# Supporting context-aware collaborative learning through automatic group formation

R. Messeguer, E. Medina, D. Royo, L. Navarro
Universitat Politècnica de Catalunya, Spain

P. Damian-Reyes, J. Favela
CICESE, Mexico

## Abstract

Collaborative learning is based on groups of students working together with traditional and computer-based tools or applications. We have found that to make these supporting applications more effective we need to address the problem of automating group awareness in CSCL applications by estimating group arrangements from location sensors and the history of interaction. This contextual information can enable the construction of applications that facilitate communication among group members in synchronous and collocated collaborative learning activities. We used data traces collected from the study of students' behavior to train and test an intelligent system. Results show that context-information can be effectively used as a basis for a middleware for automating group management. Inferring group membership is technically feasible, can be integrated in group-support applications and can be used in real-world settings.

## 1. Introduction

The ambient intelligence (AmI) vision describes an ubiquitous environment which is furnished with computational artifacts that remain in the background of our lives and that have intelligent capabilities to support user-centered activities [14]. An intelligent environment is ubiquitous in the sense that it enhances the physical environment with heterogeneous computational and wireless communication devices naturally integrated and, at the same time, invisible to the user [15]. Hence, AmI applications need intelligent capabilities to be adaptive to users and reactive to context in order to provide high quality services based on their preferences. Context-awareness allows AmI environments to take on the responsibility of serving users, by tailoring itself to their needs, and perform tasks according to the nature of the physical space.

A particular ambient that can benefit from using context-aware applications are Cooperative learning activities. Cooperative learning is an instruction method based on students working together in small groups to accomplish shared learning goals [11]. Cooperative learning entails a dynamic setting where multiple parallel groups of students join and disband rapidly to form new groups. In addition, beyond paper and pencil, students may use mobile devices –such as laptops or PDAs, with wireless network connections– combined with computer applications, which provide support not only in the creation and manipulation of data, but also in facilitating and encouraging cooperation among the students. However, in this scenario there is a potential barrier on the effort required for the integration of physical and digital objects. Bridging this barrier by sharing contextual information makes interaction across real-world and computer-based objects invisible to the user [10].

Computer-Supported Collaborative Learning (CSCL) applications are used in collocated cooperative learning settings to support group tasks in planned and unplanned situations. Such applications require up-to-date contextual information about the groups that are participating in the collaborative activity. Since this information is changing continuously, its manual introduction may be of concern to the already overloaded teacher and introduce delays and additional burden to any participant in a real-time collaboration. Thus, there is need to automate the detection of changes in the physical world. An intelligent environment can use sensors to detect changes and learn about the behavior of participants and automatically detect group membership.

In this paper, we explore how intelligent environments can detect automatically the arrangement of participants in groups for unplanned and planned situations. That is, if students participate in collaborative activities –either within the classroom or in the campus–, the

system could proactively set up a certain group before members explicitly request for it.

Our main goal is to provide support to the students' activities using an Intelligent System, which takes into account contextual information to estimate group membership. In addition, we present a case study where we show how this information can be used in collaborative applications. The contributions of this paper are:

- We define a contextual model: a list of requirements and relationships on basic contextual information that every intelligent system needs to automatically detect the creation of groups. (Sections 2 and 4)
- We propose both, training and estimating processes for the intelligent system, in order to obtain high accuracy in the group membership estimations. (Section 6)
- We demonstrate that predicting groups' membership is technically feasible and that it can be easily incorporated in CSCL applications. (Sections 7 and 9)

## 2. Context model for group awareness

To identify relevant contextual information we have used the framework for CSCL/CSCW system awareness proposed in [5] as starting point. In order to achieve our goals, we have considered information related to the identification of individuals and groups enriched with time and place information.

According to [3] our context-aware scenario is a model Class C: Context as a matter of user activity. The focus of this class of models is on "what the user is doing", consequently context history and reasoning are important issues. Time and space are considered relevant as far as they provide information about the user's current activity. The context definition is in general centralized and the user is the subject of the model. Automatic learning is used to guess user activity from sensor readings.

