Project Deliverable Report

Deliverable D3.3 – Definition and Implementation of Context Awareness v2

Work Package 3
Task 3.5
Date of delivery Contractual: 30-09-2009    Actual: 10-11-2009
Code name Version: 1.0    Draft □ Final ☒
Type of deliverable REPORT
Security PU
(distribution level)
Contributors UCY, HILDESHEIM
Authors (Partner) George A. Sielis (UCY), Christos Mettouris (UCY), Aimilia Tzanavari (UCY), George A. Papadopoulos (UCY)
Contact Person George Sielis, Christos Mettouris
WP/Task responsible George Sielis
EC Project Officer Mr. Christian Wilk
Abstract (for dissemination) This Deliverable describes the Context Awareness component v2 of the idSpace platform. It refers to the technologies used for its implementation, its structure and the methods followed for the computation of the included recommendation packages. Moreover, the meta-model of the current implementation is presented, as well as the ideal meta-model for Context Awareness. In addition, a description of how the component was preliminary tested using dummy data is presented.

Keywords List Context Awareness component, Conceptual model, Recommendations, Context Factors

idSpace Project Coordination at: Open University of the Netherlands
Valkenburgerweg 177, 6419 AT Heerlen, The Netherlands
Tel: +31 45 5762624– Fax: +31 45 5762800
# TABLE OF CONTENTS

List of Figures ............................................................................................................. 4

List of Tables ............................................................................................................... 5

1 INTRODUCTION ........................................................................................................... 6
   1.1 Background ........................................................................................................... 6
   1.2 Relation to Other Deliverables ............................................................................ 6
   1.3 Structure of this Deliverable ............................................................................... 7

2 IMPLEMENTATION TECHNOLOGIES ................................................................. 8
   2.1 Ontopia Knowledge Suite (OKS) ........................................................................... 8
   2.2 Liferay - Portlets ................................................................................................. 8
   2.3 Topic Maps ........................................................................................................... 8

3 CONTEXT AWARENESS META-MODEL, COMPONENT STRUCTURE AND WORK PROCESS ........................................................................................................ 10
   3.1 Context Awareness Meta-model ......................................................................... 10
   3.2 Context Awareness Component Structure ......................................................... 14
   3.3 Context Awareness Component Work Process .................................................. 15

4 COMPONENT IMPLEMENTATION .............................................................................. 19
   4.1 Overview .............................................................................................................. 19
   4.2 Packages ............................................................................................................. 19
   4.2.1 User Input and Factor Evaluation .................................................................... 19
   4.2.2 Recommend Solutions ..................................................................................... 20
   4.2.3 Recommend Users ........................................................................................... 21
   4.2.4 Recommend Ideas ........................................................................................... 23
   4.2.5 Recommend Resources .................................................................................... 26
   4.2.6 Recommend Pedagogical Pattern .................................................................... 28

5 COMPONENT PRELIMINARY TESTING ................................................................. 31

6 CONCLUSIONS AND FUTURE WORK .................................................................... 34

References ..................................................................................................................... 35

APPENDIX A: Context Awareness Component Code .................................................. 36
   A.1 Recommend pedagogical patterns ................................................................. 36
A.2  Code samples from the recommendation packages which use factors and relevance function ................................................................. 40
A.2.1 Sample of the Tolog queries used ................................................................. 40
A.2.2 Sample of the Relevance Function Implementation .................................. 41
A.2.3 Sample of the Scoring Process .................................................................. 41
LIST OF FIGURES

Figure 1: Merged ontology meta-model schema ............................................................ 11
Figure 2: Latest version of CA ontology meta-model schema ....................................... 12
Figure 3: Optimal ontology meta-model schema .......................................................... 13
Figure 4: Context Awareness component architecture.................................................. 14
Figure 5: Sample interface design for CA component .................................................... 17
Figure 6: Sample interface design for CA component - Recommendation results ......... 18
Figure 7: Main processes diagram. The red arrows indicate where the recommendations of solutions will be provided...................................................................................... 21
Figure 8: Main processes diagram. The red arrows indicate where the recommendations of users will be provided.......................................................................................... 23
Figure 9: Main processes diagram. The red arrows indicate where the recommendations of ideas will be provided.............................................................................................. 26
Figure 10: Main processes diagram. The red arrows indicate where the recommendations of resources will be provided................................................................. 28
Figure 11: Main processes diagram. The red arrow indicates where the recommendation of the pedagogical pattern will be provided................................................................. 29
Figure 12: The user inputs data to be recommended of solutions....................................... 33
Figure 13: The CA component provides the solution recommendations ranked.......... 33
LIST OF TABLES

Table 1: Pedagogical Pattern Scoring Example.................................................................30
1 INTRODUCTION

1.1 Background

This deliverable, which is the third one to be produced within Work Package 3, aims to describe the Context Awareness (CA) component version 2. The CA component version 1 was described in the second deliverable of WP3. The objective of the second deliverable was to introduce the architecture and the methodology of implementation of the CA component, based on the principles of context awareness, as these were analyzed in deliverable D3.1. Deliverable D3.2 presented the functional architecture of various context awareness modules, as well as the existing methods of reasoning, together with several technologies for context reasoning. This analysis, followed by the integration of the conceptual meta-model of the CA component, delimited the methodology, the technology and the architecture, which should be used for the development of the first version of the component and thereupon its evolution to the second version, described in the current deliverable.

In this deliverable, the current version of the CA meta-model is described, as well as the modifications in contrast to the previous version (D3.2). Moreover, the integrated CA prototype version 2 will be described through the various recommendation packages which constitute the component.

