requirements (see Table 5 of Appendix 6). The evidence from this re-analysis suggests that CIMLA supports modularity and modifiability to a certain extent. The quotes from the interview sections where developers assessed concretely the modularity and modifiability aspects of CIMLA are presented in Table 6 of Appendix 6.

[REQ6]: The quotes clearly suggest that CIMLA supports modularity in data processing by breaking down the involved complexity into modules. Developers considered that the design and implementation of each of the components of CIMLA is dealing with a specific problem and they are independent of each other. S2.D1 and S3.D1 mentioned that they did not use a CIMLA-like approach in their original development approaches for writing, structuring, and organizing the development code. They just took the problem and developed the code in a single module to meet the requirements. After experiencing the CIMLA approach, they admitted that they should have thought of and used a similar approach. It might have cost extra time on planning and thinking but overall it could have saved time because once they had a modular approach in front of them, then developing the components based on it would be much easier and faster. They also emphasized that following a modular approach like CIMLA leads to a better structure and organization of the code that can be further reused as well.

[REQ7]: CIMLA was designed and instantiated by keeping modifiability at the center so that developers do not need to implement the different components of an MMLA solution from scratch. The assessment of the modifiability feature from all the developers involved in the three scenarios confirms this. As S1 developers did not have any prior experience in developing an MMLA solution, they had not experienced this requirement before. However, once they developed the reference implementation, they emphasized its importance and positively assessed the way CIMLA supports modifiability. S2.D1 and S3.D1 had prior experience in developing an MMLA solution, therefore, they were aware of this requirement. Once they experienced the CIMLA approach by going through its different components, they immediately assessed that it supports modifiability in that its different components do not need to be developed from scratch while developing similar MMLA solutions later on.

5 Discussion

In this section, we discuss the evaluation results and their implications for the MMLA field.

Researchers have raised the need for an infrastructure that can support the development of context-aware MMLA solutions to partially process multimodal educational data collected from learning situations that are enacted in authentic MMLA settings involving cross-disciplinary stakeholders [Camacho et al., 2020, Ouhaichi et al., 2021, Shankar et al., 2018]. The results of CIMLA's fitness analysis show that it can cope with authentic MMLA scenarios, as long as CI from the design and enactment phases is available. Moreover, its reference implementation in one scenario showed its technical feasibility and highlighted that CIMLA can be used to develop context-aware MMLA solutions. This is also supported by its evaluation in two other scenarios where developers have used technical instantiations of CIMLA components.

Development of a context-aware MMLA solution is complex due to the data-intensive nature of such solutions [Schneider et al., 2018]. Moreover, developers often are not able to reuse the developed components of existing MMLA solutions [Di Mitri et al., 2019b]. In most cases, they need to develop the components of new solutions from scratch [Di Mitri et al., 2019a]. CIMLA follows a standard data processing pipeline (a DVC [Miller and Mork, 2013]) to break down the complex data processing activities into multiple modules. The evaluation results show that CIMLA supports developers through its modular structure and each of the modules can be reused with minor modifications in the development of other MMLA solutions.

The need of fetching datasets from a set of data repositories and handling a heterogeneous set of data formats is also one of the requirements sought by the MMLA developer community [Di Mitri, 2020]. Our results show that CIMLA deals with such heterogeneity successfully through its *Fetchers* and *Formatter* components. However, only two data repositories (*Google Drive* and *Local File System*) have been technically dealt with in the current reference implementation. Similarly, CIMLA supports only a limited set of file formats for datasets (*XLSX* and *CSV*).

This research work is not without limitations. First, CIMLA has only been evaluated in three authentic MMLA scenarios so far. We cannot say that CIMLA is able to cope with *all* MMLA scenarios. For example, CIMLA cannot be used in such MMLA scenarios where CI is not present. Similarly, in scenarios where a developer is not available to manually translate such CI to CIMLA's configuration file, it still cannot be used. Third, the reference implementation was developed for only one scenario. Two other scenarios had already-developed MMLA solutions available prior to introducing CIMLA and only a limited re-implementation effort was applied in those due to the high development costs of a full implementation. Finally, all the content analysis of interviews, skeleton codebase files, and provided CI from all three scenarios were done by only one researcher, which may limit the reliability of the qualitative analyses.