Based on [7], we defined a context model (Figure 1) that divides the current contextual information –identified by a timestamp– into three parts: Location, User identity and Group activity. This context model is centralized and users periodically update their sensed context information and record every context update with different timestamps.

The three parts of the model have the following relationships: At a given moment (timestamp) a User is in a particular Location with other users –neighbors–. By reasoning on these context components we can infer the Group activity –group membership estimation–.

This context model can bring us the following features [3]:

- Context reasoning: indicates whether the context model enables reasoning on context data to infer properties or more abstract context information (e.g., to deduce user activity combining sensor readings).
- Automatic learning features: highlights whether the system, by observing user behavior, individual experiences of past interactions with others, or the environment, can derive knowledge about the context (e.g., by studying the user's browsing habits, the system learns user preferences).

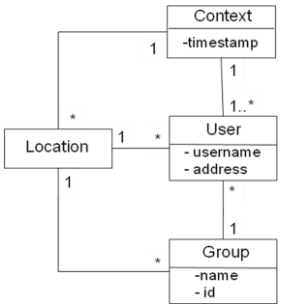


Figure 1. Context Model

## 3. Context architecture for dynamic group prediction

The architecture of our system (Figure 2) consists of three components deployed in two nodes. The first node, with which the user directly interacts, includes the context provider component –where information about the users' context is gathered. The second node contains the intelligent system where two other components reside: context collector –where contextual information is pre-processed– and the machine learning engine –where contextual information is consumed–.

At the user side, context information is gathered to generate a fingerprint for that user that is sent to the system periodically or every time a change occurs.

When the system receives new contextual information related to such user, the context collector pre-processes this information and generates the inputs to the machine learning component which estimates a specific group formation and delivers this estimation to the system in order to automatically create the group and set up the collaborative environment and the CSCL applications.

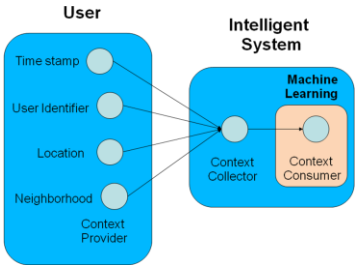


Figure 2. Context Architecture

The whole prediction process works as follows: the user triggers some actions related to group management: creating, joining and leaving groups. All these actions are also sent to the intelligent system, which uses this data to learn about user behavior –with who collaborate and when the collaboration is done–. When new contextual information from the user arrives to the intelligent system, it uses what it had previously learnt to predict if the user is going to join a particular group, and automatically set up the group's working environment.

## 4. Contextual information

We considered information related to the identification of individuals and groups [5] and we enriched it with time and location information. This contextual information –gathered by the context provider– is always associated to the user id and can be static –acquired only once– or dynamic –obtained periodically or every time it changes–.

We analyzed which contextual information was relevant for training a machine learning and we discarded information that did not improve recognition accuracy. For instance, we found a significant increase in accuracy by including in the contextual information all the Bluetooth devices (physical proximity) sensed for the user, instead of only the devices of the students that had previously collaborated with him. In the experiments, we confirmed that such information was crucial for the machine learning to be able to accurately identify the context (Table 4).

We finally selected the following items as relevant for predicting group creation (Figure 2):

- Timestamp: the time and the day of the week.
- User identifier: a unique Id. based on the identity of the user and his mobile device. (e.g. username and Bluetooth MAC address).
- Location: based on the access point fingerprint [4], our system obtains information about where users are placed: classroom, cafeteria, library, etc.
- Neighborhood: list of all sensed Bluetooth' MAC addresses.

## 5. Machine learning

In a previous research work [13], we addressed the same problem of how to estimate group membership. We used a strategy that did not include any machine learning algorithm. The limitation of this solution was its static character (known users, familiar places, specific configuration of the algorithms for those users and places, etc.). In this work, we seek a solution where the system learns from context changes (users, locations and time) and could adapt itself dynamically according to such changes. In order to obtain a dynamic behavior, we trained and evaluated two machine learning algorithms to detect when collaboration among people starts in the real-world and thus a group should be created in the computer-based environment. For this purpose, we used the Weka workbench system [9], a framework that incorporates a variety of learning algorithms and some tools for the evaluation and comparison of the results. The two algorithms used are: Instance-based learning (IBL) and Bayesian Network (BayesNet).

