A Self-Adaptive Multi-Agent System Approach for Collaborative Mobile Learning

Didac Gil de la Iglesia, Juan Felipe Calderón, Danny Weyns, Marcelo Milrad, Miguel Nussbaum

Abstract—Mobile technologies have emerged as facilitators in the learning process, extending traditional classroom activities. However, engineering mobile learning applications for outdoor usage poses severe challenges. The requirements of these applications are challenging, as many different aspects need to be catered, such as resource access and sharing, communication between peers, group management, activity flow, etc. Robustness is particularly important for learning scenarios to guarantee undisturbed and smooth user experiences, pushing the technological aspects in the background. Despite significant research in the field of mobile learning, very few efforts have focused on collaborative mobile learning requirements from a software engineering perspective. This paper focuses on aspects of the software architecture, aiming to address the challenges related to resource sharing in collaborative mobile learning activities. This includes elements such as autonomy for personal interactive learning, richness for large group collaborative learning (indoor and outdoor), as well as robustness of the learning system. Additionally, we present self-adaptation as a solution to mitigate risks of resource unavailability and organization failures that arise from environment and system dynamism. Our evaluation provides indications regarding the system correctness with respect to resource sharing and collaboration concerns, and offers qualitative evidence of self-adaptation benefits for collaborative mobile learning applications.

Index Terms—Mobile Learning, Software Architecture, Multi-Agent Systems, Self-Adaptation

1 Introduction

Mobile technologies have emerged as facilitators in the learning process, offering new ways to access and use learning materials and defining the mobile learning paradigm [1]. Kukulska-Hulme [2] states that learners should be able to engage in educational activities without being bound to a tightly-delimited physical location. In this context, mobile technologies have the capability to provide resources that meet a subset of the new learning needs, such as environment and contextual information [3]. Mobile technologies can enrich learning activities and satisfy requirements for individual and group activities. More specifically, these technologies might foster user interactions based on access to rich content across locations and at any time using portable equipments such as wireless laptops, personal digital assistants (PDAs) and smart phones [4].

In addition, mobile technologies can assist the development of collaborative learning activities. Ogata [5] mentions the advantages related to the use of mobile technologies to organize and mediate social interactions, regardless of time and location in which the learning activities take place. Zurita and Nussbaum's work [6] extends the list of potential benefits mentioning that mobile technologies can be used to facilitate information sharing, moderate the tasks to be completed, facilitate the management of rules and participant roles, and mediate the acquisition of new knowledge. Moreover, nowadays learning activities do not only take place in traditional learning environments (such as classrooms, lecture halls, etc.) but also in less-traditional environments (such as outdoor settings, learning in public places, museums, exhibits, etc.). Therefore, there is a need to better integrate learning goals across these variety of contexts [7]. Mobile technologies play a crucial role in supporting these new developments by enabling users' active participation in these new learning landscapes without limitations imposed by space and time [8].

In a recent study, Lucke & Resing [9] performed an extensive survey in the field of mobile learning, in order to identify current trends and challenges in the field. The authors mention *Software* and *Hardware* in their list of areas for future research in mobile learning. In the work described in this paper, we focus on a number of central software architecture-related aspects for the design of mobile learning applications. Engineering mobile technology-enhanced learning applications for outdoor usage poses severe challenges. Initial efforts to focus on software architecture-related aspects for mobile learning are described in [10], [11]. Concerns such as platforms for scalability, software engineering methods for interoperability and strategies for deploying robust applications were identified as challenges that the field needs to address. The collaborative nature of many Technology-Enhanced Learning (TEL) activities seems to require a technological platform that supports resource sharing (resources are abstractly defined as controllable parts of a system, such as software and hardware components, devices, etc.). This has recently prompted the usage of distributed solutions on mobile devices. The first aim of this paper is to provide a software architecture that supports mobile learning activities where resources and data

must be shared during execution.

Technology in education must avoid creating new barriers to teachers and students in their process to achieve learning goals [12]. One of these barriers can be undesired system failures and diminishing of quality in the system services, as experienced in [13], [14], [15], [16]. Robustness of the applications is particularly important to guarantee undisturbed and smooth user experiences. This is vital for learning scenarios to bring the context and tasks in focus and push the technological aspects in the background. Mobile software applications often take into consideration ideal cases of execution, based on a set of correctness assumptions. However, this view is not aligned with real environment deployment, where scenarios found at runtime do not comply with the assumptions taken during design and implementation [13], [17]. Outside the classroom settings, the environment induces several uncertainties itself. Classical examples of uncertainty in mobile applications are battery and network conditions, which may be exhausted or interrupted and affect completing the collaborative tasks. Uncertainties can also be related to certain aspects of the system, such as the availability of resources needed to cover activity requirements. In certain scenarios, an activity may require resources not available in a mobile device. This could occur when the mobile device lacks a specific resource (i.e. a barometric pressure), the available resources are not sufficient (i.e. powerful processor and large memory for demanding tasks such as specialized image and voice recognition with extensive databases), and because a requirement demands the combination of multiple devices. Obviously, the list of uncertainties can be substantially large. Therefore, the second aim of this paper is to design a software architecture for collaborative mobile learning activities that can offer robustness to achieve the activity goals, despite certain uncertainties, and reduce human intervention to address undesired system behaviors. Self-adaptation[1] is a well established approach that aims to provide certain desired quality of software applications [18].

The following research question (RQ) guides our work on providing an answer to the aims presented above:

RQ: *How should we design a robust software system to support collaborative mobile learning activities that require shared resource capabilities?*

To that end, we review current mobile learning applications (Section 2) to identify the requirements that have been determined in the literature. We take into consideration requirements that are focused both on learning and technical needs. Given that the applications found in the literature focus on just a few of the identified attributes, an architecture that considers all of these demands fulfills a large scope of learning scenarios. Such an architecture increases the possibilities of the learning activities, such as autonomy for personal interactive learning, richness for large group collaborative learning (both indoor and outdoor), as well as robustness for the reliability of the learning system. In order to validate our proposed architecture, we apply it to two mobile learning applications that meet all of the above requirements.

The rest of the paper is structured as follows: in Section 2 we present related efforts in this research area to identify current challenges in the state-of-the-art. Section 3 presents two learning scenarios with requirements that serve as a basis for design, implementation and deployment of a solution. In Section 4, we propose an architecture solution that addresses the identified requirements, which is evaluated in Section 5. Section 6 extends the proposed solution to address environmental uncertainties by applying self-adaptation techniques. In Section 7 the conclusions of this work are presented and compared with related efforts in the field, together with a description of future efforts.

2 Related efforts

In the field of collaborative TEL, the technological platform supporting the activity must support individual and collaborative learning, and promote interactions among peers [6]. One of the main objectives of a collaborative learning activity is to provide individual support in the activity and to encourage collaboration, in order to increase the success rate in the learning process. Current mobile technologies have a set of capabilities that can offer the means to enhance collaborative learning activities [4]. Some of these capabilities include the support of multimedia objects, positioning sensors, wireless connectivity, and other resources and sensors. As presented in the previous section, a current limitation in collaborative mobile learning activities is providing a software solution to offer resource sharing capabilities. In the software engineering field, this has been addressed by service composition approaches that combine multiple nodes. In our previous work, we address this challenge by the use of Mobile Virtual Devices (MVD) [20], where multiple mobile devices form organizations that provide mechanisms for offering and consuming local resources. One approach adopted in the mobile computing community is to consider mobile devices as thin clients that consume services located in the cloud (a virtual environment of distributed resources on large-capacity servers), also known as cloud-computing services [21]. This approach extends the capabilities of mobile devices, which consume local resources as well as services offered in the cloud [22], [23]. One example is presented in [22], that provides a middleware layer to connect cloud-based services with mobile clients through SMS services; and in [24], where the SMS channel is substituted with GPRS/WIFI connections. These solutions allow extendibility of the mobile device capabilities. However, they are limited to interactions between mobile devices and cloud services, which restrict desired

1. To avoid confusion with the term "adaptive educational systems" as commonly used in the field of TEL, the term "self-adaptive systems" should be understood as a technical characteristic of a learning application. In particular, self-adaptation refers to the capability of a learning system to modify its architectural structure or behavior in response to changes in the environment in which they are placed, to achieve or maintain specific software quality goals, such as robustness to partial failures, maintaining a particular level of performance, etc. [19].

peer-to-peer interactions to allow low-latency sharing of resources located in mobile devices.

To enable and promote interaction among users, some efforts have explored the use of peer-to-peer approaches with mobile technologies. Coco [25] and Mico [26] are suites of collaboration applications that use peer-to-peer technology to enable spontaneous collaborations and share computing resources across a network. The aim of these platforms is to develop a peer-to-peer platform using XML-based protocols, achieving scalability, some level of flexibility and ad-hoc networking formation. Additional efforts, with document sharing approaches towards information collaboration, have applied ad-hoc network infrastructures among mobile devices [27], [28]. These solutions focus on content sharing, but ignore more complex resources sharing and coordination-related issues.