6 Conclusions and Future Work

MMLA presents a more holistic picture of a learning situation than mono-modal LA by processing multimodal educational data and considering CI [Blikstein, 2013]. Yet, developing context-aware MMLA solutions that can be adopted in authentic MMLA situations that involve multiple cross-disciplinary stakeholders is still very complex for software developers [Patnaik et al., 2021, Hoyos and Velásquez, 2020]. This shortcoming is due to the current (ad-hoc and opportunistic) practice in MMLA development (where developers do not follow any standard data processing pipeline [Schneider et al., 2018]) and also because of the data intensiveness involved in processing multimodal educational data [Di Mitri et al., 2018].

In this paper, we derived the requirements for an infrastructure to overcome these MMLA development challenges: CIMLA. Further, we described the design and development of CIMLA and its components, following the separation of concerns design principle. To make CIMLA context-aware, we describe how it can be configured based on CI inputs structured and organized using a contextualized data model (CDM4MMLA, see our previous work [Shankar et al., 2022]). CI inputs are stored in a JSON file through the *Configuration File* component of CIMLA. The rest of the CIMLA components were instantiated in the form of a UI and a skeleton codebase. The paper also describes the evaluation of CIMLA concerning its functional and non-functional requirements using the SAAM evaluation method in three scenarios, including a full reference implementation of the infrastructure, developed for one of the scenarios. This evaluation shows not only the technical feasibility of CIMLA, but also how it successfully achieves those requirements, overcoming the aforementioned MMLA development challenges.

Future work in this line of research includes using natural language processing (NLP) in the context of automatically translating CI into CIMLA's configuration file, assessing CIMLA in more (and more varied) authentic MMLA scenarios, and decoupling the infrastructure's UI and skeleton codebase. The use of NLP would possibly remove manual developer intervention to extract the structured rules from CI inputs, which currently limits CIMLA's scaling up to operate in multiple different contexts. Further, improvements in CIMLA's documentation would help to better communicate CIMLA to stakeholders so that the communication problems that occurred during our evaluation can be resolved.

Acknowledgements

This research has been partially funded by the European Union via the European Regional Development Fund and in the context of CEITER (Horizon 2020 Research and Innovation Programme, grant agreement no. 669074). Moreover, it was also partially funded by the Estonian Research Council (Personal Research Grant no. PRG1634).

References

[Aldrich et al., 2002] Aldrich, J., Chambers, C., and Notkin, D. (2002). Archjava: Connecting software architecture to implementation. In *Proceedings of the 24th International Conference on Software Engineering. ICSE 2002*, pages 187–197. IEEE.

[Alhadad and Thompson, 2017] Alhadad, S. S. and Thompson, K. (2017). Understanding the mediating role of teacher inquiry when connecting learning analytics with design for learning. *Interaction Design and Architecture (s)*, 33:54–74.

[Alwahaby et al., 2022] Alwahaby, H., Cukurova, M., Papamitsiou, Z., and Giannakos, M. (2022). The evidence of impact and ethical considerations of multimodal learning analytics: A systematic literature review. *The Multimodal Learning Analytics Handbook*, pages 289–325.

[Barkhordari and Niamanesh, 2017] Barkhordari, M. and Niamanesh, M. (2017). Atrak: a mapreduce-based data warehouse for big data. *The Journal of Supercomputing*, 73(10):4596–4610.

[Bauer and Wössner, 2012] Bauer, F. L. and Wössner, H. (2012). *Algorithmic language and program development*. Springer Science & Business Media.

[Bayoudh et al., 2021] Bayoudh, K., Knani, R., Hamdaoui, F., and Mtibaa, A. (2021). A survey on deep multimodal learning for computer vision: advances, trends, applications, and datasets. *The Visual Computer*, pages 1–32.

[Berg et al., 2016] Berg, A. M., Mol, S. T., Kismihók, G., and Sclater, N. (2016). The role of a reference synthetic data generator within the field of learning analytics. *Journal of Learning Analytics*, 3(1):107–128.

[Blikstein, 2013] Blikstein, P. (2013). Multimodal learning analytics. In *Proceedings of the third international conference on learning analytics and knowledge*, pages 102–106.

[Camacho et al., 2020] Camacho, V. L., de la Guía, E., Olivares, T., Flores, M. J., and Orozco-Barbosa, L. (2020). Data capture and multimodal learning analytics focused on engagement with a new wearable iot approach. *IEEE Transactions on Learning Technologies*, 13(4):704–717.