1.2 Relation to Other Deliverables

The deliverable is related to the following deliverables:

- **D1.2 Templates and prototypical implementation of informal idSpace pedagogical strategies for creativity v1.** The D1.2 deliverable introduced the idea of flow patterns and how the flow patterns can positively influence learning processes involved in collaborative creation of new products. Moreover, D1.2 introduced the factors that need to be taken into account when applying such a pattern in the context of idSpace. These factors are used by the CA component for generating recommendations of Pedagogical Patterns.

- **D2.2 Semantic meta-model, integration and transformations v1.** The D2.2 document introduced a topic map based meta-model, conceptualizing creativity techniques and the creativity process. In this deliverable, this meta-model was merged with the meta-model for Context Awareness, creating thereby a new unified topic maps ontology. This ontology is used for the creation of recommendations considering the new relations and entities as context factors.

- **D4.3 Design Document v2.** The D4.3 deliverable describes the functional design of the idSpace platform, which will be built in D4.4. This description is based on processes, functionality, usability, accessibility, information flow and architecture. From the description of the aforementioned, it becomes clear where in the process the CA component needs to be triggered and what type of recommendation must be provided in the various stages of the platform’s processes.
D5.2 Report of evaluation results, v1. In deliverable D5.2 the evaluation plan and the procedures for measuring the characteristics of the platform are defined. One of the characteristics of the platform that is planned to be evaluated is the CA component. The CA prototype v2 presented in this document, is taking into account the user requirements which were defined in D5.1. The evaluation of the CA component depends on the correctness and completeness of the component’s results, based on the user requirements that were defined in deliverable D5.2.

1.3 Structure of this Deliverable

The deliverable is structured as follows. Chapter 2 describes the technologies which were used for the implementation of the CA prototype v2. Chapter 3 discusses the steps followed to reach the current version of the ontology scheme, as well as how this scheme could be extended in order to obtain the optimal ontology scheme. Moreover, it presents the structure of the CA component and its work process. In Chapter 4, a description is given of the recommendation packages developed within the component, their context factors and the adaptation method used. Following this, Chapter 5 describes the testing procedure, as well as the data used for this purpose, while chapter 6 summarizes the overall document and briefly describes the future work planned.
2 IMPLEMENTATION TECHNOLOGIES

In deliverable D3.2, in which the 1st version of the CA prototype was presented, the programming languages and the methodology used were different from those used in the current version of the prototype. In deliverable D3.2, the version of the CA component was designed based on the technologies introduced by Microcosmos, the development platform at that time.

However, the change of the development platform from Microcosmos to Liferay, as mentioned and justified in the Design Document in D4.3, affected the development of the CA component regarding the programming language to be used. The PHP scripting language and the XML mark-up language that were initially used for the development of the prototype version 1, were replaced by the Java programming language and the Tolog query engine, which runs within the Ontopia Knowledge Suite (OKS, 2009). The architecture of the CA component, as well as the reasoning method introduced in D3.2 remained the same.

Below, the aforementioned technologies are described in more detail.

2.1 Ontopia Knowledge Suite (OKS)

The Ontopia Knowledge Suite (OKS, 2009) is a complete set of tools for building, maintaining and deploying topic map-based applications. It consists of the Ontopia Topic Map Engine, the Ontopia Navigation Framework, the Web Editor Framework, the Full-text Search Integration, RDBMS Backend Connector and the Topic Map Query Engine and Schema Tools.

The Navigator Framework is an XML-based scripting language optimised for topic map application development. Using this language, we can easily collect information from the topic map using the tolog query language, and output the results in HTML and other formats. It is based on J2EE, using the Java Servlets and Java Server Pages (JSP) technologies. It is built on the Ontopia Topic Map Engine, and consists of a set of JSP tag libraries, and a Java API.

2.2 Liferay - Portlets

Liferay Portal (Liferay, 2009) is an open source enterprise portal solution using the Java and Web 2.0 technologies that supports a wide range of services. One of the advantages of Liferay is its flexibility in attaching tools developed as separate packages in the form of portlets. Therefore, the CA prototype version 2 was developed separately from the platform package using the Java programming language and built over the OKS which includes the Tolog query engine and the Navigation Framework for querying topic maps.

2.3 Topic Maps

Topic Maps technology is part of the ISO standards (ISO 13250) of the semantic web technologies. They are used in semantic web applications for finding and exchanging information using topics. Topic Maps are designed for the enhancement of navigation
and information retrieval. This is achieved by the addition of semantics into the resources and their representation as context datasets (Hatzigaidas et al., 2004). For further information about Topic Maps technology, the reader can refer to D3.2.
3 CONTEXT AWARENESS META-MODEL, COMPONENT STRUCTURE AND WORK PROCESS

3.1 Context Awareness Meta-model

The CA meta-model was firstly introduced in deliverable D3.1. The same document included the definitions of the contextual elements, which were part of the CA meta-model. The designed ontology was developed in the second version of the CA component, and presented in deliverable D3.2 as part of the CA component architecture.

During the development of the CA component and the composition of the queries for the retrieval of the context factors, which affect the computation of the CA component’s recommendations, the need to extend the first version of the ontology became obvious. The accuracy of the recommendations depends on the number of factors taken into account during their computation. Therefore, the addition of more factors being able to influence a recommendation was indeed necessary.

The refinement of the CA ontology was initiated by the merge of the CA ontology provided in D3.1 with the ontology provided by WP2 in D2.2. The merge was done for two reasons:

- The idSpace platform should use only one unified meta-model schema.
- The CA component’s need to use as many context factors as possible for the computation of the recommendations.