More recent solutions have employed distributed systems in their mobile learning activities through software agents to strengthen the individual and group tasks [29], [30], which are referred as multi-agent systems (MAS) [31]. The agents in a MAS are autonomous, which allow them to complete tasks individually, but also posses the necessary mechanisms to enable communication among agents, thus facilitating group activities [32]. Agent's autonomy offers some level of adaptation to fit the learning activities into environment context and users' settings [33]. Even though resources are distributed among the MAS platform, the current approaches are focused on sharing static resources (like multimedia content, documents, etc.), and not on sharing other more complex resources (e.g. hardware resources such as a camera, GPS sensors, processing capabilities, etc.). These approaches are still limited to use resources locally available in a device, which restricts the number of resources that can be used or requires the use of expensive devices providing all the required features.

The above-presented studies offer an overview of the current trend related to the design of collaborative mobile learning technologies with regard to information and resource sharing aspects. One particular approach uses multi-agent systems, to provide the functions for resource sharing within the platform and autonomy to allow individual activity assessment. However, none of the previous work completely covers the emerging technological requirements for collaborative mobile learning activities, that include group management, dynamic collaboration, local and remote resource accessibility and service composition on mobile settings. New mobile learning technological platforms should offer resource-sharing mechanisms to consume local and external resources, in order to anticipate mobile device hardware limitations and to support the set of requirements mentioned above.

The following research questions refine the general research question in order to tackle the above identified challenges:

RQ1: *Which are the most suitable characteristics that software architectures should have to offer resource sharing for collaborative mobile learning activities?*

RQ2: *Which software elements are necessary for collaborative mobile learning applications, in order to satisfy resource-sharing requirements for individual performance and collaborative interactions?*

RQ3: *How to guarantee robustness with respect to supporting collaboration through the mobile applications in dynamic environments?*

3 Running Examples & Requirements

This section introduces two scenarios that illustrate the aims mentioned above. From the scenario descriptions we derive a list of generic Use Cases and related requirements that help to describe future collaborative mobile learning applications. The list of requirements provides the drivers for the architecture solution and a refinement of the aims we address in this work.

3.1 English Numbers Sorting

Infante et al. [34] presented a scenario that uses a single computer with multiple-mouses to support collaborative learning activities. Based on this concept, a new effort of collaborative language laboratories was developed using multiple-headsets [35]. The main goal of this learning activity was enhancing the students' English pronunciation and understanding. The system assisted the activity via a speech-recognition solution, in order to evaluate participants' pronunciation and provide automatic feedback. This specific application had two main limitations. First, due to a shared physical space (display), the students could peek on their colleagues' information, which clearly diminishes the learning outcomes. Dividing the display into personal physical spaces would benefit the learning process, providing the suitable information to each participant in the activity [5]. Second, the application did not allow user's mobility.

We defined a new version, called English Number Sorting (hereafter, ENS), that addresses the mentioned limitations. The activity can be performed, in groups of three, in a shared lecture room during the classes and on distance from the students' home during homework. The activity is divided in two main phases: individual and collaborative. In Phase 1, each student is assigned a number, and he is requested to correctly pronounced in in order to proceed to Phase 2 (pronounced numbers are recorded to be used in the second phase). In Phase 2, the students must propose and agree on an incremental sequence of numbers. For this phase, the students do not visualize the numbers in the group, but they can hear their peers' recordings from Phase 1. Fig. 1 shows a group of three participants in the first phase of the ENS activity. The left-side member must still pronounce "seven" to activate the assigned number; the center participant has correctly pronounced the number "four" (*"Right!"* feedback), and the right-handed participant has incorrectly pronounced the number "six" (*"Wrong. Try again"* feedback). All the students are required to correctly pronounce their assigned number in order to proceed with the second phase.

Fig. 1. Group of three peers to sort randomly generated numbers.

This scenario exemplifies the need for a system that supports individual performance (we define this functional requirement as FR.2[2]) and collaboration (FR.5) for a learning activity. Based on the scenario description, a supporting system must offer multiple nodes (mobile phones) for the participants in the activity (FR.1), and these need to be connected through a communication infrastructure in order to cover resource-sharing requirements (FR.4). In particular, a supporting system should offer mechanisms to distribute the assigned numbers among the students' devices (in phase one), and share recorded audio files between participants in a group (in phase two).

Due to current technological limitations, mobile devices may not be capable to locally perform speech-recognition processes with the desired level of accuracy. Therefore, an external server is required to operate this process (FR.4). In order not to interfere with the activity flow and not to interrupt the students' performance in the activity, we set 3 sec. as an acceptable lapse for system feedback (we define this as non-functional requirement NFR.1). This threshold has been defined according to requirements from the specific domain [36], and from a study with experts in linguistics and second language acquisition in Chile.

3.2 The Hidden Treasure

"The Hidden Treasure" (hereafter, THT) provides a second example of a collaborative mobile learning activity with resource-sharing characteristics and participants mobility in outdoor settings. THT is based on our previous work on fostering the understanding of geometric concepts for students between the age of 10 and 12 assisted by mobile technologies [13]. During the activity, groups of participants perform distance measurements on the field in order to perform different geometric calculations (distance, perimeter, area, and volume). Mobile devices equipped with GPS are used as tools for distance measurement (offer information regarding the distance between the mobile devices).

In THT, the participants must apply triangulation techniques to find a hidden treasure. In order to complete the activity, each participant receives a mobile device with a customized application for the calculation of distances. Mobile devices can interact with an activity server that specifies the set of tasks to be carried out in the activity and collects activity responses for each group. Additionally, participants receive a treasure map on paper (see Fig. 2(a)).

2. Functional and non-functional requirements identifiers are used for reference in Section 4

Two marks in the field correspond to points A and B on the map, enabling the students to make a mapping between the physical world and the map representation. Given these two initial points, the distances between points (represented as A-G points and treasure in the map) and trigonometry concepts, the participants must determine the physical location of the treasure. Finding the treasure requires the discovery of the intermediate points, via triangulation. Triangulation tasks require the involvement of at least three participants. Thus, the participants must collaborate and coordinate their actions in order to determine the points in the map. When requested (attempt), the application must provide feedback in terms of current distances between members in the group. The feedback can then be compared with expected distances in the map, in order to determine if the attempt successfully corresponds with a desired location (Fig. 2(b)).

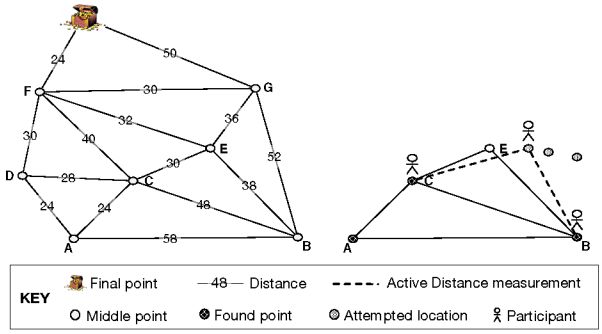

Fig. 2. (a) Treasure map. (b) Instance of a point search.

This scenario requires a system that uses mobile devices with GPS capabilities (FR.1, FR.3). With respect to resource-sharing aspects, the participants must share their current location to perform the triangulation tasks (FR.4, FR.6). Additionally, the system must offer a communication channel between participants in a group to notify about the discovery of a targeted point and assist the coordination between individual and group performance (FR.2, FR.5). In order not to disturb the activity flow, the system is required to offer distance-calculation functionality when required and to have low latency delays for the feedback (NFR.1). Based on our previous experiences [13], we defined 10 seconds as a max. waiting time to *offer* distance functionalities (a location may not be available due to low positioning accuracy) and 1 second as the accepted latency to *serve* distance calculation requests.

3.3 Requirements Overview

Fig. 3 shows a use case diagram for the two previous learning scenarios. The use cases offer key points for the design and validation of a solution for the collaborative mobile learning activities. Based on the use cases, we provide the following set of requirements that a software system should offer:

FR.1 One device per participant

- ENS: to *Sort Number, Access microphone, Access audio recording (Perform individual task)*

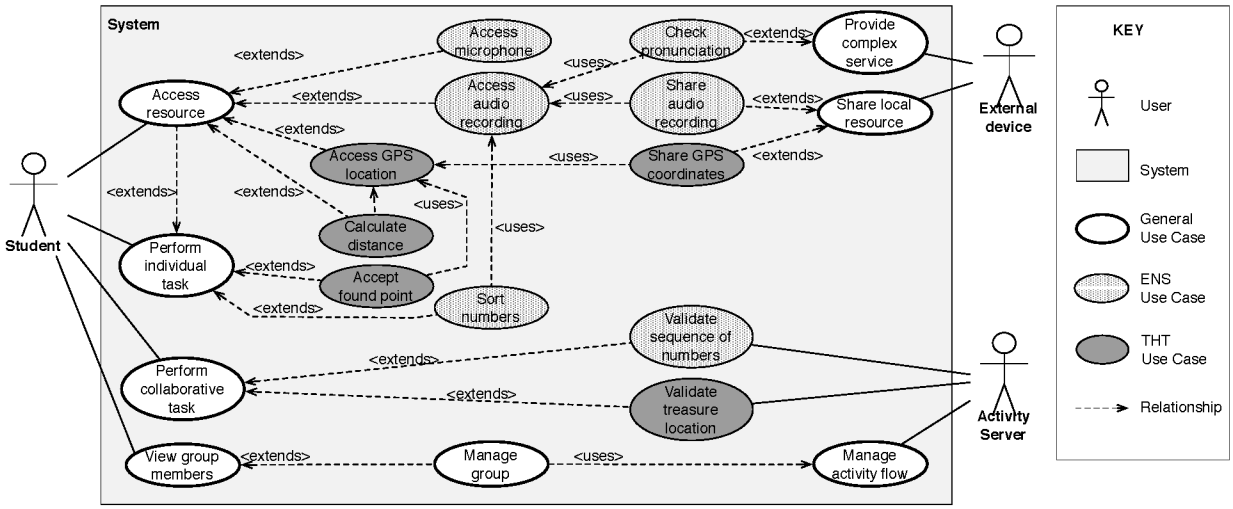

Fig. 3. Use Cases for the described learning scenarios.

