EAI Endorsed Transactions
on Context-aware Systems and Applications
Research Article EAI.EU

Context-based Project Management

Ammar Alsaig[1,*], Alaa Alsaig[2] and Mubarak Mohammad[3]

[1]Umm AlQura University, Montreal, QC, Canada
[2]Concordia University, Montreal, QC, Canada
[3]Concordia University, Toronto, ON, Canada

Abstract

Context-based computing has become an integral part of the software infrastructure of modern society. Better software are made adaptive to suit the surrounding environment. Context-based applications best fit into environments that undergo constant and frequent changes. Temperature management, Time management, GPS are just few examples where context-awareness becomes inevitable. Project Management is another domain that requires constant monitoring. The current tools of project management handle data gathering, plotting, and organizing, but requires high-level of human intervention to analyze data and integrate it. To the extent of our knowledge there is no efforts to introduce context awareness to project management domain. In this work, we introduce context and formally model project context using FCA. Additionally, we provide the results of the full implementation of our approach on a real-world software project. We show that our approach can formally answer queries that traditional tools could not answer. Also, we introduce a brief comparison between our approach and traditional project management software. Finally, we show that our approach can improve project management tools and minimize the effort spent by project managers.

Keywords: Context, Project Management, Formal Concept Analysis (FCA).

Received on 06 May 2017, accepted on 19 June 2017, published on 06 July 2017

doi: 10.4108/eai.6-7-2017.152902

1. Introduction

Context-awareness and context-based applications have become part of our everyday life. Ubiquitous computing [4], cloud computing [5], mobile applications [6], cyber physical systems [7] and others are just few fields and applications where context-awareness plays significant role. In this paper, we introduce context to the field of project management. Our goal is to make better project management tools by introducing the notion of context and using formal data modeling.

There are two essential components of project management: service requester (client), who defines requirements, and service providers (vendors) who should execute the project with respect to the defined requirements by client [3]. Project Managers, who work for service providers, aim to achieve the best balance between Time, Cost and Quality that can fulfill the service requester needs and maximize the profit (minimize the cost) for service

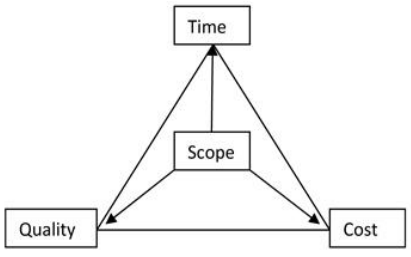

Figure 1 Main Components of Projects

*Corresponding author. Email:saigammar@gmail.com/aasaig@uqu.edu.sa

providers. We refer to client requirements and vendor data as *Project Context*. The high degree of dependency between project context components is illustrated in [Figure 1]. The balance between different components of project context depends on how well project managers are aware of the relationships between different data components within it. Project managers often need to reconsider their decision regarding the trade-offs they need to commit during the development. Thus, questions like: what is the effect of an absence of certain resources assigned to a project? What are the relations between resources and risks? what are the relations between tasks and risks? are examples of many other questions that need ready answers for project managers to make their decisions.

Project context is highly changeable. It is often redefined or updated during the project life cycle. This reflects changes also in the relations among data components of project context, which makes it challenging for project managers to follow as the project evolves. In a study provided by Seiler, M., & Paech, B, from a survey done in this study, experts emphasize the problem of tracking changes and progress that is done on features with consideration to the affected elements by those changes [18]. In many cases, project managers lose track of project changes which drive projects to failure. The current project management systems/tools such as Microsoft project management [7], redmine [9], and SAP project management [8] to name few, focus on the different area of project such as time management, cost management, resource management, and risk management. These tools work as an organization of project data. However, when it comes to relationships within the project context itself, these tools need a human analyst/project manager to put things together and to watch out for emerging risks as the project context changes [16]. In [10], the authors introduced a survey on IT project tools. It is mentioned that one of the strong reasons of projects failure is "*Not having a system in place for approving and tracking changes*". This study suggests that: "*Having a clear process that must be followed is the best way to ensure the pertinent details -- how much it will cost, why it is necessary, the impact on the overall project -- are known before the change is approved. It's also extremely effective for auditing performance during and after project completion*".