The merged meta-model schema is shown in Figure 1. The need to add more context factors for the computation of more accurate recommendation results, led to the addition of more entities in the new merged ontology, such as “Keywords”, “Domains”, “Competence” and “User Role” (see Figure 2):

- “Keywords” is an entity used to semantically describe and characterize other entities. For example, a user may be characterized as “programmer” or “doctor”, while an idea may be characterized as “SOS” or “Theoritical”.
- “Domains” is an entity used to provide the field(s) to which the entity belongs. For example, a user that holds a PhD in Human Computer Interaction may have “HCI” as his domain.
- “Competence” is an entity used to indicate a user’s competency in each of the domains she belongs in. The relation between “competence” and “domains” is one-to-many. There exist only four competency values: expert, proficient, competent and novice and each of these values may be associated, on one hand to any domain, and on the other hand to a user, denoting in this way the competence of that user with respect to the domain. This means that for each domain that a user belongs to, a competence value should be declared that shows the competence of the user regarding that domain. As a result, a user may be associated with several domains with the same competence value, for example “expert”.
“User Role” is the entity that specifies the role of each user in the platform. It indicates the privileges the user has in the platform and what actions a user may perform within a creativity project. For the time being, there are two “User Role” properties defined for the idSpace platform, the “Moderator” and the “Learner”.

The CA component currently offers 5 types of recommendations, including recommendations of ideas, resources, users, solutions and pedagogical patterns. The integrated types are designed and developed based on the latest version of the ontology, as it is depicted in Figure 2.

In Figure 3, the ideal CA ontology schema is presented. This ontology includes all entities that should ideally be used for the process of generating accurate recommendations. However, this exact version of the ontology is difficult to be fully utilized by the CA component at the present, due to the project’s time constraints. The difficulty of such an integration lies in the high complexity of the resulting implementation to be done, as well as the need to use AI (Artificial Intelligence) techniques for the inclusion of all the factors in the computation of recommendations.

Figure 1: Merged ontology meta-model schema
Figure 2: Latest version of CA ontology meta-model schema
Figure 3: Optimal ontology meta-model schema
3.2 Context Awareness Component Structure

The Context Awareness component was designed and developed based on the architecture presented in D3.2.

The component is capable of producing recommendations based on the context of the idSpace platform. Each recommendation is developed as an individual software package, which may be triggered by a central component, the Context Manager (Figure 4). Each software package contains a Wrapper and an Adaptation Manager layer, which are described in the next paragraphs.

Figure 4: Context Awareness component architecture

The CA component contains five recommendation packages, which are described in detail in chapter 4:

1. Recommendation of Ideas
2. Recommendation of Resources
3. Recommendation of Solutions (or Problems)
4. Recommendation of Users
5. Recommendation of Pedagogical Patterns

The Context Manager takes into account the actions made by the user, her input and the stage in the main processes diagram where the component is triggered at, to determine the type of the recommendations and therefore to activate the corresponding Wrapper responsible for that recommendation type. The Wrapper uses all available input to query the topic map and capture all the related context data. The resulting recommendations are passed on to the adaptation layer, consisting of the Adaptation Manager, where a
Relevance Function is used to rank the recommendations. The ranked recommendations are then presented to the user.

The reasoning method used relies on the semantics. It is based, on one hand, on the keywords and domains given as input by the user upon requesting recommendations and on the other hand, on the keywords and domains that describe the current idSpace project. The data describing the project were provided by the moderator during the “project creation” stage and are considered to characterize the entire project. By using the keywords and domains dataset, the Wrapper queries the topic map to find all entities of the requested type that are associated with these specific keywords and domain entities in the ontology. The wrapped data are then forwarded to the Adaptation Manager, where the context information is retrieved, based on the predefined context factors, and finally the overall wrapped data are filtered. The filtered context data are parsed according to the adaptation method and are then presented to the end user as ranked recommendations.

For the adaptation, the Relevance Function (Kakousis et al., 2008) method is used. The Relevance Function is an equation of contextual factors and relevance weights. An explanation of how the Relevance Function, the context factors and the weights are used follows in paragraph 4.2.

### 3.3 Context Awareness Component Work Process

The CA component may provide recommendations on demand, or automatically. In the first case, the user specifies the type of recommendations s/he would like to receive and gives the necessary input that the CA component needs to provide the recommendations. In the second case, the CA component automatically generates recommendations based on (a) the stage in which the project is at that time and (b) on previous user input. For the CA component to be able to provide recommendations automatically, it is imperative that the idSpace platform invokes the component and at the same time provides the necessary data through the topic map.

Whether a particular type of recommendation should be offered to users on demand and/or automatically is an issue open for discussion. We argue that every recommendation type should be provided automatically so that the platform actively guides the user through the workflow, while at the same time giving the user the “on demand” option if s/he wishes to provide specific requirements to get more targeted recommendations. The CA component is currently implemented to support both cases.

When a new project is created, the moderator has to define/enter keywords and domain(s). The keywords characterize the problem to be solved and the domains delimit it. Next, these data are saved in the topic map so that the CA component can use them as predefined input to compute recommendations. When a user, whether a moderator or not, needs to be provided with any type of recommendations, the CA component will include in it’s input the moderator’s predefined data, along with any additional keywords and domains the user may want to add. In the case of automatic recommenda-
tions, the CA component will provide recommendations based entirely on the initial input of the moderator.

In the case where the CA component provides recommendations on demand, it presents the user with the available types of recommendation as options, depending on the stage in which the project is at that time. The user then chooses the type of recommendations s/he would like to receive and provides the appropriate input.

For example, let’s assume that the CA component is triggered during the “create project” stage. Then, the component’s interface will provide the options “Recommend Users”, “Recommend Solutions”, “Recommend Pedagogical Patterns” and “Recommend Resources”, because at that stage, these recommendation types are available for the user to choose from. However, if the component is triggered during another stage, for example the “ideation session”, the options provided will differ, since the needs of the user at that stage of the project are different. More particularly, the recommendation options at the “ideation session” stage would include the “Recommendation of Ideas”, the “Recommendation of Users” and the “Recommendation of Resources”. The various stages in which each recommendation type must be provided are referred to in Chapter 4 and depicted in Figures 7 to 11.