[Chan et al., 2020] Chan, M. C. E., Ochoa, X., and Clarke, D. (2020). Multimodal learning analytics in a laboratory classroom. In *Machine learning paradigms*, pages 131–156. Springer.

[Chango et al., 2022] Chango, W., Lara, J. A., Cerezo, R., and Romero, C. (2022). A review on data fusion in multimodal learning analytics and educational data mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, page e1458.

[Chejara et al., 2020] Chejara, P., Prieto, L. P., Ruiz-Calleja, A., Rodríguez-Triana, M. J., Shankar, S. K., and Kasepalu, R. (2020). Quantifying collaboration quality in face-to-face classroom settings using mmla. In Nolte, A., Alvarez, C., Hishiyama, R., Chounta, I.-A., Rodríguez-Triana, M. J., and Inoue, T., editors, *Collaboration Technologies and Social Computing*, pages 159–166, Cham. Springer International Publishing.

[Chejara et al., 2021] Chejara, P., Prieto, L. P., Ruiz-Calleja, A., Rodríguez-Triana, M. J., Shankar, S. K., and Kasepalu, R. (2021). Cotrack2: A tool to track collaboration across physical and digital spaces with real time activity visualization. In *11th International Conference on Learning Analytics & Knowledge LAK21*, Companion Proceedings of the 11th International Conference on Learning Analytics & Knowledge LAK21, New York NY United States. Association for Computing Machinery.

[Ciordas-Hertel et al., 2019] Ciordas-Hertel, G.-P., Schneider, J., Ternier, S., and Drachsler, H. (2019). Adopting trust in learning analytics infrastructure: A structured literature review. *JUCS-Journal of Universal Computer Science*, 25:1668.

[de Silva Joyce and Feez, 2018] de Silva Joyce, H. and Feez, S. (2018). Multimodality across classrooms.

[Di Mitri, 2020] Di Mitri, D. (2020). *The Multimodal Tutor: Adaptive Feedback from Multimodal Experiences*. PhD thesis, Dutch Research School for Information and Knowledge Systems.

[Di Mitri et al., 2017] Di Mitri, D., Scheffel, M., Drachsler, H., Börner, D., Ternier, S., and Specht, M. (2017). Learning pulse: A machine learning approach for predicting performance in self-regulated learning using multimodal data. In *Proceedings of the seventh international learning analytics & knowledge conference*, pages 188–197.

[Di Mitri et al., 2019a] Di Mitri, D., Schneider, J., Klemke, R., Specht, M., and Drachsler, H. (2019a). Read between the lines: An annotation tool for multimodal data for learning. In *Proceedings of the 9th international conference on learning analytics & knowledge*, pages 51–60.

[Di Mitri et al., 2018] Di Mitri, D., Schneider, J., Specht, M., and Drachsler, H. (2018). The big five: Addressing recurrent multimodal learning data challenges. In *CrossMMLA@ LAK*.

[Di Mitri et al., 2019b] Di Mitri, D., Schneider, J., Specht, M., and Drachsler, H. (2019b). The multimodal learning analytics pipeline. In *4th International Conference on AI+ Adaptive Education*, pages 1–2.

[Domínguez and Chiluiza, 2016] Domínguez, F. and Chiluiza, K. (2016). Towards a distributed framework to analyze multimodal data. In *Proc. of Workshop Cross-LAK–held at LAK '16*, pages 52–57.

[Domínguez et al., 2021] Domínguez, F., Ochoa, X., Zambrano, D., Camacho, K., and Castells, J. (2021). Scaling and adopting a multimodal learning analytics application in an institution-wide setting. *IEEE Transactions on Learning Technologies*, 14(3):400–414.

[Eradze, 2020] Eradze, M. (2020). *Learning Interactions Across Spaces: a Framework for Contextualised Multimodal Observations*. Tallinna Ülikool.

[Eradze et al., 2020] Eradze, M., Rodríguez-Triana, M. J., and Laanpere, M. (2020). Context-aware multimodal learning analytics taxonomy. In *CrossMMLA@ LAK*, pages 1–6.

[Fiaidhi, 2014] Fiaidhi, J. (2014). The next step for learning analytics. *It Professional*, 16(5):4–8.