- THT: to *Calculate distance, Accept found point (Perform individual task)*

FR.2 Activity flow management
- ENS & THT: to *Perform Individual Task, Perform collaborative task*

FR.3 Local resource accessibility
- ENS: to *Access microphone*
- THT: to *Access GPS location*

FR.4 Remote resource accessibility. Resource-sharing capabilities.
- ENS: to *Share audio recording, Check pronunciation, Validate sequence of numbers*
- THT: to *Share GPS coordinates, Validate treasure location*

FR.5 Group management, coordination & collaboration
- ENS: to *Sort numbers*
- THT: to *Calculate distance*

FR.6 Service composition
- ENS: to *Check pronunciation*
- THT: to *Calculate distance*

NFR.1 A defined QoS in terms of response time.
- ENS & THT: to *Perform Individual tasks, Perform collaborative task, Share local resource, Manage group*

NFR.2 A defined QoS in terms of service robustness.
- ENS & THT: to *Perform Individual tasks, Perform collaborative task, Share local resource, Manage group*

4 The Software Architecture

In this section we propose an architecture to support collaborative mobile learning activities. The architecture has been designed following the requirements identified in the use cases presented in the previous section. We answer RQ1 with a description of the software decisions that we take for the design of an architecture and their relationships with the identified requirements. Later, we specify the architecture design, elaborate on the software elements of the architecture, providing an answer to RQ2. RQ3 and NFR.2 (robustness) are studied in Section 6.

4.1 Software Decisions

The requirements of mobile learning applications are particularly challenging as many different aspects need to be catered, such as resource access, resource sharing, communication between peers, group management, activity flow, etc. According to the requirements presented in Section 3, we require a distributed architecture that provides the infrastructure for individual assignment of devices and offers one device per participant in the activity (FR.1). Distributed architectures can facilitate the distribution of resources among the participants in the activity and can offer capabilities for sharing resources within the activity.

In particular, we advocate for the use of multi-agent systems as an approach of designing a distributed system. MAS are composed of multiple devices with autonomous capabilities for decision taking [31], [32]. In a MAS, each agent holds a set of behaviors that describe the logic for selecting actions and communication protocols to interact with other autonomous agents. These autonomous capabilities are highly beneficial for the definition of individual student tasks (to determine the individual performance) and to define the flow, participation and collaboration criteria for collaborative tasks [37]. Also, multi-agent systems provide mechanisms for low latency communication that result in an increasing message transmission performance. Such a solution reduces bottleneck issues when compared with centralized approaches [38]. In our design, each mobile device becomes an agent of the MAS.

In the context of this work, agents are capable of autonomous decision-making for tasks that should be completed locally (e.g. choosing a service among various service providers, FR.3), and facilitate delegation of certain responsibilities with respect to the entire system (i.e. numbers sorting and calculation of distances, FR.4, FR.6). These features give autonomy to the nodes, in comparison to a centralized coordinator solution. A decentralized solution reduces response times in large scenarios where partial and located knowledge is sufficient, as it avoids potential

bottlenecks (NFR.1). Based on these characteristics, an architecture based on autonomous agents has good potential for scalability. With the definition of autonomous specific behaviors for the mobile devices it is possible to define the autonomous logic to use the phone microphone during the Phase 1 of the ENS scenario, to define criteria to access the GPS in THT scenario, and to provide personal feedback locally and autonomously on the mobile device.

Multi-agent systems frameworks provide mechanisms for communication via specialized protocols for data exchange between agents [31]. This supports the communication needed for a collaborative activity. For instance, this feature has been used to offer communication between peers in the activities, between the mobile devices and activity servers and for resource sharing capabilities.

Resource sharing mechanisms are required to satisfy the lack of particular resources in user nodes. MAS enable devices to access remote resources (offered as services by other agents in the MAS), when these type of resources are not available locally, using standard approaches (e.g. yellow pages services offer mechanisms for service registration and discovery). In our previous work [20], we presented a middleware for resource-sharing in organizations with mobile devices based on the notion of Mobile Virtual Device (hereafter, MVD). A MVD consists of the aggregation of multiple mobile devices to create a virtual entity that shares resources located on mobile devices within the MVD. The MVD middleware also allows the composition of existing services and resources located on the devices. Accordingly, the combination of a MAS with a MVD middleware provides the capabilities for the use of local and remote resources within the work group, which covers the necessary technological needs in order to satisfy the previous use cases. Table 1 summarizes and relates Requirements with Use Cases and Software Design Decisions.

TABLE 1
Mapping learning scenarios requirements with software decisions

Concern	Requirement	Software Design Decisions
Individual work	(FR.1) One device per participant (FR.2) Activity flow management: Perform individual task (FR.3) Local resource accessibility: Microphone, GPS location	Autonomous agents deployed on mobile devices
Collaboration (communic.)	(FR.2) Activity flow management: Perform collaborative task (FR.4) Remote resource accessibility GPS location, Audio recording (FR.6) Service Composition	Use of a distributed system through Multi-agent systems
Collaboration (organization management)	(FR.5) Group Management: View group members (FR.6) Service Composition: Provide complex service	Agents on devices organize dynamically as MVD
Resource Sharing	(FR.6) Service Composition: Provide complex service	Multi-agent system, MVD middleware
Adequate Response Time	(NFR.1) QoS (response time): Perform indiv./collab. task Share local resource Provide complex service	Communication via a Multiagent system

4.2 Architecture Design

Here, following the software design decisions presented in Section 4.1, we first introduce the suggested architecture through a layer diagram. The diagram provides a high level abstraction of the deployment of the distributed system (Fig. 4). Second, we enter in a deeper level of detail with a component diagram in order to identify the basic components in the system and understand their roles and interactions within the system (Fig. 5).

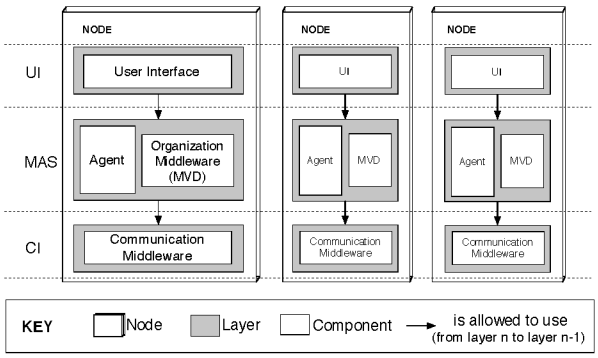

Fig. 4. Architecture described through a layer diagram.

Three main layers have been defined to structure the distributed system (see Fig. 4). The User Interface (*UI*) layer offers the points of interactions with the user, such as windows, buttons, videos and feedbacks regarding the activity. This layer is allowed to use the underlying layer, Multi-Agent System (*MAS*), which is in charge of providing domain specific functionalities that are required for the activity at hand. The MAS layer contains an *Agent* and an *Organization Middleware (MVD)*, based on the MVD concepts as described above. Agents are in charge of executing the roles that are necessary to perform tasks in the activity, such as accessing mobile device resources (both local (FR.3) and remote (FR.4)), managing the activity logic (FR.2), providing interaction between mobile devices and servers and interaction between members in a group. In order to manage group-related aspects, such as the identification of members in the group, the MAS layer includes the MVD middleware. Together, the *Agent* and *Organization middleware* are capable of providing functionalities for collaborative tasks (FR.5), such as distance calculation between mobile devices (FR.6). Finally, the Communication Infrastructure (*CI*) layer provides basic mechanisms for communication within the platform, such as channels and protocols.