In addition to the above, the CA component gives the user the possibility to receive even more tailored recommendations, according to his/her own preferences, by specifying which recommendation type factors are considered more important and to what extent. This is done by applying weights to factors. One weight applies to each factor, determining the factor’s relevance, or in other words significance in the recommendation. The user can choose among three possible values: “high relevance”, “medium relevance” and “low relevance”. Each value corresponds to the level of significance of each factor to the final recommendation and affects the final ranking of that recommendation. More detailed review of factors and weights is given in the next chapter.

In the case of provision of automatic recommendations, these are computed and presented to the user without requiring any direct user input. These are dependent on the stage the project is in at the time, as well as stored information for the project, the user himself, etc. Again, more information on the implementation details behind these recommendations is given in the chapter that follows.

A first design approach of how the CA component should appear in the platform is depicted in Figures 5 and 6. In these designs, the recommendation packages are presented as tabs. Each tab contains forms to receive input parameters from the user. These input data will be used by the CA component to formulate the recommendations as a result (Figure 6). As shown in Figure 5, the user has the option to define the relevance value for each factor, as well as the keywords and domains to be used by the component. A tab will be activated (visible) or deactivated (hidden) depending on the stage of the idSpace platform’s main processes workflow. The stages of the platform’s main processes workflow and the recommendation types available in each stage are described in more detail in chapter 4.
Figure 5: Sample interface design for CA component
Figure 6: Sample interface design for CA component - Recommendation results
4 COMPONENT IMPLEMENTATION

4.1 Overview

Each type of recommendation provided by the CA component has been implemented as a separate software package. Each package receives all necessary input from the topic map and the user, computes the recommendations and presents them to the user. In this chapter, we will describe these packages in detail.

4.2 Packages

The packages, although providing recommendations of different type each, were designed based on the same structure, the same architecture and the same reasoning and adaptation method. Each package has the following characteristics:

- Its recommendations are based on a set of factors.
- Each factor i is evaluated using a Fitness Function $f_i(x)$. The Fitness Function shows how relevant a recommendation is in respect to a factor.
- Weights are introduced to allow end users to specify the importance of each factor, according to their personal preferences.
- To each factor i corresponds one weight $W_i$
- Relevance Function $R(x)$: compute the relevance score of each recommendation (rank of appearance)

$$ R(x) = \sum \frac{W_i * f_i(x)}{\sum(W_i)} \cdot N $$

where N: number of Factors

4.2.1 User Input and Factor Evaluation

The CA component provides recommendations of various types that are related to the problem the moderator is trying to solve in the current idSpace session. At the start of such a session, the Moderator provides the IdSpace system with keywords related to the project she is initializing and domains that the project belongs in or is associated with.

Moreover, the CA component allows the user to provide more keywords in order for the recommendations to be more accurate regarding her preferences. These keywords, along with the aforementioned keywords and domains provided by the moderator, constitute the input that the CA component uses to calculate the recommendations. They are used, not only as discrete context factors for each recommendation type, but also as metrics for other context factors.
As said, each recommendation type is based on a set of factors. These factors are ontology entity types, for example “keywords”, “problems”, “domains”, etc. Depending on the recommendation type, the CA component examines all instances within the appropriate entity types and measures their relevance to the given problem to be solved, in order to opine which are the most relevant to be recommended. The relevance is being measured by using ontology associations. If an instance of such an entity type is highly associated with the current idSpace session, then it is highly recommended by the component. The instance and the current idSpace session are highly associated if they have a number of common keywords and domains. The more common data they have, the more associated they are considered to be. Thus, the procedure of inputting keywords and domains to describe a problem or any other contextual entity in an idSpace session is critical and should be done with great caution.

Following, the different recommendation types are further explained:

### 4.2.2 Recommend Solutions

The CA component provides recommendations of relevant solutions\(^1\). These solutions constitute the solution of projects who confronted similar problems to the one the moderator is trying to solve in the current idSpace session. The recommended solutions will indicate to the Moderator how related projects were solved in the past. Some facts concerning the recommendation of solutions can be seen below.

**Necessary Input:**
- Keywords provided by Moderator to describe the problem (“create project” stage)
- Domains provided by Moderator to define the problem’s area (“create project” stage)

**Optional Input:**
- Additional Keywords provided by user (during the recommendation procedure)
- User defined weights

**Factors:**
- Related Keywords
- Related Domains
- Related Problem Statements
  - Find problem statements relevant to the input data
  - Use the relevant problem statements to retrieve related solutions

---

\(^1\) A solution is the outcome of a solved project in the idSpace platform.
• Related Problems
  o Find problems relevant to the input data
  o Use the relevant problems to retrieve related solutions

Based on the main process model, the recommendation of solutions will be used in the stages indicated in Figure 7.

![Main processes diagram]('https://example.com/main_processes_diagram.png')

Figure 7: Main processes diagram. The red arrows indicate where the recommendations of solutions will be provided

The recommendations of solutions will be provided in the phase of starting a new project. At this point, the Moderator is thinking of how to correctly state a problem in the IdSpace platform, in order to use the platform to find a solution. Thus, in the stages “Create a Project” and “Create a Problem statement”, the system will provide relevant solutions.

4.2.3 Recommend Users

The CA component provides recommendations of users that are most relevant to the particular idSpace session (or current project being solved), in terms of competence, expertise in various domains, as well as relevance based on the user’s keywords. User recommendations will help the moderator in forming a proper user group, which will have the best potentials to solve the session’s problem. The purpose of user recommendations is to recommend, not only the users who are experts in the project’s thematic area, but also users that may have experience in related fields or may have
solved relevant problems in the past. Some facts concerning the recommendation of users can be seen below.