[Fuller et al., 2018] Fuller, K. A., Karunaratne, N. S., Naidu, S., Exintaris, B., Short, J. L., Wolcott, M. D., Singleton, S., and White, P. J. (2018). Development of a self-report instrument for measuring in-class student engagement reveals that pretending to engage is a significant unrecognized problem. *PloS one*, 13(10):e0205828.

[González-Vélez and Leyton, 2010] González-Vélez, H. and Leyton, M. (2010). A survey of algorithmic skeleton frameworks: high-level structured parallel programming enablers. *Software: Practice and Experience*, 40(12):1135–1160.

[Harrer, 2013] Harrer, A. (2013). Analytics of collaborative planning in metafora: Architecture, data, and analytic methods. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, pages 255–259.

[Hassan et al., 2021] Hassan, J., Leong, J., and Schneider, B. (2021). Multimodal data collection made easy: The ez-mmla toolkit: A data collection website that provides educators and researchers with easy access to multimodal data streams. In *LAK21: 11th International Learning Analytics and Knowledge Conference*, pages 579–585.

[Hoyos and Velásquez, 2020] Hoyos, A. A. C. and Velásquez, J. D. (2020). Teaching analytics: current challenges and future development. *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, 15(1):1–9.

[Huertas Celdrán et al., 2020] Huertas Celdrán, A., Ruipérez-Valiente, J. A., García Clemente, F. J., Rodríguez-Triana, M. J., Shankar, S. K., and Martínez Pérez, G. (2020). A scalable architecture for the dynamic deployment of multimodal learning analytics applications in smart classrooms. *Sensors*, 20(10):2923.

[Hürsch and Lopes, 1995] Hürsch, W. L. and Lopes, C. V. (1995). Separation of concerns. Technical report, College of Computer Science, Northeastern University Boston.

[Kazman et al., 1996] Kazman, R., Abowd, G., Bass, L., and Clements, P. (1996). Scenario-based analysis of software architecture. *IEEE software*, 13(6):47–55.

[Kazman et al., 1994] Kazman, R., Bass, L., Abowd, G., and Webb, M. (1994). Saam: A method for analyzing the properties of software architectures. In *Proceedings of 16th International Conference on Software Engineering*, pages 81–90. IEEE.

[Kazman et al., 2003] Kazman, R., Nord, R. L., and Klein, M. (2003). A life-cycle view of architecture analysis and design methods. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

[Kim et al., 2018] Kim, Y., Soyata, T., and Behnagh, R. F. (2018). Towards emotionally aware ai smart classroom: Current issues and directions for engineering and education. *IEEE Access*, 6:5308–5331.

[Martens and Vogten, 2005] Martens, H. and Vogten, H. (2005). A reference implementation of a learning design engine. In *Learning Design*, pages 91–108. Springer.

[Miller and Mork, 2013] Miller, H. G. and Mork, P. (2013). From data to decisions: a value chain for big data. *It Professional*, 15(1):57–59.

[Mu et al., 2020] Mu, S., Cui, M., and Huang, X. (2020). Multimodal data fusion in learning analytics: A systematic review. *Sensors*, 20(23):6856.

[Munoz et al., 2018] Munoz, R., Villarroel, R., Barcelos, T. S., Souza, A., Merino, E., Guiñez, R., and Silva, L. A. (2018). Development of a software that supports multimodal learning analytics: A case study on oral presentations. *Journal of Universal Computer Science*, 24(2):149–170.

[Muñoz-Cristóbal et al., 2018] Muñoz-Cristóbal, J. A., Rodríguez-Triana, M. J., Gallego-Lema, V., Arribas-Cubero, H. F., Asensio-Pérez, J. I., and Martínez-Monés, A. (2018). Monitoring for awareness and reflection in ubiquitous learning environments. *International Journal of Human–Computer Interaction*, 34(2):146–165.

[Muslah and Ghoul, 2019] Muslah, E. and Ghoul, S. (2019). Requirements variability specification for data intensive software. *arXiv preprint arXiv:1904.12314*.

[Ochoa et al., 2017] Ochoa, X., Lang, A. C., and Siemens, G. (2017). Multimodal learning analytics. *The handbook of learning analytics*, 1:129–141.