Both algorithms are very simple and have either few or no parameters to be tuned. They also produce classification models that can be easily interpreted. We chose IBL because it is a simple, yet robust learning algorithm, can tolerate noise and irrelevant attributes and can exploit inter-attribute relationships. On the other hand, we selected BayesNet as a baseline because it is a well-known learning algorithm.

Instance-based learning (IBL) stores the training data. When a new input vector arrives, a set of similar related instances is retrieved from memory and their corresponding outputs are used to predict the output for the new input vector [1].

IBL algorithms are derived from the nearest neighbor classifier. It classifies an unknown input vector by choosing the output of the nearest instance in the training set as measured by a distance metric. A generalization of this method is the k-nearest neighbor (k-NN) method, when more than one neighbor is used.

Bayesian Networks (BayesNet) are structured as a combination of a directed acyclic graph of nodes and links, and a set of conditional probability tables [12]. Nodes represent features or classes, while links between nodes represent the relationship between them. An estimation algorithm is used to create the conditional probability. We used the Simple Estimator algorithm, which estimates probabilities directly from the training data.

## 6. Training and estimating processes

Both training and estimating processes are divided into two stages: 1) an individual phase for training/estimating the specific groups to which the user belongs to and the context required for such groups to be created and 2) a shared phase when the system checks if the users required for the creation of a particular group are active or not. This second stage, unlike the individual one, is unique for all group members and the training and estimation processes are common to the group members. The estimation process of this group phase is only performed after the first stage in order to confirm the former estimation process.

### 6.1. Training process

The individual training phase is different for every user in the system. Each user has his own history with the groups which he has belonged to and the specific context that he sensed while he was working in each group. The action of joining a group by a user is the start signal for the collection of training data (Figure 3).

The training data is composed of a set of context vectors sent by the users to the context collector. C[n] is the n-esim context vector sensed by the user at the arrival of a group joining event, and C[m] represents the m-esim context vector sensed upon arrival of a group leaving event.

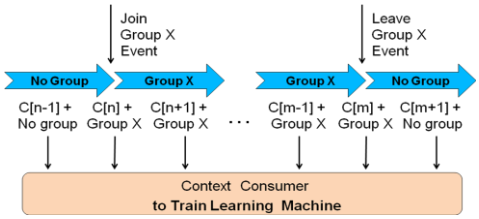


Figure 3. Training process

This data is collected according to the following pattern:

- Group joining event: Upon arrival of this user action, the collected training data consist of the preceding context event C[n-1], associated with non-group state, and the current context event C[n] associated with the group just created (Group X).
- Group operation period: During this phase, the sequence of training data is composed of the context events periodically sensed by the user C[n+1], C[n+2],… –associated with the time period during which the group remains working–.
- Group leaving event: Finally, at the time of arrival of this action –for finishing the group collaboration–, the training data is composed of the last context event C[m] –associated with the group operation state– and the next context event C[m+1] –associated with the following non-group state–.

The shared training phase consists in a record of the group history. It stores the different combinations of active members sensed during the group operation period of the training process.

### 6.2. Estimation process

The individual estimating phase is the first step for the prediction of a group formation. In this stage the system predicts the formation of groups based upon the particular history of each user. Therefore, the groups estimated for different users can be different.

The Intelligent System launches the groups' estimations according to the next pattern:

E[n] denotes the n-esim group formation estimation and C[n] represents the n-esim context vector sensed by the user. This C[n] vector indicates that one or more changes have occurred in the context sensed since the last estimation.

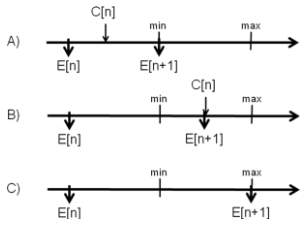


Figure 4. Estimation process

From the last group estimation E[n], a minimum (min) and a maximum (max) time are defined to make the next group estimation E[n+1].