Fig. 5 shows the different types of nodes involved in a distributed system. The four types are the *mobile devices*, the *activity servers*, *service proxies* for external services and *infrastructure servers*. Typically, mobile devices are the point of interaction for participants in the activity, as these provide the application interface, some local resources and mobility features that are required for the mobile learning activity. The activity server entities have the role to manage the groups and control the task flow of the participants in the activity. In particular, nodes with

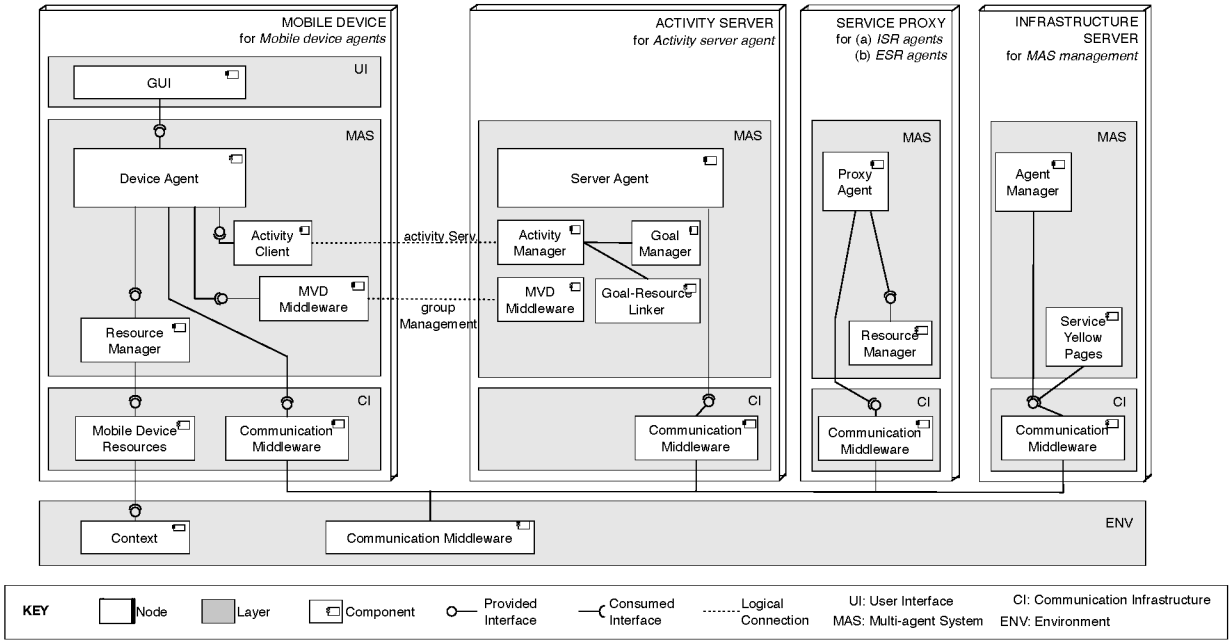

Fig. 5. The proposed Architecture described through a components diagram.

this role define the participants of each group, control the subsequent tasks, track the responses to tasks in the activity, and provide proper feedback to the participants. It is important to notice that this role may be provided by one of more nodes in the system, offering solutions that range form centralized to decentralized management. Multiple activity servers can be used, for example, to offer different levels of specialization. For example, one server manages group compositions and task assignment, while dedicated servers provide specialized feedback for the activity at hand. Another motivation for the use of multiple activity servers is redundancy, where a backup server can take over the management tasks when the active server fails. In some cases, the distributed platform may lack resources that can be relevant for the activity. Under these conditions, an optional service proxy can extend the platform functionalities by providing a link to external service providers. One example is a proxy node to access Google Speech API services in the ENS scenario. Proxy nodes can as well be used for concerns such as learning content provision through a Learning Management System (such as Moodle), and remote information access (such as access to news RSS feeds). Finally, an infrastructure node is required to provide functionalities for the infrastructure, such as agent- and service-registration services, messaging channels and logging repositories.

In addition to the the top-level application components, Fig. 5 also incorporates the *environment (ENV)* layer, which includes such components interacting with the system (i.e. GPS satellites communicating with GPS devices) and channels for communication among devices MAS.

The *Activity Manager* and *Activity Client* components are responsible for managing the activity flow, such as student-teacher interaction mechanisms, to achieve the desired goals. The responsibilities of these components may include the control of the activity flow, presentation of task announcements, feedback and complementary materials needed for the activity at hand and the specification of needed resources.

In order to participate in the MAS, all nodes require a software agent. This component is responsible of executing the node's behavior. This behavior can contain the global activity logic (*Server Agent* in the activity server), individual behaviors for tasks within the activity (*Device Agent* in mobile devices), behaviors that offer access to resources via services and additional behaviors for managing the MAS (*Agent Manager* in the infrastructure server) and connecting to remote services (*Proxy Agent* in the service proxy). Agents in mobile devices have the autonomy to execute local behaviors. This feature covers the requirement of performing individual tasks in the learning activities.

The agents in the MAS can exchange messages. This feature is used for the collaborative aspects that are required in the groups. However, an additional component *MVD Middleware* is necessary to support this collaboration (FR.5). The MVD middleware component manages knowledge with respect to the members belonging to the groups, so meaningful communications can be established only between the relevant agents in the groups. Additionally, nodes offering composed services (through service composition), may use the MVD middleware to determine the location of services in the MVD that are required for such service composition. E.g., a distance calculation requires 2 or more coordinates to be performed. This may imply that 2 GPS resources (offered by GPS services) may be requested. Therefore, a distance-calculation service is offered as a composition of 2 GPS services.

The *Resources Manager* provides the required mecha-

nisms for accessing local resources in a node and resources in the rest of the platform. Among its responsibilities, the resource manager should be able to locate resources needed for the activity and register local resources (as services) in the *Yellow Pages* component, so that other nodes in the platform can make use of them. The Resource Manager can also be used to provide information about services offered outside the MAS, making use of a *Proxy agent* in *Service Proxy* nodes. Finally, the communication infrastructure should contain components that are specific for the communication requirements in the distributed system, to allow communication among agents (*Communication Middleware*), and to interact with the environment, such as reading the GPS coordinates (*Context*). More detailed information of the different components and their roles can be found at the project website[3].

5 Implementation and Assessment

In this section we present two implementations that we created following the architecture design presented in Section 4. The implementations realize the requirements for the activities presented in Section 3. We start by providing a description of the implementations. Afterwards, we describe the metrics used for the assessment, the settings used for the assessment (deployment and relevant environmental conditions) and reason about and link the results to the research questions. Additional material, such as log files and software implementations, is available at the project website.

5.1 Implementation Description

The implementation of the MAS is based on JADE [39]. JADE provides a framework for creating software agents and implementing the desired agent behaviors. JADE is available for multiple platforms, including Android mobile devices. Using JADE we define specific behaviors for agents that run on the mobile devices and on the activity server. On the mobile devices, we implement behaviors for audio recording, position gathering and other domain specific functionalities. On the activity server, we implement a behavior to manage groups of participants, control the activity flow for the groups, and register group performance.

Communication between agents is supported via FIPA[4] messages [39] (which implement the Communication Middleware). Additionally, JADE offers mechanisms for agent and service registration and discovery; i.e., an *Agent Manager* offers agent registration management and a *Directory Facilitator* offers the yellow pages' role. Some specific functionalities are not feasibly deployable on mobile devices, such as a high accuracy speech recognition service for our English learning purposes. Therefore, the system is complemented with multiple instances of proxy agents on servers to provide such services. One example is the *ISR Agent* (*internal speech recognition*), deployed on a server in the MAS, designed to provide Microsoft Speech API (SAPI) services. The Microsoft SAPI is limited to one recognition request at a time. We include a second speech recognition service via a proxy agent that we call *ESR Agent* (*external speech recognition*), to access the Google Speech API.

Organization concerns are deployed among nodes in the MAS using the MVD middleware [20], including servers and mobile devices involved in the platform. The MVD middleware allows local management of groups and identification of existing resources in the platform. I.e., on the Activity Server node, the MVD component declares the number of groups that participate in the activity and the members assigned in each group. On mobile device nodes, the MVD components identify peers and the physical mobile devices that are used by participants in the group. Physical device identification is necessary to determine resources shared in the group. More detailed information regarding the software implementation can be found in the project web.

5.2 Functionality Availability Assessment

ENS Scenario. We studied the desired functionalities in the ENS activity with the involvement of 6 participants divided in two groups. We assessed the functionalities needed for the use cases presented in Fig. 3 by checking their correct behavior and availability. The activity took place in Santiago of Chile and had 40 minutes of duration. We studied the microphone recording (access microphone resource (FR.3)), the audio playing (access audio recording (FR.3)), and the speech recognition feedback functionalities (FR.4), by checking whether or not the functionalities were present or not when required with the mobile devices nodes. The MAS deployment included one dedicated computer (Intel Core I5, 4GB RAM) to offer the *Activity Server* role and offer Microsoft SAPI service (*ISR*), a *Service Proxy* computer (Intel Core I3, 2GB RAM) to access Google Speech API services (*ESR*) and 6 Lenovo tablets with a 2.1GHz processor, 1GB of RAM and running Android 3.1 as *Mobile Devices*.

Service availability issues were only found in the *Check Pronunciation* use case, in which the speech recognition was required. The Microsoft SAPI was defined as the preferred service, and the Google Speech API service as an alternative option when the first was not accessible (blocked processing another audio file). During an initial test (in Table 2), the speech recognition process was initiated 49 times (instances). Two instances were not correctly initiated on the mobile device due to mobile client issues accessing the microphone. The remaining 47 instances fired a request to consume the local speech recognition resource (*Internal in MAS*), from which 32 could be directly processed, indicating a 68.09% of availability of Microsoft SAPI resource. However, 15 needed to be forwarded to a remote speech recognition resource (*External to MAS*) because the Microsoft SAPI was blocked by another ongoing request. From these 15 instances submitted to the External to

3. http://homepage.lnu.se/staff/digmsi/SA-MAS/
4. Foundation for Intelligent Physical Agents. http://www.fipa.org/

MAS resource, 12 were successfully processed (i.e. 80% of availability in Google Speech API resource), while the remaining 3 resulted in a time-out error from the *External to MAS* resource (not complying with NFR.1).