[Ochoa et al., 2022] Ochoa, X., Lang, C., Siemens, G., Wise, A., Gasevic, D., and Merceron, A. (2022). Multimodal learning analytics-rationale, process, examples, and direction. In *The Handbook of Learning Analytics*, pages 54–65. Soc. Learn. Analytics Res.

[Ouhaichi et al., 2021] Ouhaichi, H., Spikol, D., and Vogel, B. (2021). Mbox: Designing a flexible iot multimodal learning analytics system. In *2021 International Conference on Advanced Learning Technologies (ICALT)*, pages 122–126. IEEE.

[Patnaik et al., 2021] Patnaik, R., Raju, K. S., and Sivakrishna, K. (2021). Internet of things-based security model and solutions for educational systems. In *Multimedia Technologies in the Internet of Things Environment*, pages 171–205. Springer.

[Praharaj et al., 2022] Praharaj, S., Scheffel, M., Schmitz, M., Specht, M., and Drachsler, H. (2022). Towards collaborative convergence: Quantifying collaboration quality with automated co-located collaboration analytics. In *LAK22: 12th International Learning Analytics and Knowledge Conference*, LAK22, page 358–369, New York, NY, USA. Association for Computing Machinery.

[Prieto et al., 2019] Prieto, L. P., Rodríguez-Triana, M. J., Martínez-Maldonado, R., Dimitriadis, Y., and Gašević, D. (2019). Orchestrating learning analytics (orla): Supporting inter-stakeholder communication about adoption of learning analytics at the classroom level. *Australasian Journal of Educational Technology*, 35(4).

[Ruffaldi et al., 2016] Ruffaldi, E., Dabisias, G., Landolfi, L., and Spikol, D. (2016). Data collection and processing for a multimodal learning analytic system. In *2016 sai computing conference (sai)*, pages 858–863. IEEE.

[Schneider and Blikstein, 2015] Schneider, B. and Blikstein, P. (2015). Unraveling students' interaction around a tangible interface using multimodal learning analytics. *Journal of Educational Data Mining*, 7(3):89–116.

[Schneider et al., 2018] Schneider, J., Di Mitri, D., Limbu, B., and Drachsler, H. (2018). Multimodal learning hub: A tool for capturing customizable multimodal learning experiences. In *European Conference on Technology Enhanced Learning*, pages 45–58. Springer.

[Schuller, 2018] Schuller, B. (2018). Multimodal user state and trait recognition: An overview. *The Handbook of Multimodal-Multisensor Interfaces: Signal Processing, Architectures, and Detection of Emotion and Cognition-Volume 2*, pages 129–165.

[Segal et al., 2017] Segal, A., Hindi, S., Prusak, N., Swidan, O., Livni, A., Palatnic, A., Schwarz, B., and Gal, Y. K. (2017). Keeping the teacher in the loop: Technologies for monitoring group learning in real-time. In *International Conference on Artificial Intelligence in Education*, pages 64–76. Springer.

[Serrano Iglesias et al., 2021] Serrano Iglesias, S., Spikol, D., Bote Lorenzo, M. L., Ouhaichi, H., Gómez Sánchez, E., and Vogel, B. (2021). Adaptable smart learning environments supported by multimodal learning analytics. In *European Conference on Technology Enhanced Learning, EC-TEL 2021*.

[Shankar et al., 2018] Shankar, S. K., Prieto, L. P., Rodríguez-Triana, M. J., and Ruiz-Calleja, A. (2018). A review of multimodal learning analytics architectures. In *2018 ieee 18th international conference on advanced learning technologies (icalt)*, pages 212–214. IEEE.

[Shankar et al., 2022] Shankar, S. K., Rodríguez-Triana, M. J., Prieto, L. P., Ruiz-Calleja, A., and Chejara, P. (2022). Cdm4mmla: Contextualized data model for multimodal learning analytics. In *The Multimodal Learning Analytics Handbook*, pages 205–229. Springer.

[Shankar et al., 2020] Shankar, S. K., Rodríguez-Triana, M. J., Ruiz-Calleja, A., Prieto, L. P., Chejara, P., and Martínez-Monés, A. (2020). Multimodal data value chain (m-dvc): A conceptual tool to support the development of multimodal learning analytics solutions. *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, 15(2):113–122.