A group estimation is performed depending on when significant contextual changes occur:

- In Figure 4A, if one or more contextual changes, C[n], occur before the minimum time interval of 5 minutes (min), the next estimation, E[n+1], will be done at min time.
- In Figure 4B, if a contextual change event C[n] occurs within a period of from 5 to 15 minutes, the estimation E[n +1] will be done at the time of the arrival of the C[n] event.
- In Figure 4C, if no changes occur, the estimation E[n+1] will be done at the end of the 15 minutes period (max).

The shared estimating phase simply consists in the verification that the Bluetooth devices of the members required for the creation of the group estimated in the previous phase, are active. The number and identity of the required members are determined according to the group history recorded during the shared training phase.

## 7. Experiments and results

This section presents the experimental setting and results of the simulations conducted to assess the automatic recognition of group formations.

### 7.1. Experimental setting

This study has been done at the EPSC campus of UPC, an engineering school designed for collaborative and project based learning models. The classrooms are equipped with tables and chairs that facilitate teamwork. In addition, each student has a laptop equipped with WiFi and Bluetooth cards so they can interact with one or more peers through the learning activities.

We studied the behavior of the students during 15 weeks in order to identify the students' activities and the relevant contextual information needed to represent such activities. We recorded the time of each of the students' actions and some data regarding to the nature of such actions: the place, the list of students involved in a certain group activity, the subject they were attending at that moment, etc. We gathered information to develop a model that describes the students' activities and their characteristics.

The model used to perform our experiments considered a group of 30 students. These students represent a set of all the students in the campus that can collaborate with each other in any different situation. Only 20 of these students follow the same course and the rest are others who can form a group out of the classroom context.

Six different sites were selected as potential locations where the students could collaborate: classroom, library, group study room, cafeteria, vending machines and university grounds. Some of these places are not typical for a learning meeting but are daily meeting places for students where collaboration can occur spontaneously.

It was also identified that a particular student can be part of an average of 6 different groups during an academic semester. Of these groups, 4 are planned for regular work and the other 2 are spontaneous (unplanned) groups formed unexpectedly when the students have an opportunity to collaborate.

The regular class schedule of the considered set of students comprises six daily hours on mornings beginning at 8 o'clock. Moreover, the students have 6 different subjects (which are taught in the same classroom) but only 4 of them require teamwork.

Out of the lectures hours, students meet in different places in other to carry out the tasks requested by their teachers.

### 7.2. Data collection

We use data traces collected from the study of the students' behavior to train and test our system.

This data represent the contextual information of the activities of the selected group of students during a whole week while they are collaborating. The traces contain:

- The week day and time.
- The user Id: the username and Bluetooth MAC address.
- The place where they are located: based on the access points fingerprint [4], our system obtains information about where users are placed (classroom, cafeteria, library, etc).
- The neighbors list: obtained by recording the MAC addresses of the Bluetooth devices sensed by the user. For the collection of the Bluetooth traces, we run a Java API for discovering Bluetooth devices (JSR 82). This API sensed the signals transmitted from Bluetooth USB devices connected to laptops.
- The activity that is being performed (group or non-group identifier).

| **Time stamp** | | **User** | | **Place** | **Neighbors** | |
|---|---|---|---|---|---|---|
| ***Week day*** | ***Time*** | ***Username*** | ***Bluetooth MAC*** | | ***Bluetooth MAC*** | **…** |
| Monday | 9h30 | Anna | 00:FF… | Cafeteria | 00:FF… | … |
| Monday | 9h45 | Anna | 00:FF… | Cafeteria | 00:FF… | … |

Table 1. Contextual input vector

We transform the traces into input vectors for our system (Table 1). These vectors include the contextual information of a given activity from the point of view of one of the students, and the outputs of the system are the estimated group or non-group identifiers associated to such activity. This information is used to train the learning algorithms, which store this training data and use it later to predict the output at the arrival of a new contextual event.

The training data is a set of input vectors and the corresponding outputs (group membership). The items of an input vector are: a numerical value representing the time, a class corresponding to the day of the week, another class for the possible locations where collaboration could take place, and finally, a set of class items corresponding to the number of students that participate in the collaborative activity. These last items are Boolean values reflecting the presence or absence of Bluetooth signal from the students' devices.