**Necessary Input:**
- Keywords provided by Moderator to describe the problem ("create project" stage)
- Domains provided by Moderator to define the problem’s area ("create project" stage)

**Optional Input:**
- Additional Keywords provided by user (during the recommendation procedure)
- User defined weights

**Factors:**
- Related Keywords
- Related Domains
- User’s Previous Work
  - Find solutions relevant to the input data
  - Retrieve the user group that participated in the project that resulted each solution
  - Retrieve the users of each user group
  - Each user in the group receives the score of the corresponding recommended solution
- User’s Competence
  - Retrieve related users based on their expertise, as it was stated in their user profile
- User’s User Role
  - For the user group, select only moderators, learners or both, according to moderators needs (A user role may be either a “moderator” or a “learner”)
- User’s Social Role
  - Find the users that belong in social groups related to the project domains
  - Rank the users based on their expertise in these social groups

---

2 The social role factor is taking into consideration other people’s opinion about a user, and uses these opinions to rank the user. This factor could be utilized in a platform where forums and blogs exist, and a user is been evaluated and scored by other users for her expertise in a certain field. Since in the idSpace platform forums or blogs are not yet implemented, the social role factor is not going to be activated in this version of the CA component, even though it has been implemented.
Based on the main process model, the recommendation of users will be used in the stages indicated in Figure 8.

User recommendations will initially be needed at the stage of composing a new group of users. At this point, the moderator wants to add to her team the most expert users in certain domains, the most experienced as well as the most competent ones. The system helps by recommending to him the ranked users, those with higher relevance shown first, according to the previously stated factors.

Recommendation of users will also be needed in the “create common ground” stage. During this procedure, the moderator may eventually decide that some of the selected users in her group were not a good choice, so a new set of user recommendations would be very helpful. To assist the system recommend new users, the moderator should redefine the set of keywords and domains she inputted, or add more. This is an example of recommendations on demand.

Last but not least, user recommendations would be beneficial when a user asks for help or guidance during the ideation procedure. The system should then be able to recommend other users that are experts in a certain field or have experience in a certain technique or procedure. Thus, user recommendations are also important in the stages of “individually generate ideas” and “collaboratively generate ideas”.

4.2.4 Recommend Ideas

The CA component provides recommendations regarding relevant ideas. The purpose of these recommendations is to inspire the user during the “ideation session” stage and provide him with motives to become more creative.
There are two modes in which this recommendation functions.

- The first is the default mode, where the component receives as input the moderator’s keywords and domains stated at the “create project” stage, as well as any additional user keywords (as described in the aforementioned recommendation types). Based on this input, the CA component computes the recommendations and provides them in order of relevance.

- The second mode uses the concept of the highlighted idea. The highlighted idea is the particular idea the user is working on at the time. It is the idea he is trying to further explore, either by expanding its textual content or by explicitly attaching more ideas to it. We argue that, in some cases, the context of this idea (the keywords it may be associated with and the domains it may belong to) may help the CA component provide more relevant recommendations, and hence help the user become more creative. During the “ideation” stage, when a user wants to further exploit a certain idea, the idSpace platform should trigger the CA component properly to function in this mode. As a result, the CA component uses as input, besides the regular data, all the keywords and domains associated with the highlighted idea. Thus, the resulting recommendations are closely related to this idea as well.

Some facts concerning the recommendation of ideas can be seen below.

**Necessary Input:**

- Keywords provided by Moderator to describe the problem (“create project” stage)
- Domains provided by Moderator to define the problem’s area (“create project” stage)

**Optional Input:**

- Additional Keywords provided by user (during the recommendation procedure)
- User defined weights
- A highlighted idea (“ideation” stage)

**Context factors:**

- Related Keywords
- Related Domains
- Related Problems
  - Find problems relevant to the input data
  - Use the relevant problems to retrieve related ideas
• Related Solutions
  o Find solutions relevant to the input data
  o Use the relevant solutions to retrieve related ideas

• Related Resources
  o Find resources relevant to the input data
  o Use the relevant resources to retrieve related ideas

• User’s Competence
  o Find the active user’s domains of competence
  o Use these domains to retrieve related ideas

• User’s Expertise
  o Based on active user’s Social Role, find the domains she has a level of expertise
  o Use these domains to retrieve related ideas

Based on the main process model the recommendation of ideas will be used in the stages indicated in Figure 9.

3 The user working in the current ideation session
4 The user’s expertise factor is taking into consideration the Social Role of the user, which, as previously stated, is not going to be activated in the current version of the CA component. Consequently, the expertise factor will not be activated in this version of the component as well, though it has been implemented.
Figure 9: Main processes diagram. The red arrows indicate where the recommendations of ideas will be provided

4.2.5 Recommend Resources

The CA component supports the recommendation of relevant resources. These recommendations will guide the user to creating new ideas during the “ideation” stage.

A resource might be a web page, a document, a picture etc. that users used to support their ideas. The recommendation of resources is a helpful tool for the user, because it may initiate her thinking towards directions she would not normally explore. Moreover, this kind of recommendations could stimulate one’s imagination, thus, making him or her more creative.

More particular, a resource may be associated with one or more ideas. By recommending a resource to a user, the component in practice is recommending many new concepts and ideas to him, derived from relevant past sessions. For instance, a resource may be a journal paper in pdf format, associated with many ideas. These ideas may be irrelevant to each other, meaning that no implicit association exist among them. In this case, the paper constitutes the only association between them. When a user has received this paper as a resource recommendation, she has also the opportunity to explore the ideas that are associated to it, and furthermore any resources attached to them and so on. Thus, the user through a recommendation of a resource (the journal paper) is being inspired from other ideas and resources he would otherwise not be able to reach.