THT Scenario. In a previous experiment of the THT scenario, 12 mobile devices were used in outdoor settings, in order to use GPS's in a real environment. The activity took place in Växjö, south of Sweden and had 55 minutes of duration. The mobile devices were HTC Hero running Android 2.3 with a 600MHz processor. During the experiment, the participants performed distance calculations in groups [13]. However, in some conditions, the mobile devices could not provide accurate measurements for the distance calculations (FR.6). Human involvement was required to recover from the errors, by restarting the mobile devices. This restart process had a duration of around 5 minutes (not complying with NFR.2), since the affected students identified the phone as failing until a manual recovery was effectuated (by a technician). Based on our logs, these failures represented a 1.51% of the expected up-time for distance calculation functionality, and 2 over 4 groups that experienced problems during 5 minutes, which can become frustrating (affecting their future performances) and even jeopardizing the learning activity (as they may learn misleading concepts).

100% availability of resources and services (NFR.2) is far from common in mobile device applications. We address this robustness concern in Section 6.

5.3 Performance and Complexity Assessment

Concerning non-functional requirements, we focused on how the implementations fulfill domain requirements and how architecture software decisions satisfied the desired system response time. We studied performance and complexity of the system measuring communication overhead and identifying the number of components involved in the service consumption, as a metric to study the complexity of a service consumption in a distributed system.

ENS Scenario. We selected the ENS scenario implementation for this assessment, due to the richness of the case. To assess the system performance and overhead (NFR.1), we studied three different cases for resource consumption: whether the resource was found in the same node it originated the request (*Local to the node*), in another node inside the MAS (*Internal in MAS*), or in a node external to the MAS (*External to MAS*). *Local* resource consumption provides a point of reference for the measurements. The last two cases describe situations in which the resource to be consumed is not located in the same node, but is present in another node in the MAS (*Internal in MAS*), or it is external to the MAS and accessed through a Proxy Agent (*External to MAS*). We studied *Local* resource consumption by requesting speech recognition services *from* the *ISR agent* itself. *Internal in MAS* is represented by requests initiated on a mobile device to consume Microsoft SAPI services in the *ISR agent*. *External to MAS* is represented by requests initiated on a mobile device to consume Google Speech Recognition services in the *ESR agent*.

During the test, we analyzed the resource selection complexity with respect to the location of the selected resources. The *Local to the Node* resource consumption for speech recognition service requires the involvement of 7 components, including the following: (1) *Activity Client*, (2) *Activity Manager*, (3) *Device Agent*, (4) *Server Agent*, (5) *MVD Middleware*, (6) *Communication Middleware* and (7) *Resource Manager*. The first six enumerated components are involved during the process of acquiring the assigned number to pronounce. The seventh, offers access to the local resource. For an *Internal in MAS* resource consumption, the number of components involved increases to 10. In addition to the previous six components for number acquisition, four more components are used to access external resources. (7) *Agent Manager*, (8) *Yellow Pages*, (9) *ISR: Proxy Agent* and (10) *ISR: Resource Manager*. The Agent Manager communicated with the Yellow Pages component in order to identify the location of the desired resource. Once this is located, the resource is consumed through the Proxy Agent. The consumption of speech recognition resources External to the MAS requires two additional components, (11) *ESR: Proxy Agent* and (12) *ESR: Resource Manager*, in order to access Google Speech API services. *External to MAS* resources are only requested if *Internal to MAS* resources are not available, therefore, components (9) *ISR: Proxy Agent* and (10) *ISR: Resource Manager* need to be requested as well.

The access to the speech recognition services requires the discovery of resource location, the usage of the network infrastructure that supports the MAS and the use of Service Proxies. When compared with local resource consumptions, the speech recognition service consumption presented an additional overhead. This overhead is due to required internal communication to consume services in the MAS, and additional Internet communications to access *External to MAS* resources. The comparison between the three latency measurements made possible to determine the efficiency of the system in searching and provisioning of resources, which will enable future comparisons in terms of performance with systems providing equivalent functionalities.

The communication overhead was measured by the time required to select a speech recognition resource for *Internal in MAS* and *External to MAS* cases and compared with the time required to consume an equivalent resource *Local to the Node* (see Test2 in Table 2). The audio files to be processed were between 90-100KB, and the network settings consisted of a dedicated 48Mbps WIFI connection and a 2.4Mbps Internet connection. In order to measure the communication overhead variables, 250 speech recognition requests were performed (125 using the *Internal in MAS* resource and 125 using the *External to MAS* resource - Microsoft SAPI). Additionally, we measured 100 executions of service consumptions that were locally originated in the *ISR agent* (server offering the Microsoft SAPI service), to study the overhead implications of using distributed services in

TABLE 2
Results of availability, performance and complexity analysis on the ENS application

	Metric	Local to the Node (Microsoft SAPI)	Internal in MAS (Microsoft SAPI)	External to MAS (Google Speech API)
Test1 (Resource Selection Assessment)	Involved Components	7	10	12
	Resource selection (number of requests)	N/A	47	15
	Resource availability (successful responses [%])	N/A	32 [68.09%]	12 [80%]
Test2 (Communication Overhead)	Number of samples	100	125	125
	Resource selection (avg. in ms)	10	18	18
	Resource usage (avg. in ms)	23	23	24
	Resource response time (avg. in ms)	33	42	480
	Total time (avg. in ms)	43	65	498
	Overhead (avg. in %)	N/A	51.16%	1058.14%
	Re-transmissions (%)	N/A	2.7%	2.9%

the platform in comparison to local resource consumption.

We measured the elapsed time in four different points during the service consumption process. The first metric involves the time required to determine the location of the speech recognition service (*Resource selection* in the table). Local resource discoveries were faster (10ms) than external resource discoveries (18ms). This overhead can be explained by the need of using the Yellow Pages services to determine the location of the desired resource. The second metric offers a view on the time used by the service provider to process the speech recognition (*Resource usage*), and indicates a similarity between Microsoft and Google solutions.

We defined the total time for a service delivery (*Resource response time*) as the time spent for the service execution plus the time spent during communication processes. It is not surprising that this time substantially increased when resources *External to MAS* were accessed. When using the Microsoft SAPI *Local to the Node*, only internal node calls are required (33ms) to consume the service. When accessing resources *Internal in MAS*, the communication overhead (42ms) is limited to a local network usage both used for service discovery through the Yellow Pages and for the service consumption. When requiring resources *External to MAS*, the overhead values are increased (480ms), due to the use of Internet connections, which severely impact latency. Through these numbers we can observe that the usage of a distributed infrastructure based on a MAS can affect the system's performance in terms of delivery times. When services are located in the MAS, these increases are negligible in terms of absolute values (65ms vs. 43ms) and due to Yellow Pages requests and WIFI communications. However, the latency increases considerably when services need to be found outside the MAS, and Internet communications are required. In this case, the latency has a ten times growth to consume resources *External to MAS*. A combination of such delays may compromise the usability of the learning application.

When consuming MAS services, an alternative approach is to consider direct connections between agents, avoiding *Yellow Pages* services. Such approach would improve performance and latency measurements (NFR.1), but would affect the system's maintainability and extensibility, as resources should be known in forehand by the agents involved in the platform.

The use of *External to MAS* resources could have more implications besides performance and latency. Even response-time within desired time constrains is a variable for robustness, the reliability of the services should also be considered. Reliability could be not guaranteed, which becomes a threat with respect to system robustness, and can lead to risks for the learning activity. One example is the use of Google Speech API in our learning scenario. In this case, services are not managed by the platform, but external actors are involved (Google). The availability of the external service (due to high request load, maintenance processes, system failures, among others) becomes an uncertainty. Additionally, for the THT scenario, we found GPS reliability issues, which may influence the outcomes of the activities. In the following section we present an extension of the platform that provides mitigation mechanisms to provide guarantees for certain quality concerns, such as robustness in our case.

6 Self-adaptation

The assessment study reveiled a lack of resource availability in particular cases (NFR.2), mostly due to uncertainty regarding the resource and service state. Resource availability and system robustness are critical concerns that must be provided in collaborative mobile learning applications, and widely present in multiple other domains. New system's design and implementation to address robustness concerns can imply an increase of systems' complexity. To avoid this complexity increase, state of the art advocates the application of self-adaptation (hereafter, SA) mechanisms [19], which principle is based on the separation of concerns. A self-adaptive system is divided into two subsystems: the managed system (which supports functions that are specific to the domain) and the managing system (responsible for contributing with quality properties to the system) [40].

SA mechanisms are placed on top of a managed system, and aim at providing selected quality concerns via designed adaptations. A well-established approach to realize SA is through MAPE-K (Monitor-Analyze-Plan-Execute and Knowledge) feedback loops [41]. A MAPE-K loop is composed of four main roles that *Monitor* the environment and managed system states, *Analyze* the completeness of the system with respect to the desired goals, *Plan* mitigation

actions in order to address identified errors, and *Execute* the selected plan. These four activities are supported on a *Knowledge* base that provides a level of abstraction of the activity, managed system, environment and goals [42].