[Shankar et al., 2019] Shankar, S. K., Ruiz-Calleja, A., Prieto, L. P., Rodríguez-Triana, M. J., and Chejara, P. (2019). An architecture and data model to process multimodal evidence of learning. In *International Conference on Web-Based Learning*, pages 72–83. Springer.

[Siemens and Long, 2011] Siemens, G. and Long, P. (2011). Penetrating the fog: Analytics in learning and education. *EDUCAUSE review*, 46(5):30.

[Slupczynski and Klamma, 2021] Slupczynski, M. and Klamma, R. (2021). Milki-psy cloud: Facilitating multimodal learning analytics by explainable ai and blockchain. In *Proceedings of the First International Workshop on Multimodal Immersive Learning Systems (MILeS 2021)*, pages 22–28.

[Spikol et al., 2018] Spikol, D., Ruffaldi, E., Dabisias, G., and Cukurova, M. (2018). Supervised machine learning in multimodal learning analytics for estimating success in project-based learning. *Journal of Computer Assisted Learning*, 34(4):366–377.

[Tamura et al., 2019] Tamura, K., Lu, M., Konomi, S., Hatano, K., Inaba, M., Oi, M., Okamoto, T., Okubo, F., Shimada, A., Wang, J., et al. (2019). Integrating multimodal learning analytics and inclusive learning support systems for people of all ages. In *International conference on human-computer interaction*, pages 469–481. Springer.

[Wagner et al., 2011] Wagner, J., Lingenfelser, F., and Andre, E. (2011). The social signal interpretation framework (ssi) for real time signal processing and recognitions. *Proceedings of INTERSPEECH, Florence, Italy*.

[Woods and Rozanski, 2005] Woods, E. and Rozanski, N. (2005). Using architectural perspectives. In *5th Working IEEE/IFIP Conference on Software Architecture (WICSA'05)*, pages 25–35. IEEE.

[Wright, 2012] Wright, D. R. (2012). *Principles, patterns, and process: A framework for learning to make software design decisions*. North Carolina State University.

[Yan et al., 2022] Yan, L., Zhao, L., Gasevic, D., and Martinez-Maldonado, R. (2022). Scalability, sustainability, and ethicality of multimodal learning analytics. In *LAK22: 12th International Learning Analytics and Knowledge Conference*, pages 13–23.

Proposal	No. of tools	Data processing activities					Contextual information		Involved stakeholders		
		Preparation	Organization	Fusion	Analysis	Visualization	Design	Enactment	Teacher	Researcher	Developer
[Schneider et al., 2018]	3	✓			✓		✓			✓	✓
[Munoz et al., 2018]	3	✓	✓		✓	✓				✓	✓
[Tamura et al., 2019]	3			✓	✓	✓	✓			✓	
[Ciordas-Hertel et al., 2019]	2			✓	✓					✓	✓
[Huertas Celdrán et al., 2020]	3	✓	✓		✓	✓				✓	✓
[Camacho et al., 2020]	3			✓	✓		✓	✓		✓	
[Domínguez et al., 2021]	3				✓	✓	✓			✓	
[Serrano Iglesias et al., 2021]	4	✓	✓	✓	✓	✓	✓			✓	
[Slupczynski and Klamma, 2021]	2		✓		✓		✓	✓		✓	
[Di Mitri et al., 2017]	3	✓	✓	✓	✓	✓	✓			✓	✓
[Muñoz-Cristóbal et al., 2018]	3	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
[Fiaidhi, 2014]	11	✓			✓	✓				✓	
[Domínguez and Chiluiza, 2016]	2	✓		✓	✓		✓			✓	
[Ruffaldi et al., 2016]	3		✓	✓	✓	✓			✓	✓	✓
[Harrer, 2013]	2	✓	✓		✓	✓				✓	
[Berg et al., 2016]	4			✓	✓	✓			✓	✓	
[Wagner et al., 2011]	3	✓	✓							✓	✓
[Segal et al., 2017]	2				✓	✓				✓	

Table 3: Summary of the 18 reviewed MMLA infrastructures

Component	S1	S2	S3
Fetchers	L1	L4	L1
Formatter	L1	L4	L4
Output Formatter	L1	L4	L4
Synchronizer	L4	L4	L4
Time homogenizer	L4	L4	L4
Denoiser	L1	L4	L2
Aggregator	L1	L2	L1
Variable Excluder	L1	L4	L1
Case Excluder	L1	L1	L4
Transformer	L1	L2	L2
Mapper	L1	L1	L1
Joiner	L1	L2	L2
Integrator	L1	L3	L4
Input of required rules from the configuration file for defining the behavior of ...			
Fetcher	L1	L1	L1
Formatter	L1	L1	L1
Preparation Core	L1	L2	L2
Organization Core	L1	L2	L2
Fusion Core	L1	L2	L2
Output Formatter	L1	L1	L1