The testing data is similar to the training one. It has similar input vectors but it includes all the contextual information that the student senses at any time and not only when the student joins, leaves or is working in a given group. Moreover, we found that the testing data has differences in contextual information, such as the starting time and duration of the group operation period, some sensed Bluetooth MAC addresses, etc.

For our simulations we collected data during two weeks. We gathered a total number of 132 training vectors during the first week. On the other hand, we collected another set of 214 vectors during the second week that was used for testing.

### 7.3. Selection of the algorithm

We performed some experiments using the data gathered during the first week for training the system. The purpose of these experiments was to validate the quality of the proposed learning model and to confirm experimentally that the IBL algorithm is appropriate for our objectives. We used the training data set as input for the learning model created by the machine learning algorithm to confirm that we obtain the expected outputs. It shows the ratio of correctly and incorrectly classified contextual events when the system's input is the same data used to construct the model. Due to the nature of the IBL algorithm and the importance that it gives to the training data when estimating the output for a new input vector, it achieves a recognition rate of 100%. In contrast, BayesNet gets the proper output in 85% of events.

Subsequently, we applied another method of evaluation that uses only some parts of the training data to create the model and the rest are used to test the performance of the system. We selected a method that randomly splits the original data sample into 10 subsamples. Then, 1 subsample is used as validation data for testing the model and the remaining 9 subsamples are used for training. This process is repeated 10 times using each of the subsamples exactly once for validation. Lastly, the results from the 10 testing subsamples are combined to obtain a single estimation. Table 2 shows the results of this validation method. The percentage of success decreases because the learning algorithm has less input information for training, and in addition, the

discarded information might be more relevant than the remaining one.

| | Correct | Incorrect |
|---|---|---|
| BayesNet | 106 (80.3%) | 26 (19.7%) |
| IBL (K=1) | 107 (81.1%) | 25 (18.9%) |

Table 2. Predictions accuracy using different subsamples of the training data

Finally, we used a different data set –gathered during the second week– to make a more realistic evaluation. This new data is used as testing data and it is different from the training data –collected the first week–. It also includes a wider context sensing. Therefore, The training data is used to create the model and the testing data is used to assess the performance of such model when it has to make a prediction based on new contextual information. Once again, the system computes the ratio of correctly estimated contextual events.

| | Correct | Incorrect |
|---|---|---|
| BayesNet | 149 (69.6%) | 65 (30.4%) |
| IBL (K=1) | 208 (97.2%) | 6 (2.8%) |
| IBL (K=2) | 202 (94.4%) | 12 (5.6%) |
| IBL (K=3) | 136 (63.6%) | 78 (36.4%) |

Table 3. Predictions' accuracy using the testing data

Table 3 compares the accuracy in the prediction of group formation for the BayesNet and the IBL algorithms when using a testing data set different from the training one. For the IBL algorithm, we tested several configurations with different values of the K parameter. When using the default configuration (K=1), IBL classifies an unknown input vector by choosing the output of the nearest instance in the training set. In order to use more than one nearest instances for the classification of the input vectors, we also used K=2 and K=3.

Using this evaluation method, we can see the actual performance of the system and the differences between IBL and BayesNet. The results obtained show a clearly superior performance of IBL. It is also noticeable the fact that for higher K values the accuracy of the results decreases. However, both IBL K=1 and K=2 still have a clear superior performance than BayesNet.

The results show that IBL with K=1 has the higher prediction accuracy, however we also chose K=2 because it estimates the output by using the 2 nearest instances. It allows us to detect in which estimations of the IBL algorithm, the degree of confidence is not 100%. This is due to the possibility that the 2 nearest instances can result in different outputs –non-group versus a group identifier or two different group identifiers–. This information can be useful as an indicator of the quality of context information.