In our case, robustness is achieved (only) if the groups involved in an activity include the number of resources that are required for the activity goals (FR.3, FR.4), and these resources provide the desired service quality (NFR.2). Due to space limitations, we use the THT scenario as a running example in this section. The THT system can be considered robust when the required number of GPS resources are available in a group and these resources provide their GPS location under a specified accuracy. Depending on the task at hand, the required number of GPS resources may vary. Also their accuracy tolerance can depend on the characteristics of task. In Fig. 6, we illustrate an extension of the previous MAS architecture, in order to guarantee robustness through a SA layer, based on [40]. In order to center the focus on SA aspects, the UI and ENV layers have been removed from the figure. In our particular case, the managed system represents the distributed system that we described in Section 5. The managing system is responsible for analyzing the current behavior of the managed system and, in case of service failures, adopting the necessary measures to maintain system's robustness.

Two SA loops provide robustness for two separate concerns. A first MAPE-K loop is concerned on the consistency of the accuracy that GPS resources provide (FR.3, NFR.2). In other words, the first feedback loop is concerned with managing the GPS service availability based on the current GPS service quality. The second MAPE-K loop is concerned with managing the number of GPS resources in a group (FR.4, NFR.2), so it can achieve the activity goals.

Below, we provide a description of components in the managed and managing systems [40], that are required to provide SA. This shows the required modifications in a legacy system in order to provide points for monitoring and adapting the managed system.

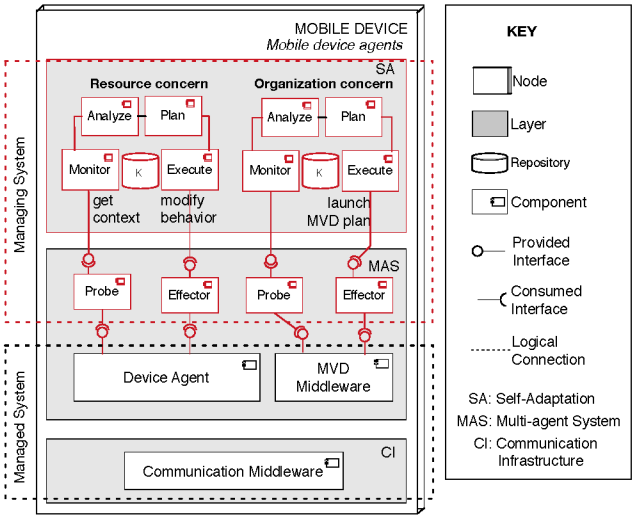

Fig. 6. Self-adaptation components on mobile nodes.

6.1 The Managed System

The managed system requires domain specific components that provide the necessary functions necessary for the activity. In addition to these components, it is necessary to include points of interaction to the managed system so that SA can be applied. There are two types of components that are the touch points for interactions between the managed and managing systems: *Probes* and *Effectors*.

The role of Probe components is to gather information from the system, to support the reasoning processes performed by the managing system. Probe components are expected not to interfere with the behavior of the managed system. In particular, for the THT scenario, probe components have been implemented to collect data of the GPS accuracy in the mobile devices, the current state of the organizations in the groups (represented in the MVDs), and the activity requirements. Therefore, probes must have access to read certain parameters from the Device agent component (current acquired GPS accuracy) and from the MVD middleware component (current organization deployment).

Analogously, the *Effector* components deal with enacting the adaptation tasks, decided by the managing system, to the managed system. For THT scenario, Effectors have been implemented to manage the GPS service in accordance to the GPS accuracy, deactivating or activating the GPS service when required, and to modify the group compositions when required. Probes and effector components are the only communication points between the managed and managing system, which enable system monitoring and carrying out adaptation decisions to address the quality concerns.

6.2 The Managing System

The managing system focuses on provisioning quality properties that are desired in the managed system. Our particular quality concern in the collaborative mobile learning activities is providing robustness of the system in the face of resource failures.

In order to provide SA for such concern, it is necessary, first, to *Monitor* the necessary system and environment parameters that can have an influence in our concern. In the case of The Hidden Treasure scenario, it becomes necessary to monitor the current state of the managed system in terms of accuracy of each of the GPS resources and the number of GPS resources in a group, and monitor the current activity requirements in terms of the required GPS accuracy and required number of members in a group to fulfill the activity goals. Second, it becomes necessary to *Analyze* the correctness of the managed system behavior. I.e., it is necessary to analyze the GPS service quality with respect to the activity requirements; and the completeness of a group, in terms of GPS resources. Third, a *Plan* needs to be selected to mitigate detected issues. An example is to deactivate a GPS service if its performance is not accurate, or determine an additional GPS resource to be integrated in a group if this is found incomplete. And fourth, *Execute* the planned actions to the managed system.

Given the nature of the scenarios, the suggested solution is based on a distributed system. Therefore, it becomes necessary that the components of the managing system are distributed among the nodes of the MAS. I.e., in order to self-adapt GPS services, it is necessary that nodes in the MAS locally contain the *Monitor*, *Analyze*, *Plan* and *Execute* components. For concerns that refer to distributed nodes, it becomes necessary to deploy the SA components in a distributed manner as well [40]. For instance, for group SA, it becomes necessary to locate *Monitor* and *Execute* components in each of the members of the organization, while the *Analyze* and *Plan* components can be located on specific member of the group, following a master-slave pattern approach [40].

In our studies, aspects such as the service quality in the use of GPS coordinates and the availability of audio and microphone service are some of the parameters monitored on mobile device nodes and the quality (in terms of availability) of speech recognition service is a parameter monitored on the server side. The SA mechanisms may be customized to address the quality concerns that are desired for the system at hand. Thus, SA mechanisms for GPS accuracy may be different from the SA mechanisms used for microphone recording services.

6.3 Self-Adaptation Assessment

We present the assessment process of self-adaptation applied to THT scenario to provide robustness for GPS service reliability and group completeness due to GPS inaccuracy.

A first set of SA mechanisms has been designed to manage the GPS services. SA is based on GPS accuracy, deactivating the GPS service and un-publishing it (from the *Yellow Pages*) if the GPS quality is not satisfying or activating and publishing the GPS service if the quality recovered. The GPS service self-adaptation is deployed in all the mobile devices involved in the activity and it is locally and individually managed in the nodes. In other words, the SA mechanisms are locally controlled and implemented on each of the mobile devices, providing autonomy and decentralization to the SA solution. The GPS service SA is represented in Fig. 6, by the *Resource concern* MAPE loop on the mobile devices SA layer.

The deactivation of GPS services can lead to groups having lack of resources necessary to cover the MVD requirements. Therefore, a second MAPE-K loop is designed to monitor the completeness of the groups in the activities and, to mitigate potential issues, by incorporating additional phones to the group when required (*Organization concern* MAPE loop on the mobile devices). This SA loop is shown in Fig. 6 with the *Organization concern* MAPE-K loop. One device in each organization is selected as a master device to be in charge of the SA decisions. This device is then responsible to gather information with respect to the organization state, the activity requirements (in terms of number of GPS resources) and to determine mitigation plans when required. Due to the distributed deployment of a group, the Monitor role is distributed among the nodes in the organization, which means that all the mobile devices are in charge of monitoring the current GPS service state and to notify the master device when changes are monitored. Equally, the *Execute* role is distributed in the organization, in order to transfer the mitigation actions across the members in the organization. The interested reader can find more information with respect to this self-adaptation behavior in [43].

The assessment is performed in lab settings based on The Hidden Treasure scenario, with an activity that requires 3 GPS resources per group, and a minimum accuracy of 10 meters for the GPS service. We assess the GPS self-adaptation through Test1 with one mobile device running Android 2.3 on a 1.2GHz processor and 1GB RAM. We use a dedicated 3MB/s WIFI connection for the communication. In order to study the self-adaptation in detail, we include an additional component that emulates the GPS behavior. This component emulates GPS locations and accuracies, which allows us to provoke GPS inaccuracies on devices and to study SA in high stress scenarios (in terms of GPS failures). A total of 200 failures and recoveries were emulated on the GPS services. 100 of these failures emulate the behavior of a GPS module affected by cloudy conditions, giving inaccurate measurements (errors $>$10 meters) during 3 seconds. This behavior is played in a loop that lasts 42 sec. The other 100 errors emulate a flapping GPS device behavior having inaccuracy measurements during 0.5s every 22 seconds.

SA incurs with a processing overhead, not only during the adaptation processes (to mitigate risks when a failure has been detected), but also during desired system's behavior due to the periodic monitoring process. An initial cost of self-adaptation is induced by a *GPS probe* component that would periodically monitor the GPS service quality, with a period of 500 ms (milliseconds). The figures represented in GPS columns in Table 3 offer an overview of the additional overhead produced by the implemented SA layer and the efficiency of the SA behaviors with respect to self-healing. For local GPS service SA, the processing overhead does not result in a considerable increase. It is interesting to notice that, in this first SA experiment, all the failures and recovers were correctly treated by the implemented SA mechanism. In the worst-case scenario, the SA mechanism requires 411 ms to adapt the GPS service in the system. This number was always lower than the changes in the environment, which updated the GPS state on a 500 ms frequency-basis.

TABLE 3
Results of the self-adaptation mechanisms.