Table 4: A summary of fitness analysis of CIMLA in three MMLA scenarios [L1- Direct, L2- Direct but with an intervention, L3- Indirect, L4- Not Applicable]

[Requirement : Scenario]	[Score: 1 - Very Unsuccessful, 5 - Very Successful] / [Quote(s)]
[REQ1 : S1]	[4] [S1.D1 - ‘It fetched datasets from Google Drive and Local File System’] [S1.D1 - ’Fetcher is very useful in getting the datasets from different storage’] [S1.D2 - ‘Fetching data file from Google Drive was a little bit difficult but fetching data from the local system was easier’]
[REQ1 : S2]	[5] [S2.D1 - ‘Though I don’t need this Fetcher function but there could be another scenario where structured data need to be downloaded from the database then it can be an option’] [S2.D1 - ‘Both the defined functions in the skeleton codebase would do it work to the fullest’]
[REQ1 : S3]	[5]

	[S3.D1 - 'It should work in my case where I need to access the recorded audio files either from the local system or through the server of the university. I can see that both the defined functions in the skeleton codebase here can be directly used for this']
[REQ2 : S1]	[5] [S1.D1 - 'As we are comfortable in processing CSV files in Python, this Formatter helped us to format the excel sheets to CSV files'] [S1.D2 - 'Once both the datasets were in CSV format then processing them was easier']
[REQ2 : S2]	[5] [S2.D1 - 'In my expert opinion, I have seen multiple file formats used in MMLA literature like JSON. As I can see the function definition and from that my guess is that it should work to format the data files into CSV'] [S2.D1 - 'Formatter in not required in my case, but, I think the Formatter would also achieve its functionality'] [S2.D1 - 'Although I see its usefulness, I wonder if we need to unify the datasets in one file format. There are multiple libraries in Python that supports the processing of different file formats']
[REQ2 : S3]	[5] [S3.D1 - 'In my case the audio files were in wav format. I also formatted them in CSV and I see here that Formatter also does the same.'] [S3.D1 - 'It is very useful because once all the data files are in CSV then selecting the particular data science language and library beforehand of development is easy. And, once the selection is well thought out then the risk of changing the libraries due to limited support would be wiped off']
[REQ3 : S1]	[5] [S1.D1 - 'The concept of configuration file to provide the arguments is one of the best part in PAMEL because it gives me the boundaries where I need to place my codes in each of the components'] [S1.D2 - 'Context-aware data processing seemed quite puzzling to me but when I went through the configuration file, the development was straight forward'] [S1.D2 - 'However, extracting the rules from the teacher's input and saving them to the configuration file was a quite boring task']
[REQ3 : S2]	[4] [S2.D1 - 'I have provided the contextual information in the sheet but I do not know how this unstructured text is going to be used directly as arguments to the different functions. The document explains that the developer would do the intervention in extracting the rules from the provided text but I do not know how would developers feel in doing that'] [S2.D1 - 'In my opinion, the NLP should be used here as an option so that the developer intervention can be removed from the whole process'] [S2.D1 - 'I do not know if the current configuration file covers all contextual information for any kind of learning scenario but in its current form, it covers almost all in my case']