Aside from the above, the aforementioned highlighted idea concept may be used in this recommendation type as well. In paragraph 4.2.4 we argued that in some cases, the context of the highlighted idea may help the CA component provide more relevant recommendations of ideas, and hence help the user to become more creative. Regarding the resources recommendation type, we use the recommended ideas resulted by using the highlighted idea concept to fetch the resources that support them, and recommend these resources to users as relevant.

Some facts concerning the recommendation of resources can be seen below.

**Necessary Input:**

- Keywords provided by Moderator to describe the problem (“create project” stage)
- Domains provided by Moderator to define the problem’s area (“create project” stage)

**Optional Input:**

- Additional Keywords provided by user (during the recommendation procedure)
- User defined weights
- A highlighted idea (“ideation” stage)

**Context factors:**
- keywords
- domains
- Related Problems
  - Find problems relevant to the input data
  - Use the relevant problems to retrieve related resources
- Related Solutions
  - Find solutions relevant to the input data
  - Use the relevant solutions to retrieve related resources
- Related Ideas
  - Find ideas relevant to the input data
  - Use the relevant ideas to retrieve related resources

Based on the main process model the recommendation of resources will be used in the stages indicated in Figure 10.

![Figure 10: Main processes diagram. The red arrows indicate where the recommendations of resources will be provided](image)

Apart from the “Ideation Session” stage, recommendations of resources may be provided at the “Idea Evaluation” stage as well. During this procedure, the user is attempting to evaluate the ideas that resulted from the ideation session. One way of doing so, is by finding tangible evidence that support the idea (e.g. resources that
document it) and/or identify it in previous projects. The CA component could provide the user with such evidence in the form of resource recommendations.

4.2.6 Recommend Pedagogical Pattern

The CA component provides recommendations regarding the best suited pedagogical pattern to be used in the current idSpace session. The recommendation of the pattern is ontology independent. It is based on parameters provided by the moderator, regarding the specific problem he is trying to solve in the current project. These parameters are inputted by the moderator during the preparation of an idSpace session, in order to facilitate the creativity process regarding the usage scenario of the session.

Each pedagogical pattern has a combination of characteristics, which differentiate it from the others. The characteristics of each pattern are provided in xml format. The selection of the best pattern to be used is the result of a scoring method based on the values of these characteristics, and on the values of the aforementioned parameters provided by the moderator.

Within the CA component, the package developed for the recommendation of the appropriate pedagogical pattern uses the xml pedagogical pattern samples provided by WP1. The package is based on a scoring method that compares the moderator’s input with the characteristics given in the xml files. A comparison is being made between the predefined characteristics of each pattern and the moderator’s input, resulting in each pattern getting a score, based on the successful comparison hits. The pattern with the highest score is presented to the moderator as the most suitable one, considering the session’s needs.

The recommendation of the Pedagogical Pattern takes place during the “Create Project” stage of the overall creativity process as it is shown in Figure 11.
Figure 11: Main processes diagram. The red arrow indicates where the recommendation of the pedagogical pattern will be provided.

An example of how a pedagogical pattern may be selected was given by WP1 in (Bitter & Sie, 2009). Considering a usage scenario, the moderator defines the following attributes: problem type, problem definition, problem complexity, divisible problem, employees, moderator, objectives, co-operation and expert knowledge required. For instance, for the data below (Bitter & Sie, 2009), the most suitable pattern, in this example the Pyramid pattern, is presented in Table 1 (Bitter & Sie, 2009).

- Problem type: open
- Problem definition: ill-defined
- Problem complexity: medium
- Divisible problem: yes
- Employees: 7
- Moderator: yes
- Objectives: positive interdependence
- Co-operation: yes
- Expert knowledge required: yes
Table 1: Pedagogical Pattern Scoring Example

Based on the example above, the CA component processes the moderator’s data and results with the pedagogical pattern which is most appropriate to be used for the current session.
5 COMPONENT PRELIMINARY TESTING

After the CA component was developed, it became essential to examine its functionality with dummy data for testing purposes, before proceeding to its adjustment to the idSpace platform. Discussing a formal testing of the component, since it uses stored data in the topic map to calculate the recommendations, it would be mandatory to have real data with real user profiles available prior to testing, for the latter to be feasible. Since real data and real users will be available only after the adjustment of the CA component to the idSpace platform and during the testing of the platform as a whole, the testing of the component as a stand alone application was not feasible at this time. However, we have been operating our component to examine any functionality issues and to assure that its operation within the platform would be error free. Therefore, dummy data were created and saved in the topic map for the component to query, retrieve and use them in the recommendation computations. The dummy data were formatted according to the use case scenarios (presented in D5.1), meaning that they contain meaningful associations.

The purpose was to fill with data all contextual entities in the CA ontology meta-model (Figure 2) that the component uses to calculate the recommendations. More particular, the entities filled were the following: Keywords, Domains, User, Competence, User-role, Social-role, Social group, Problem, Problem statement, Creative project, Solution, Idea and Knowledge source.

The data used for operating the CA component were extracted from the ACM digital library (The ACM Digital Library, 2009), which resulted in a concrete dataset that could lead to conclusions, regarding the functionality of the component. More particular, each ACM paper was considered to be an idSpace creative session, where its contents correspond to the following ontology entities:

- Domains → Proceeding’s Sessions
- Creative Project → Title of paper
- Keywords → Paper’s keywords
- Problem & Problem statement → Text from Abstract or Introduction
- Ideas → Text from paragraphs
- Solution → Paper’s Conclusion
- Users → Authors
- Resources → References

Based on the above transformation template, the “Proceeding’s Sessions” from which the paper was selected corresponded to the “Domain” entity of the idSpace ontology meta-model schema, the title of the paper corresponded to the “Creative project” entity, the paper’s keywords corresponded to the “Keywords” entity, a part of text from the paper’s abstract corresponded to the “Problem” and “Problem statement” entities, text
from various paper’s paragraphs corresponded to the “Idea” entity, the paper’s conclusion corresponded to the “Solution” entity, the authors of the paper corresponded to the “User” entity and finally, the references of the paper corresponded to the “Knowledge source” entity. By entering the above data into the topic map, we had managed to build a database of dummy projects that supposed they were solved using the idSpace platform. Therefore, the CA component was now able to operate based in these data.