Therefore, we propose a formal approach to monitor the changes in project context and keep track of the emerging relationships between its main components. We start from categorizing and classifying project data. Then, we formally model the data using FCA-lattice. Finally, we provide a full implementation of our approach.

1.1. Running Example

As a proof of concept, we included in this paper a simple project as an example. This example, starting from client requirement, until laying out the project plan, is a running example that will gradually evolve with every section as we explain our approach. In every section, we refer to this example and illustrate how it is managed and modeled using our approach.

The example illustrates the planning of a project titled as "user-friendly easy calculator", where client requires a calculator software to be implemented within a week of time and 500 CAD of budget.

2 Data of Project Management

There are three types of project data categorized by its source: client, provider, and project managers. In addition, there is also data that dynamically emerges and develops during the project life cycle. Thus, we classify context project data into the following:

A) Client Context (CC): It can also be called "input data". This data is the information based on which the project is initiated. In most cases, it defines the scope of the project, time frame and available budget. In some cases, some of this information is left open to be defined by service provider, which means service provider can trade-off on this open specification. For example, if the time frame is left open for service provider, service provider can enhance quality, and minimize cost by extending the period of development and/or by assigning less resources to work on the project.

B) Project Context (PC): data that is defined by project manager of service providers. It includes the expected times of deliverables, the resources assigned on the project and their related tasks, the cost of these resources, and the risks associated with the project. This data can be updated later on when project context changes. For example, client during the project development decreases the time frame of the project. Project managers should consult the project plan, check the available resources, and review the costs and risks to know the effect of this decrease on the project context.

C) Relations Context (RC): This data represents the relationships among the entities and components of project context. It is domain dependent. Also, it is considered as the connection between each data component. For example, the link or relation between a task, to whom it is assigned and the risks related to it, is defined in the Relations Context. In the scope of this paper, we will deal with relations between different project information, e.g. budget depends on quality, as given facts from the domain of project management. The explanation of how and why these relationships there are, i.e. how time and quality are related, is out of the scope of this paper. Interested readers can refer to the project management references [1] [2] [3].

3. Modeling Data

In this section we introduce Formal Concept Analysis. Also, we model each data entity introduced in previous section using FCA.

3.1. Formal Concept Analysis (FCA)

FCA is a formal approach that defines relations between different data entities. In [17], they define FCA as *"a method mainly used for the analysis of data, i.e. for deriving implicit relationships between objects described through a set of attributes on the one hand and these attributes on the other. The data are structured into units which are formal abstractions of concepts of human thought, allowing meaningful comprehensible interpretation (Ganter & Wille, 1999)"* .

Because our focus is to capture and keep track of relationships within project context, we rely on FCA to formally define these relations. Therefore, our first class object is the formal concept. This means that all data entities we have in our model will be based on objects and attributes. Formally, a formal context is a triplet <X,Y,I> where X and Y are non-empty sets and I is a binary relation between X and Y. That is, $I \subseteq X \times Y$. X represents the set of objects of the formal context and is referred to as "Extent" and Y represents the set of attributes and is referred to as "Intent". For example, consider a formal context FC that has a set of objects X and a set of attributes Y. Let x, y be an object and attribute that belong to X and Y, respectively. We say there is a relation between x and y if and only if the pair <x,y> is defined in the binary relationship $I, i.e. < x, y > \in I$. This means that the set of pairs defined for a formal context are ordered pairs.

FCA has two formal operators called Concept Forming Operators. Which simply defines the extent and intent of any Formal Concept [6]. Hence,

$$Intent = A\uparrow = \{ y \in Y \mid for\ each\ x \in A :< x, y > \in I$$
$$Extent = B\downarrow = \{ x \in X \mid for\ each\ y \in B :< x, y > \in I$$

The above two formal operations navigate through the different concepts in formal context until a fixpoint is reached. Fixpoint means no further relation can be discovered and once reached a formal concept is reached. Formally, fixpoint is defined for a concept <A,B> where $A \in X\ and\ B \in Y$, as follows:
The pair <A,B> is a formal concept(fixpoint) iff $A\uparrow = B \wedge B\downarrow = A$

Thus, the set of ordered pairs of concepts along with the operators (↑,↓) form what is referred to as closed and complete FCA lattice Tree. Which exhaust all possible pairs of objects and attributes in the concept. The extent and intent operations can help us traverse the tree in both direction from least upper bound to the greatest lower bound and discover all different relationships that can be explored until a formal concept (fixpoint) is reached.