### 7.4. Results and findings

We evaluated our model and the quality of the contextual information selected.

| All sensed MACs | Correct | Incorrect |
|---|---|---|
| IBL (K=1) | 208 (97.2%) | 6 (2.8%) |
| IBL (K=2) | 202 (94.4%) | 12 (5.6%) |
| Only known MACs | Correct | Incorrect |
| IBL (K=1) | 145 (67.8%) | 69 (32.2%) |
| IBL (K=2) | 140 (65.4%) | 74 (34.6%) |

Table 4. Assessment of Bluetooth' MAC impact

Table 4 shows the results of the experiments when considering all the MAC addresses of the Bluetooth devices sensed by the user versus the results when the contextual vector only includes the MAC addresses of the students' devices that have previously formed a group with the user. The results prove that it is very important to include in the contextual vector all the MACs sensed by the user and not only the MACs of the known group members. Not including this information would result in a drop of the rate of correctly classified instances from 97.2% to 67.8%.

| | Group | No Group |
|---|---|---|
| Group | **102 (47.7%)** | 8 (3.7%) |
| No Group | 26 (12.1%) | **78 (36.5%)** |

Table 5. Confusion matrix of group predictions

Table 5 shows the confusion matrix in IBL K=1. In this table, 'Group' represents all groups to which the user has belonged whereas 'No Group' represents the non-group activity state. The matrix shows the number of correctly classified events in the diagonal and we can see that in this test, the ratio of incorrectly classified events –items out of the diagonal– was approximately 15.8%. From the matrix, we can deduce that the worst results are obtained because sometimes IBL confuses the 'No group' state with the 'Group' ones. However, some of the incorrectly classified items are not a

major problem because although groups are not properly detected, they do not confuse or disturb the students while they are working.

In the previous experiments, we assumed our model inputs were accurate. However, in practice, inputs might be calculated through error-prone methods or the real-world context of the user might be considerably different from the context used for training the machine learning component. To address this, we tested our model's robustness by introducing noise into the contextual vector that we use as input data for our system.

| | Correct | Incorrect | |
|---|---|---|---|
| Test data | | Total errors | Time errors |
| IBL (K=1) | 208 (97.2%) | 6 (2.8%) | 4 |
| IBL (K=2) | 202 (94.4%) | 12 (5.6%) | 10 |
| Test data with time errors | | Total errors | Time errors |
| IBL (K=1) | 183 (85.5%) | 31 (14.5%) | 19 |
| IBL (K=2) | 180 (84.1%) | 34 (15.9%) | 22 |

Table 6. System's robustness to time errors

We define an error as a highly significant change or a missed value in the contextual vector. In order to assess the system's robustness, we tested its sensitivity to each of the contextual vector items. Table 6 shows the effect in the accuracy of the results by introducing errors in the "time" item of the vector. This table compares the results obtained using the original testing data with those obtained by introducing errors in time on the order of hours. It specifies the number of incorrectly classified instances that are due to such errors in time.

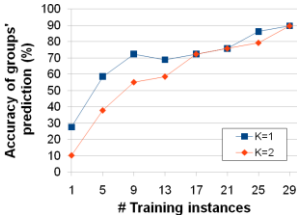


Figure 5. Estimations accuracy of spontaneous group formation versus number of training instances

In addition to the previous experiments for the evaluation of the accuracy of the estimations, we also calculated the evolution of the estimations accuracy with the number of training instances used to construct the learning model. Figure 5 shows the results when testing the accuracy in the estimation of the formation of a spontaneous group for IBL K=1 and K=2. We can see how the ratio of correctly classified instances increases by incrementing the number of training instances. In K=1 we obtained a 87% accuracy ratio when we have 21 training instances.

Finally, we performed some tests in order to identify the effect of the number of groups used for training in the estimations accuracy of a specific group. Therefore, we intended to evaluate if the rest of the groups in the user history could be a source of interferences for the estimations. This could be interesting to know if there is a maximum limit for the system to be able to distinguish two different groups. Figure 6 shows the results of this experiment for both, planned and spontaneous groups. As expected, the accuracy (100%) of planned groups seems to be independent of the number of total groups –at least for 6 or less groups–. However, the accuracy of spontaneous groups estimations decreases up to 87% for 3 or more interfering groups.