	Test1 (GPS)		Test2 (MVD)	
Variable	**Avg.**	**Max.**	**Avg.**	**Max.**
Self-Adaptation overhead (%)	1.38%		3.15%	
Failure detection (ms)	14	63	511	973
Alternative plan identification (ms)	19	78	1338	15994
Correction application (ms)	75	411	1435	16197
Effectiveness of failure correction (% of resolved failures)	100%		97.50%	

With respect to the organization robustness concern, the SA was expected to identify lack of resources in a group and heal these states by autonomously integrating a new device in the group. In our lab settings, a lack of GPS resources in a group can be originated by GPS service failures. Test2 was performed to assess the organization SA. Five mobile devices were involved in this test, emulating a scenario with groups of three participants and two additional spare devices for contingency issues. In Test2, the GPS behavior were defined to have a 10% failure rate (in terms of accuracy) during periods between 10s and 20s. The results are shown in the MVD columns in Table 3.

Based on the scenario requirements defined in Section 3, we consider the SA to be successful when a group can be recovered in less than 10 seconds (defined by previous experiences [13]). During our test, 40 failures were provoked on GPS resources in the MVD. 39 group incompleteness were recovered in less than 2s. However, the remaining consumed around 16s for the recovery. In this case, the self-adaptation had an effectivity of 97.50%.

The network infrastructure plays an important role in the SA process due to the distributed deployment of the nodes, which implies higher periods for managing organizational failures that when compared to local self-adaptation. For example, the detection of a failure in a slave (a GPS being turned off in one member of the organization) requires a communication process between the slave and the master and it took up to 973 ms.

The SA mechanisms presented in this section have been designed to mitigate potential risks that the system may face at runtime. Although the mechanisms have demonstrated a high level of effectiveness for failures, not all the failure instances are correctly addressed. Therefore, it is necessary to perform further efforts in analyzing the SA mechanisms and understanding the environmental conditions that can lead to the not-addressed cases. This aspect demands the use of rigorous methods (such as the application of formal methods [43]) to specify the behaviors of system and SA mechanisms. However, the potential impact of the SA is evident. Human involvement can be extremely reduced, moving from 5 minutes required to manually recover a group to an average of 1.435 seconds with SA.

7 Conclusions and Future Efforts

Mobile devices are quickly being adopted in education due the numerous possibilities they offer, such as personalization and collaboration, among others [44]. However, these technologies bring uncertainty. In this work we focused on two critical aims, resource-sharing and robustness. We proposed a software architecture to cover these aims to increase the range of possible scenarios in collaborative mobile learning activities. The key aspect of this architecture is to combine the benefits of multi-agent system solutions together with self-adaptive mechanisms.

One critical aspect in collaborative mobile learning scenarios is the autonomy of actions taken by the participants in the activity combined with resource sharing aspects for collaboration. The RQ1 *"Which are the most suitable characteristics that a software architecture should have to offer resource sharing for collaborative mobile learning activities?"* concerns the software architecture aspects that are necessary to support such activities. We propose the use of a distributed system architecture composed by mobile devices and servers to support the activity. Based on a multi-agent system (MAS) architecture, we propose the design of a software architecture that offers one device per participant with a certain level of autonomous behavior (for individual performance aspects) extended with communication features (in order to support collaboration with the rest of the participants in the activity). Additionally, a distributed architecture solution, in which each participant carries a personal mobile device, allows new mobile devices to be integrated to the system dynamically by registering them into the MAS. In terms of resource access, our approach allows that nodes in the system can access local and remote resources, as soon as they are offered by nodes in the distributed system.

With respect to the RQ2 *"Which software elements are necessary for collaborative mobile learning applications, in order to satisfy resource-sharing requirements for individual performance and collaborative interactions?"*, we want to provide a more detailed level of description of the structural architecture design for a collaborative learning application. We suggest the use of a set of components for local and distributed resource management, node behavior management and organization management that we illustrate in Fig. 5. Efforts like Huang & Yin [22], and Rao et al., [24] have tackled the resource management issue by extending mobile device capabilities with cloud-computing approaches. Even though these efforts can extend the set of functionalities present on a mobile device, these still lack features with respect to resource-sharing between mobile devices. Johnson's group [25], [26] provides an innovative solution in the mobile learning community, that brings a peer-to-peer based solution to offer resource-sharing between devices. However, this solution does not provide instant communication between peers, but all mobile devices are supported by a server-instance where resources (mainly data) are stored and pulled in a periodic basis. Our solution is one step ahead with respect to Johnson's approach. A solution based on MAS is presented by Khan et al. in [29]. This solution uses agents with the purpose of sharing services located in each of them. In their case, the creation of organizations is specific for the activity, existing centralized negotiators, coordinators and manager agents for their creation. We also suggest the use of a MAS-based solution to allow peer-to-peer and low-latency communication, in order to accomplish the response-time restrictions that are set for mobile learning scenarios. However, we offer a decentralized solution in which organization concerns are distributed among the nodes in the MVD middleware. This particular aspect is the concern expressed in the RQ3 *"How to guarantee robustness with respect to supporting collaboration through the mobile applications in dynamic environments?"*. Through our experiments, we

identified that a distributed architecture based on mobile MAS can cover response-time constraints that are necessary for collaborative mobile learning applications. Moreover, due to the MAS based approach, this solution has potential for good scalability of learning scenarios. Future research will focus on providing evidence in medium and large scale scenarios.

Finally, we have focused on robustness concerns for collaborative mobile learning applications to complete the answer to RQ3. We have presented an extension of the MAS solution with self-adaptation capabilities towards resource and organization robustness aspects, which offer guarantees for specific QoS. Additionally, we studied the processing overhead that self-adaptation mechanisms can imply on a legacy system, showing that SA-related overheads do not create a noticeable negative impact on the system. On the contrary, the results show that the implemented self-adaptation mechanisms (for THT scenario) could self-adapt service states in around 75 ms and organization issues in 1300 ms. This contrasts to our previous experiences, where human involvement was necessary, and fixing service and organization could require more than 5 minutes.

To the best of our knowledge, there are few relevant efforts that have focused on failure recovering in collaborative mobile learning applications. More studies should emphasize providing quality properties to learning activities.

The designed self-adaptation mechanisms imply changes in the behaviors of the nodes in the MAS. This aspect becomes more relevant when working in a distributed environment, as it may affect the behavior of groups in the system or, in the worse case scenario, even the overall system. Formal methods could be used to evaluate the correct design of a system, and verify that the desired goals and the required property qualities are achieved through the self-adaptation processes [45].

We believe the presented work is a first step towards the definition of a reference architecture for the field, which covers the aims related to resource-sharing and application robustness.

In our future efforts, we plan to study new possibilities regarding pedagogical activities that the proposed software architecture offers by covering resource-sharing and robustness aims. One concrete scenario is the creation of an agent, to be used on the teacher's device, with a dashboard interface to visualize relevant information regarding the students performance. This agent would facilitate the teacher in the role of supporting, on time, students with needs.

REFERENCES

[1] M. Sharples, J. Taylor, and G. Vavoula, "Towards a theory of mobile learning," *Proceedings of mLearn 2005*, vol. 1, no. 1, pp. 1–9, 2005.

[2] A. Kukulska-Hulme, *Mobile usability and user experience*. Routledge, 2005, pp. 45–56.

[3] M. Sharples, I. Arnedillo-Sánchez *et al.*, *Mobile Learning: Small devices, Big Issues*. Springer Netherlands, 2009, pp. 233–249.

[4] M. Sarrab, L. Elgamel, and H. Aldabbas, "Mobile learning (m-learning) and educational environments," *International Journal of Distributed and Parallel Systems*, vol. 3, no. 4, pp. 31–38, 2012.

[5] H. Ogata, "Computer supported ubiquitous learning: Augmenting learning experiences in the real world," in *WMUTE '08*. IEEE, 2008.

[6] G. Zurita and M. Nussbaum, "Computer supported collaborative learning using wirelessly interconnected handheld computers," *Computers & education*, vol. 42, no. 3, pp. 289–314, 2004.

[7] M. Sharples, J. Taylor, and G. Vavoula, "A theory of learning for the mobile age," in *Medienbildung in neuen Kulturrumen*, B. Bachmair, Ed. VS Verlag fr Sozialwissenschaften, 2010, pp. 87–99.

[8] H. Beetham and R. Sharpe, *Rethinking pedagogy for a digital age: Designing for 21st century learning*. routledge, 2013.

[9] U. Lucke and C. Rensing, "A survey on pervasive education," *Pervasive and Mobile Computing*, 2013.

[10] O. Pettersson and B. Vogel, "Reusability and interoperability in mobile learning: A study of current practices," in *WMUTE '12*, 2012.

[11] L. Bollen, S. Eimler *et al.*, "Enabling and evaluating mobile learning scenarios with multiple input channels," in *Collaboration and Technology*, ser. LNCS. Springer, 2012, vol. 7493.

[12] J. S. Fu, "ICT in Education: A Critical Literature Review and Its Implications," *International Journal of Education and Development using Information and Communication Technology (IJEDICT)*, vol. 9, no. 1, pp. 112–125, 2013.

[13] D. Gil de la Iglesia, J. Andersson, M. Milrad, and H. Sollervall, "Towards a decentralized and self-adaptive system for m-learning applications," *Proceedings of WMUTE'12*, pp. 162–166, 2012.

[14] D. Vogel, D. Kennedy, and R. C. W. Kwok, "Does using mobile device applications lead to learning?" *Journal of Interactive Learning Research*, vol. 20, no. 4, pp. 469–485, 2009.