3.2. Client Context

Client Context(CC) is the input data defined by client. It usually includes requirements/needs, time constraints and budget. In most cases, client submit these documents in forms of contracts, tables, excel sheets or bulletins to service providers. In turn, service providers start making and implementing the project plan. In our framework, we map client information to set of data component, each data component is made up of set of items. For example, "needs data component" is where all client requirements are listed, "time data component" is where all deliverables that are bound by specific time are listed, and so on. In Client Context CC of a project P <CC(P)>, there are no relations defined. Thus,

$$CC(P) = \{DC_1, DC_2, \dots, DC_N\}$$

where, DC is a data component that consist of set of items (i_x),

$$DC_x = \{i_1, i_2, i_3, \dots, i_M\}$$

Example

In our example, the client data are given as follows:
"We want to build a nice, user-friendly calculator that's able to perform the four basic functions and that has a precision of 2 decimal points at least. This is to be done in no longer than a week. The price offered for this project is 500 CAD".
In our model and based on the above explanation, this is mapped to three data components:

1- *Client needs component (CNC)= {nice interface, addition, substraction, multiplication, division, two decimal precision}*

2- Client time component (CTC) = {7 days}

3- Client budget component(CBC) = {500 CAD}

Thus,

$$CC(\text{"}calculatorProject\text{"}) = \{CNC, CTC, CBC\}$$

3.3 Project Context

Project Context represents that actual project plan. It is similar to Client Context. However, it includes relations that links between Client Context and provider plan. Thus, it is defined as FCA as opposed to set of items. That is, every data component in Project Context is defined as an FCA with objects, attributes and relations among them (if any). In standard project management tools, project plan starts by listing all (technical) requirements and map them to time,

resources. However, in our model, project manager should map client context, i.e., needs, time, and budget, to the technical requirements, risks, and resources. Thus, Project Context is defined as set of FCA components, where each FCA component <A,B,I>, where A is set of items in a data component in Client Context, and B is a set of items relevant to A defined by project manager. Project Context can have as many FCAs as project manager finds necessary.

Formally, Project Context PC of a project P <PC(P)> is defined as,

$$PC(P) = \{F_1, F_2, \dots, F_N\} = \{< DC_1, PDC_1, I_1 >, < DC_2, PDC_2, I_2 >, \dots, < DC_N, PDC_N, I_N >\}$$

where, each F is an FCA concept such that DC_1 represents object which is a data component from $CC(P)$ or a defined items by project manager, and PDC_1 is the attributes which are the project manager set of items related to it, and I_1 is a binary relation. Note that DC_1, can be also reused with PDC_2 if project manager finds it suitable. It is left to him as a domain expert to use data components from Client Context and link it to some items in Project Context.

Example

The project manager should now link each client need (data component) to set of items/actions to be done to meet the client requirement. This is to be performed as necessary for each data component [†]. In our example, the project manager made four components in Project Context, defined as follows [Figures 2,3,4,5]:

1- *A1 = CNC/ Technical Requirements (TR)*

2- *A2 = TR/ Resources(R)*

3- *A3 = CTC/ Technical Requirement(TR)*

4- *A4 = Risks (RK)/ R*

Thus,

$$PC(\text{“}calculatorProject\text{”}) = \{A_1, A_2, A_3, A_4\} = \{< CNC, TR, I_1 >, < TR, R, I_2 >, < CTC, TR, I_3 >, < RK, R, I_4 >\}$$

A1	database design	database tuning	DB package	GUI	f1 dev	f2 dev	f3 dev	f4 dev	f5 dev	testing
nice interface	X	X	X	X						
addition	X	X	X		X					X
substraction	X	X	X			X				X
multiplication	X	X	X				X			X
division	X	X	X					X		X
2decimal point perci	X	X	X						X	X

Figure 2 Client Needs VS Technical Requirements FCA Data Component

database tuning	X			X
DB package	X			X
GUI			X	X
f1 dev			X	X
f2 dev			X	X
f3 dev			X	X
f4 dev		X		X
f5 dev		X		X
testing	X		X	X

Figure 3 Technical Requirements VS Resources

A3	database design	database tuning	DB package	GUI	f1 dev	f2 dev	f3 dev	f4 dev	f5 dev	testing
2nd day	X	X	X							
4th day				X	X	X				
6th day							X	X	X	
finish										X

Figure 4 Deliverables VS Technical Requirements

A4	R1	R2	R3	PM
only one resource	X			
busy resource	X	X		
can't accomplish on time			X	
costly		X		X

Figure 5 Risks VS Resources

[†] What we perform in this example is not exhaustive and project managers are free to add any relation they want in project context.