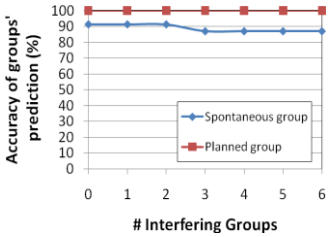


Figure 6. Estimations accuracy versus the number of interfering groups of the training data

By performing all our experiments, we observed the importance of the appropriate selection of the time intervals to collect the training data. In addition, the necessity of including in the training data some events with the contextual information when no groups were established was also noticeable. If such events were not included, the ratio of correctly classified events suffered a dramatic decrease.

On the other hand, we observed a slight improvement when multiple events reinforcing the fact of the existence of a group were introduced.

## 8. Effect on the user: interruptions

It should be noted that an erroneous estimation does not necessarily imply an interruption in the

attention of the student. For example, two consecutive erroneous or uncertain estimations are not two interruptions but actually they are just one. The first one interrupts the student and changes its focus away from the main activity towards the change of context, tools, group, etc. decided by the system, but the second one does not cause any interruption because the student has already left his main activity.

For that reason we defined a burst as a sequence of erroneous estimations. The end of the errors burst, i.e., the return to the correct behavior, is identified with two correct consecutive estimations. From the log data we identified the bursts of erroneous estimations. The impact on the activity of the student, measured in number of interruptions. When no interruptions management is used, each student has an average of 26 interruptions during a week. We think that it is a very large value and can complicate the execution of the groups and student activities. By introducing management of error bursts, these interruptions go down to an average of 4. We believe this new value is acceptable and has little impact in the student activities.

## 9. A case study

The system described in this paper can be easily integrated into some existing collaborative learning support environments such as Sugar [16].

Sugar is a software platform that provides a collaboration framework for learning. Collaboration is implemented with the telepathy library (telepathy Gabble), a D-Bus/XMPP framework for Instant Messaging protocols.

Figure 7 shows how our system can be integrated into Sugar. At the user side, both Sugar and the Context Provider are executed. The Context provider sends all the information related to one user to the intelligent system through the same XMPP server that Sugar uses.

When our Intelligent System predicts if the user is going to collaborate with a particular group, it sets up the group working environment. In the case of the Sugar framework, the system starts an activity, for example file sharing, and it shares this activity with the other users who are part of the group by sending invitations (telepathy library) to them through the XMPP server. Only those users who receive an invitation can share the activity.

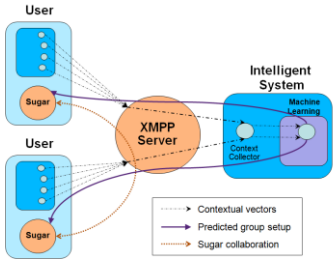


Figure 7. Intelligent System integration with Sugar

The screenshot in Figure 8, shows users working together on a collaborative activity. Three users are working in a group detected automatically by the Intelligent System and are learning Mathematics through a Sugar memorization activity.

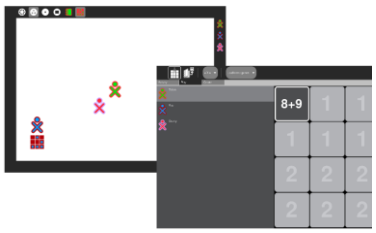


Figure 8. Sugar collaboration screenshot

## 10. Related work

Although there are many tools for providing some degree of automation and support for activities within a group, we have not found specific tools for providing the applications with automatic group awareness based on environmental contextual data.

In [6], the authors present an approach to the classroom context by identifying the students' activity. The main goal is to acquire physical interaction identifying and obtaining context services. This proposal facilitates a service to a single user taking into account his contextual information but it does not provide service adaptation. Our system proposes a complementary approach. We intend to provide services to a group of students instead of just one student. In addition, we adapt the service provided to the user dynamically, learning from new situations.