[15] P. Damin-Reyes, J. Favela *et al.*, "Uncertainty management in context-aware applications: Increasing usability and user trust," *Wireless Personal Communications*, vol. 56, no. 1, pp. 37–53, 2011.

[16] D. G. Zhang and X. D. Zhang, "A new service-aware computing approach for mobile application with uncertainty," *Applied Mathematics & Information Sciences*, vol. 6, no. 1, 2012.

[17] A. Neyem, S. F. Ochoa, and J. A. Pino, "A patterns system to coordinate mobile collaborative applications," *Group Decision and Negotiation*, vol. 20, no. 5, pp. 563–592, 2011.

[18] J. Kramer and J. Magee, "Self-managed systems: an architectural challenge," *FOSE '07*, pp. 259–268, 2007.

[19] B. Cheng, R. de Lemos, and Others, "Software engineering for self-adaptive systems: A research roadmap," in *Software Engineering for Self-Adaptive Systems*, ser. LNCS. Springer, 2009, vol. 5525.

[20] D. Gil de la Iglesia, J. Andersson, and M. Milrad, "Mobile virtual devices for collaborative m-learning," in *Proc. of the 18th ICCE'10*. Asia-Pacific Society for Computers in Educations, 2010.

[21] H. T. Dinh, C. Lee *et al.*, "A survey of mobile cloud computing: architecture, applications, and approaches," in *Wireless communications and mobile computing*. Wiley Online Library, 2011.

[22] S. Huang and H. Yin, "A new mobile learning platform based on mobile cloud computing," in *Advances in Future Computer and Control Systems*, vol. 159. Springer, 2012, pp. 393–398.

[23] P. Pocatilu, F. Alecu, and M. Vetrici, "Measuring the efficiency of cloud computing for e-learning systems," *WSEAS Transactions on Computers*, vol. 9, no. 1, pp. 42–51, 2010.

[24] N. M. Rao, C. Sasidhar, and V. S. Kumar, "Cloud computing through mobile-learning," *International Journal of Advanced Computer Science and Applications*, 2012.

[25] D. Johnson and I. M. Bhana, "Pervading collaborative learning with mobile devices," in *Technological Advances in Interactive Collaborative Learning*. CRC Press, 2012, p. 59.

[26] D. Johnson *et al.*, "A platform for supporting micro-collaborations in a diverse device environment." *iJIM*, vol. 3, no. 4, pp. 8–16, 2009.

[27] H. A. Neyem, S. F. Ochoa, and J. A. Pino, "Integrating service-oriented mobile units to support collaboration in ad-hoc scenarios." *J. UCS*, vol. 14, no. 1, pp. 88–122, 2008.

[28] M. Broy, I. H. Krüger, and M. Meisinger, "A formal model of services," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 16, no. 1, p. 5, 2007.

[29] A. Khan, M. Matskin, and C. Rossitto, "Towards a places and spaces based city-wide mobile learning through multi-agent support," *Proc. of Digital Ecosystems and Technologies*, pp. 164–169, 2011.

[30] O. Garcia, R. S. Alonso *et al.*, "Cafcla, a framework to ease design, development and deployment ami-based collaborative learning applications," in *CISTI '12*. IEEE, 2012, pp. 1–6.

[31] M. Wooldridge, *An introduction to multiagent systems*. John Wiley & Sons, 2002.

[32] R. Haesevoets, D. Weyns, and T. Holvoet, "Architecture-Centric Support for Dynamic Service Collaborations," *ACM Transactions on Software Engineering and Methodology*, vol. 23, no. 1, 2013.
[33] A. Andronico, A. Carbonaro *et al.*, "Integrating a multi-agent recommendation system into a mobile learning management system," *Proc. of Artificial Intelligence in Mobile System*, pp. 123–132, 2003.
[34] C. Infante, P. Hidalgo, M. Nussbaum, R. Alarcón, and A. Gottlieb, "Multiple mice based collaborative one-to-one learning," *Computers & Education*, vol. 53, no. 2, pp. 393–401, 2009.
[35] J. F. Calderón, M. Nussbaum *et al.*, "A single-display groupware collaborative language laboratory," 2014, accepted in Interactive Learning Environments.
[36] Z. Handley and M. J. Hamel, "Establishing a methodology for benchmarking speech synthesis for computer-assisted language learning (call)," *Language Learning & Technology*, vol. 9, no. 3, 2005.
[37] P. Ciancarini, A. Omicini, and F. Zambonelli, "Multiagent system engineering: The coordination viewpoint," in *Intelligent Agents VI. Agent Theories, Architectures, and Languages*, ser. LNCS. Springer, 2000, vol. 1757, pp. 250–259.
[38] J. Ferber, *Multi-agent systems: an introduction to distributed artificial intelligence*. Addison-Wesley Reading, 1999, vol. 1.
[39] F. Bellifemine, G. Caire *et al.*, "Jade: A software framework for developing multi-agent applications. lessons learned," *Information and Software Technology*, vol. 50, no. 1, pp. 10–21, 2008.
[40] D. Weyns, B. Schmerl *et al.*, "On patterns for decentralized control in self-adaptive systems," in *Software Engineering for Self-Adaptive Systems II*, ser. LNCS, vo. 7475. Springer, 2013, pp. 76–107.
[41] J. Kephart and D. Chess, "The vision of autonomic computing," *IEEE Computer Society*, vol. 36, no. 1, pp. 41–50, 2003.
[42] D. Weyns, S. Malek, and J. Andersson, "FORMS: Unifying Reference Model for Formal Specification of Distributed Self-Adaptive Systems," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. V, 2011.
[43] D. Gil de la Iglesia and D. Weyns, "Guaranteeing robustness in a mobile learning application using formally verified mape loops," in *SEAMS*, 2013, pp. 83–92.
[44] A. Echeverría, M. Nussbaum *et al.*, "Face-to-face collaborative learning supported by mobile phones," *Interactive Learning Environments*, vol. 19, no. 4, pp. 351–363, 2011.
[45] D. Weyns, M. U. Iftikhar *et al.*, "A survey of formal methods in self-adaptive systems," in *C3S2E '12*. ACM, 2012, pp. 67–79.

Didac Gil de la Iglesia Didac is a PhD in Computer Science with the specialization in Media Technology at the Faculty of Technology at Linnaeus University (LNU) in Sweden. He holds a Master Degree in Mobile Communications from the Politechnical University of Catalonia (UPC) in Spain. His current research focuses on aspects related to mobile collaborative systems by sharing resources between mobile devices. Currently, Didac is using an Agent-based approach for resource and service sharing in a mobile devices-based infrastructure. His main areas of interest include Technology Enhanced Learning, Mobile and Ubiquitous Technologies, Software Engineering, Distributed Systems and Software and Technology Design.
didac.gil-de-la-iglesia@lnu.se

Juan Felipe Calderón Juan Felipe Caldern is PhD Candidate in Computer Science and Adjunct Instructor at the School of Engineering of the Pontificia Universidad Catlica de Chile. His research and teaching is focused on software engineering, software design, distributed systems and computer-supported collaborative learning and new strategies for computer science teaching. He received a MsC and a Bachelor degree on Civil Engineering from the Pontificia Universidad Católica de Chile. Contact him at Pontificia Universidad Católica de Chile, Vicuña Mackenna 4860, 56223544440, Santiago Chile; *jfcalder@ing.puc.cl*

Danny Weyns Danny is a Full Professor at the Department of Computer Science of Linnaeus University, Växjö campus, where leads the AdaptWise research group. His research interest is in software engineering of self-adaptive systems, including formalisms and design models to realize and assure self-adaptation for different quality goals. Domains for empirical validation of research results include smart homes, multi-robot systems, and mobile learning applications. Before joining Linnaeus, Danny was affiliated with DistriNet Labs at the Katholieke Universiteit Leuven, where he received a Ph.D for work on multiagent systems and software architecture. After his Ph.D, Danny Weyns worked as a postdoc fellow, funded by the Research Foundation Flanders. *danny.weyns@lnu.se*

Marcelo Milrad Marcelo is a Full Professor of Media Technology at the Faculty of Technology at Linnaeus University (LNU) in Sweden. He is also the director of the Center for Learning and Knowledge Technologies (CeLeKT). His current research interests include the design of learning environments to support learning about complex domains, collaborative discovery learning and the development of mobile and wireless applications to support collaborative learning. Marcelo he has been a program committee member in a number of international scientific conferences and an editorial board member in scientific journals in the field of Technology Enhanced Learning. Pr. Milrad has acted as executive member of the IEEE Comp. Soc. Technical Committee on Learning Technology (LTTC) and he is one of the initiators of the IEEE International Conference on Wireless and Mobile Technologies in Education (WMTE). *mmilrad@lnu.se*

Miguel Nussbaum Miguel Nussbaum is full professor for Computer Science at the School of Engineering of the Pontificia Universidad Catlica de Chile. His pedagogical methodologies for in classroom interactive (collaborative) learning has been used in schools in Argentina, Chile, Brazil, Guatemala, India, England, USA and Uruguay, and is endorsed by UNESCO. He received a Master Degree from the Georgia Institute of Technology, Atlanta, and a PhD from the Eidgenössische Technische Hochschule, Zürich, Switzerland. Contact him at Pontificia Universidad Católica de Chile, Vicuña Mackenna 4860, 56223544440, Santiago Chile; *mn@ing.puc.cl*