To operate, the CA component takes as initial input the keywords and domains provided by the Moderator at the “create project” stage of the idSpace platform’s workflow to describe the particular problem to be solved. Since, at the time of the above operation testing, the component had not been integrated in the idSpace platform and was thus functioning as a stand alone application, these data could not be available and therefore had to be obtained in some other way. For this reason, an interface was built for the user to be able to select a set of predefined keywords and domains. This interface will obviously not be needed in the CA component to be found in the idSpace platform, since the information describing the problem will indeed be provided by the moderator.

Moreover, the user was able to define the weights for each factor’s relevance. Based on the recommendation type chosen, a set of factors was presented to the user with the option to change their weights, so that to define the importance of each factor in the recommendation computation procedure. The weights of the factors are being used only in relation to one another, meaning that setting a factor’s weight to a certain value effects the final recommendation ranking only if the rest of the factors have weights of different value (Figure 12). Thus, if all factors’ weights are adjusted to the same value, regardless of the value, then all factors are considered equal.

The results of this preliminary functionality test were as expected. The CA component was able to provide recommendations of any type (user, idea, solution, knowledge source or pedagogical pattern) based on the user’s requests (the type of the recommendation she wishes to be provided), based on her input (keywords, domains and weights) and by using the topic map containing the dummy projects. The recommendations were presented ranked and with a link attached to them, so that the user could click on it and further explore the entities proposed by the component.

Figures 12 and 13 illustrate an example regarding solution recommendations. Here, the user wishes to receive recommendations of solutions to past problems that are related to the active problem that is confronted at the current idSpace session. In the first figure, the user has selected the keywords that best describe the active problem and the domains she believes that this problem belongs in, from the predefined keywords and domains in the lists. Moreover, the user has selected the weights for the four factors, denoting a preference in keywords and problem statements by selecting a “High Relevance” weight for them. Finally, the user has inputted additional keywords to be considered by the CA component, so that the recommendations provided would consider them as well. After submitting the data, the CA component provides the ranked recommendations presented in Figure 13.
Figure 12: The user inputs data to be recommended of solutions

Figure 13: The CA component provides the solution recommendations ranked

By following the “Go to Project” link below a recommended solution, the user is presented with the project’s profile, in order to learn more about the context in which the aforementioned solution was created. By doing so, the user may be informed about the project’s initial problem, its problem statement, other solutions, ideas and resources proposed during the project’s session, as well as the people attending the session and more.
6 CONCLUSIONS AND FUTURE WORK

The aim of this deliverable is the description of the work that has been done for the development of the Context Awareness component prototype v.2. A discussion is made about why the technologies presented in the previous deliverable are not used here, and the new technologies used for the development of this second version are described. Following, the range of recommendations developed are presented, as well as an analysis of how they were developed, which context factors each one uses and when each one is triggered. In addition, the ontology currently used by the component is provided, along with a presentation of the ideal ontology schema. Finally, the basic functionality of the application is preliminary tested with the use of dummy data (added to the topic map), and the indicative results were presented.

After completing the implementation of prototype version 2, it has been made apparent that it now better meets the requirements set by the idSpace platform. It covers the most important and meaningful recommendations one would expect to find in a creativity support tool. The logic behind the computations of the recommendations is based entirely on the assumptions made in idSpace and its particular characteristics. The goal at all times is to offer the users useful recommendations that will support them during the creativity process. The types of recommendations included in prototype version 2, along with the reasoning behind their design and development aim at achieving this goal.

The CA component prototype v2 has been tested with respect to its functionality and has been found stable. It is therefore ready to be integrated in the overall idSpace platform. As with all integration procedures, some obstacles may be encountered that would obviously have to be overcome. The incorporation of the CA component in the platform will allow an appropriate testing, as well as a formal evaluation. Evaluation in this context would concern both accuracy of recommendations and also their usefulness for the user.
REFERENCES