MCI-Supporter [2] is an application supporting collaborative learning methods in the classroom. It was conceived by first analyzing the best known collaborative learning practices trying to find out which are the real needs for mobility and face-to-face. However, the groups have to be manually established by the teacher. In [8], the authors discuss on contextual information about groups. They focus on workspace and social awareness and they even comment on team formation support: closed and open teams joined and left manually and dynamic teams formed automatically by the system based on context and meta-information. Our approach differs from the presented above mainly because we predict users' interactions taking into account the users past actions and our system does not require the users' intervention. Related works are less flexible, use only static information, do not learn and self-adapt depending on the circumstances, and they need the users to perform some extra actions.

## 11. Conclusion

In this paper, we presented the motivation and the problem of automating the incorporation of group awareness information into CSCL applications without shifting this burden to group participants or overloading the teacher.

We proposed an Intelligent System which, based on contextual information, is able to automatically estimate group membership. We evaluated its utility in terms of the rate of group recognition success. Simulations showed that context-information can be effectively used as a basis for a middleware for automatic and dynamic group management. Therefore, we conclude that inferring group membership is technically viable and can be used in real-world settings.

Our results could facilitate the construction of applications that effectively assist group members in automatically sharing, communicating and coordinating as they move and reorganize in synchronous and collocated collaborative learning activities.

## Acknowledgement

This work has been partially supported by the Spanish MEC project P2PGrid TIN2007-68050-C03-01

## References

[1] Aha, D. Kibler, D. Albert, M.: Instance-based learning algorithms, Machine Learning, Kluwer Academic Publishers, 37-66, 1991.

[2] Baloian, N. Zurita G.: MC-Supporter: Flexible Mobile Computing Supporting Learning though Social Interactions, Journal of Universal Computer Science, Vol. 15, No. 9, 2009

[3] Bolchini, C. Curino, C.A. Quintarelli, E. Schreiber, F.A. Tanca, L.: A data-oriented survey of context models. SIGMOD Rec. 36, 4, 19-26, 2007

[4] Bolliger, P.: Redpin - adaptive, zero-configuration indoor localization through user collaboration Proc. Workshop on Mobile Entity Localization and Tracking in GPS-Less Environments, 2008

[5] Borges, M.R.S. Brézillon, P. Pino, J.A. Pomerol J. C.: Groupware System Design and the Context Concept, Proc. Computer Supported Cooperative Work in Design 2004

[6] Bravo, J. Hervás, R. Chavira, G.: Ubiquitous Computing in the Classroom: An approach thorough Identification Process, Journal of Universal Computer Science, Vol. 11, No 9, 2005

[7] Derntl, M. Hummel, K.A.: Modeling Context-Aware e-Learning Scenarios. Proc. Pervasive Computing and Communications Workshops, IEEE, 2005.

[8] Ferscha, A. Holzmann, C. Oppl S.: Team Awareness in Personalized Learning Environments, Proc. Mobile Learning, 2004.

[9] Hall, M. Frank, E. Holmes, G. Pfahringer, B. Reutemann, P. Witten, I.H.: The WEKA Data Mining Software: An Update; SIGKDD Explorations, Vol. 11, No 1, 2009

[10] Hwang, G.J. Tsai, C.C. Yang, S.J.H.: Criteria, Strategies and Research Issues of Context-Aware Ubiquitous Learning. Educational Technology & Society, Vol. 11 No 2, 2008

[11] Johnson, D.W. Johnson, R.T.: An overview of cooperative learning, J. Thousand, A. Villa and A. Nevin (Eds), Creativity and Collaborative Learning; Brookes Press, Baltimore, 1994

[12] Langley, P. John, G.H.: Estimating continuous distributions in bayesian classifiers. Proc. Uncertainty in Artificial Intelligence, 1995.

[13] Messeguer, R. Damián-Reyes, P. Favela, J. Navarro, L.: Context awareness and uncertainty in collocated collaborative systems, Proc. CRIWG, Vol. 5411 of LNCS. Springer, 41-56, 2008.

[14] Shadbolt, N. Ambient Intelligence. IEEE Intelligent Systems. Vol. 18 No. 2, 2-3, 2003.

[15] Weiser, M., The Computer for the 21st Century. Sci American. Vol. 265 No. 3, 94 -104, 1991.

[16] http://www.sugarlabs.org