	[S2.D1 - 'In my case, as I was involved as a researcher in the learning scenario I had the contextual information in my head and I developed the solution without asking from someone else. But, I understand the other case where developer and researcher are not the same person, and, in that case, it should work very well']
[REQ3 : S3]	[4] [S3.D1 - 'I wish I could use PAMEL next time so that I do not need to spend much time in development and working on contextual information'] [S3.D1 - 'Using some kind of AI would be helpful in removing the intervention of developer for extracting the rules']
[REQ4 : S1]	[5] [S1.D1 - 'Initially, I thought that why PAMEL involves so many core components and their sub components to process the data, when the processing can be done in an easier way. But when I really developed the solution, I must say that I was completely wrong'] [S1.D1 - 'If the PAMEL would not have provided different components following the standard data processing pipeline to process the data, I could not have develop such a complex system on my own'] [S1.D2 - 'Although many data processing steps were not used in our scenario but when I thought about other kind of noisy datasets, I realized the importance of each of them'] [S1.D2 - 'The sequence of data processing steps gave me the overall understanding of how should I go from the datasets to the expected fused dataset']
[REQ4 : S2]	[5] [S2.D1 - 'In my case, very few functions were used but the structure that PAMEL presents eases the task of data processing'] [S2.D1 - 'I was confused in selecting the functions for a specific task. The explanation provided in the document file was not enough for me to decide. A help file would be helpful in defining each of the components'] [S2.D1 - 'If we have the pre-defined structure like skeleton codebase here then it will help programmers to better and faster develop the code']
[REQ4 : S3]	[5] [S3.D1 - 'In my opinion, this structure would really help developer to better understand the problem and save the development time'] [S3.D1 - 'I used only few components but I can think of several cases where other components can be used']
[REQ5 : S1]	[5] [S1.D1 - 'This is just like Formatter but now I see its applicability very important especially when third party tools are going to be used for the analysis and visualization purpose'] [S1.D2 - 'It was easy to develop because I used the same function of Formatter just in reverse']
[REQ5 : S2]	[5]

	[S2.D1 - 'It was not required in my case but in my opinion, it should work in the cases where the fused dataset needs to be exported in a specific format']
[REQ5 : S3]	[5] [S3.D1 - 'It is a useful function to export the fused file in a file format that is supported by a set of analysis and visualization tools']

Table 5: Development scores and quotes reported in assessing functional requirements

[Requirement : Scenario]	**[Quote(s)]**
[REQ6 : S1]	[S1.D1 - 'PAMEL as an architecture is much more modular that contains different modules like preparation, organization, and fusion to reduce the complexity'] [S1:D2 - 'It follows the modular approach and gives more clarity'] [S1:D2 - 'The use of data processing pipeline itself makes PAMEL modular and the codes for different functions and classes helped me in writing better and faster code for such a complex problem']
[REQ6 : S2]	[S2.D1 - 'When I look back to my development approach, I did not devote much time in giving a thought on defining the development approach and structuring the overall problem into smaller modules. But, now when I look back, I think I should have spent some time there'] [S2.D1 - 'PAMEL provides this better organization structure by breaking the complexity further and yes, it supports modularity'] [S2.D1 - 'And having something already prepared will save the MMLA researcher and developer from investing time in structuring code which I normally don't do']
[REQ6 : S3]	[S3.D1 - 'PAMEL does not cover the complete ecosystem of MMLA but the part that it covers, covers does it well'] [S3.D1 - 'When I extended the skeleton codebase, I saw different parts that clearly represent different themes of MMLA. This clearly looks distinct and modular'] [S3.D1 - 'The structure is very modular but if there would be a more specific function definitions in the codebase than the abstract ones then it would be much easier like just filling in the blanks']
[REQ7 : S1]	[S1.D1 - 'If we using PAMEL or alike structure then the issue of writing the code every time from scratch would be easily solved because most of the abstract components in the solution are going to be the same'] [S1.D1 - 'Take an example of the Fetcher class, it has two functions to fetch datasets from Google Drive and Local File system in this case. But, if someone also needs to fetch from Google Drive then he can use the same function. He just needs to change the public URL and credentials'] [S1.D2 - 'Every component in the UI, classes, and functions are defined separately that can be modified individually without risking the overall working of the system. And, they can be directly reused in other case with adding or subtracting something as per the requirement']

[REQ7 : S2]	[S2.D1 - 'As I have gone through the skeleton codebase and considering the current level of abstraction, I see the modifiability aspect'] [S2.D1 - 'I cannot say how much modifiability it will bring to other MMLA cases but in my case, it works'] [S2.D1 - 'If I could have used it in my first case then I could have reused most of the code in my second case']
[REQ7 : S3]	[S3.D1 - 'Basically, you don't need to write from scratch and suppose you are using Fetcher you already have three defined different functions that can be reused'] [S3.D1 - 'Once you have this structure and everything then it's much easier for anyone be the novice developer or who have experience can use it in any context, reuse, modify, delete, everything can be done']

Table 6: Quotes reported in assessing non-functional requirements