APPENDIX A: CONTEXT AWARENESS COMPONENT CODE

A.1 Recommend pedagogical patterns

```java

<%!  
public boolean isTextNode(Node n)  
{  
   return n.getNodeName().equals("#text");  
}  
%>

<%  
String problemType=request.getParameter("problemType");  
String problemDefinition=request.getParameter("problemDefinition");  
String complexity=request.getParameter("complexity");  
String problemDivisible=request.getParameter("problemDivisible");  
String minParticipants=request.getParameter("minParticipants");  
String maxParticipants=request.getParameter("maxParticipants");  
String preferredParticipants=request.getParameter("preferredParticipants");  
String moderator=request.getParameter("moderator");  
String procobjective=request.getParameter("procobjective");  
String cooperation=request.getParameter("cooperation");  
String expertKnowledge=request.getParameter("expertKnowledge");  
String phase=request.getParameter("phase");  
%>

<html>
<head><title>Parsing using the DOM</title></head>
<body>

<%  
DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();  
DocumentBuilder builder = factory.newDocumentBuilder();  
Document doc = builder.parse("http://localhost:8080/WP1/strategy.xml");  
%>

<h1>List of Patterns</h1>
```
<table>
<thead>
<tr>
<th>Identifier</th>
<th>Title</th>
<th>Description</th>
<th>Problem Type</th>
<th>Problem Definition</th>
<th>Complexity</th>
<th>Problem Divisible</th>
<th>Min Participants</th>
<th>Max Participants</th>
<th>Preferred Participants</th>
<th>Moderator</th>
<th>Procobjective</th>
<th>Cooperation</th>
<th>Expert Knowledge</th>
<th>Phase</th>
</tr>
</thead>
</table>

```java
Element root = doc.getDocumentElement(); // "patterns" node
NodeList strategyNodes = root.getChildNodes(); // 2 "person" nodes
int[] totalArray = new int[strategyNodes.getLength()];
for (int y=0; y<totalArray.length;y++)
{
    totalArray[y]=0;
}
for (int i=0; i<strategyNodes.getLength(); i++)
{
    int hits=0;
    Node currentStrategy = strategyNodes.item(i);
    NodeList attributeList = currentStrategy.getChildNodes(); // "name" and "age" nodes
    if (!isTextNode(currentStrategy)) // skip whitespace node
        continue;
    NodeList attributeList = currentStrategy.getChildNodes(); // "name" and "age" nodes
    %>
    <%
    if (strategyNodes.item(i).getFirstChild().getNodeValue().equals(problemType))
        hits=1;
    totalArray[i]+=hits;
    }
    if (strategyNodes.item(i).getFirstChild().getNodeValue().equals(problemDefinition))
        hits=1;
    totalArray[i]+=hits;
    %>
```
if (strategyNodes.item(i).getChildNodes().item(11).getFirstChild().getNodeValue().contains(complexity))
    {
        hits=1;
        totalArray[i]+=hits;
    }
if (strategyNodes.item(i).getChildNodes().item(13).getFirstChild().getNodeValue().equals(problemDivisible))
    {
        hits=1;
        totalArray[i]+=hits;
    }
if (strategyNodes.item(i).getChildNodes().item(15).getFirstChild().getNodeValue().equals(minParticipants))
    {
        hits=1;
        totalArray[i]+=hits;
    }
if (strategyNodes.item(i).getChildNodes().item(17).getFirstChild().getNodeValue().equals(maxParticipants))
    {
        hits=1;
        totalArray[i]+=hits;
    }
if (strategyNodes.item(i).getChildNodes().item(19).getFirstChild().getNodeValue().equals(preferredParticipants))
    {
        hits=1;
        totalArray[i]+=hits;
    }
if (strategyNodes.item(i).getChildNodes().item(21).getFirstChild().getNodeValue().equals(moderator))
    {
        hits=1;
        totalArray[i]+=hits;
    }
if (strategyNodes.item(i).getChildNodes().item(23).getFirstChild().getNodeValue().equals(procobjective))
    {
        hits=1;
        totalArray[i]+=hits;
if (strategyNodes.item(i).getChildNodes().item(25).getFirstChild().getNodeValue().equals(cooperation)) {
    hits=1;
    totalArray[i]+=hits;
}
if (strategyNodes.item(i).getChildNodes().item(27).getFirstChild().getNodeValue().equals(expertKnowledge)) {
    hits=1;
    totalArray[i]+=hits;
}
if (strategyNodes.item(i).getChildNodes().item(29).getFirstChild().getNodeValue().contains(phase)) {
    hits=1;
    totalArray[i]+=hits;
}
System.out.println(strategyNodes.item(i).getChildNodes().item(1));
System.out.println(totalArray[i]);
max=totalArray[x];
strategy=count;
}
// System.out.println(strategy);
System.out.println(totalArray[x]);
count=count+1;
}

NodeList nodeList = root.getElementsByTagName( "strategy" );
//NodeList nodeList2 = root.getElementsByTagName( nodeList.item(strategy));
System.out.println(nodeList.item(strategy).getChildNodes());
NodeList nodeList2 =nodeList.item(strategy).getChildNodes();

for (int j=0; j<nodeList2.getLength(); j++ )
{
Node currentItem = nodeList2.item(j);

if ( isTextNode(currentItem))
    continue;

    <td><%=currentItem.getFirstChild().getNodeValue() %></td>
    <% }%> </tr>
</table>
</body>
</html>

A.2  Code samples from the recommendation packages which use
factors and relevance function

A.2.1  Sample of the Tolog queries used

for(ptr=0;ptr<krdDmnlDsArray.length;ptr++)
{
    String javaQuery = "role-player($Role, "+krdDmnlDsArray[ptr]+"), association-role($Assoc, $Role), association-role($Assoc, $Role2), role-player($Role2, $resultTopic), $Role /= $Role2, instance-of($resultTopic, "+ ProblemTopicId + ")?";
    //out.println("javaQuery: "+javaQuery);
    %>
    <ul>
    <%--<li>--%>
    <tolog:foreach query="%=javaQuery %>"
    <%--<li> --%>
A.2.2 Sample of the Relevance Function Implementation

```java
for(int i=0;i<problemIdsArray.length;i++)
{
    ptr=i+1;
    overallScoringArray[i]=(W1*Double.parseDouble(fkeywordScoresArray[i])+
W2*Double.parseDouble(ffdomainScoresArray[i])+W3*Double.parseDouble(ProblemScoresArray[i])
)/((W1+W2+W3)*3);
}
```

A.2.3 Sample of the Scoring Process

```java
for(k=0;k<ArrayLength;k++)
{
    for(p=0;p<ArrayLength-1;p++)
    {
        if(overallScoringArray[p]<overallScoringArray[p+1])
        {
            temp=overallScoringArray[p];
            overallScoringArray[p]=overallScoringArray[p+1];
            overallScoringArray[p+1]=temp;

            tempStr=problemNamesArray[p];
            problemNamesArray[p]=problemNamesArray[p+1];
            problemNamesArray[p+1]=tempStr;
        }
    }
}
```
tempId = problemIdsArray[p];
problemIdsArray[p] = problemIdsArray[p+1];
problemIdsArray[p+1] = tempId;
}
Definition and Implementation of Context Awareness v2