3.4. Relations Context

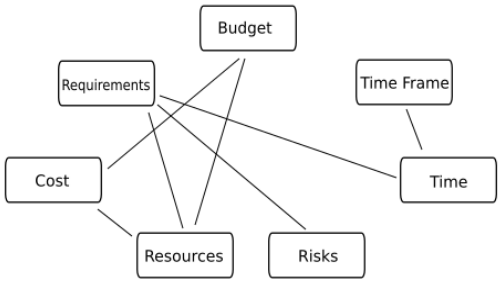

Figure 6 Relations Context for Project Context

Relation Context (RC) is to link between different FCA components within the Project Context [Figure 6]. It is a formal approach to facilitate searching for relevance within the set of components in Project Context. RC should also be defined by project manager. Formally it is defined as,

$$RC(P) = < F_1, F_2, I >, < F_1, F_3, I >, \dots, < F_N, F_M, I >$$

where $F_x \in PC(P), I$ is a binary relation set.

Example

As Project Manager defines the relations between client context and their plan, he should map the data components that have attributes/objects in common to each other in RC [Figure 7]. Thus, RC becomes,

$$RC(\text{“}calculatorProject\text{”}) = \{< A_1, A_2, I >, < A_2, A_3, I > < A_1, A_3, I >< A_2, A_4. I >\}$$

Relation Context				
	A1	A2	A3	A4
A1		X	X	
A2	X		X	X
A3	X	X		
A4		X		

Figure 7 Project Context VS Project Context (Internal Relations)

4 Query Answering Algorithm

This algorithm is used to find all relevant information to a particular data component or piece of information in Project Context. Also, it is used to see the reflection of a data component of in project context on the Client Context. It is based on formal approaches and hence, it is considered a formal approach to answer any query on project context.

The algorithm is enumerated as follows:

1- Determine given information, goal information, and data component

   a. **Given information** is the information that a project manager has and wants to see its relation to other information. Store it in variable “input”

   b. **Goal information** is the information/data component on which the project manager wants to see the given information effect. Store it in variable “G”

   c. **Data component** is a data component in Project Context that contains the given information. Store it in variable “DC”

2- Determine the relations to that data component in Relations Context.

   a. This is achieved by running extent or intent (both are similar in this stage) of DC on the Relation Context. Store result in list “L”

3- Determine bridging results (results through which project manager can find transitive closure and relevant information)

   a. This is achieved by running extent of “input” on DC. Store it in “extentR1”

   b. Run intent of “input” on DC. Store it in “intentR1”

   c. Bridge1 = intentR1 UNION extentR1.

4- For each item “i” in “L” run the following and store in “Ans” (the “Ans” set, by the end of this step, will include all direct/indirect pieces of information affected by input data):

   a. extent of “Bridge1” on “i” UNION intent of “Bridge1” on “i”

5- Finding Results,

   a. Intersect data component content (list of items) or pieces of information in “G” with “Ans” to find affected information and store it in “Result”.

6- If “Result” is empty set, Set “input” to “Ans” and re-run algorithm from step(2). Keep running until all data components are investigated. If no results, then there is no relationship to “input”

4.1 Algorithm Pseudocode

```
input = Determine Input Data {}
DC = getDataComponentName(input){}
G = setGoal("project manager goal"){}
check_here:
L= extent(DC,RelationContext) U intent(DC,
  RelationContext){}
if(Empty) {
  Result  = Empty;
}
bridgingR = extent(input,DC) U intent(input
  ,DC){}
while (i in L){
  Ans = extent(bridgingR, i) U intent
    (bridgingR, i);
}
Result = G ∩ Ans{}
if (Result = Empty){
  setGoal(Ans);
  loop check_here;
}
Return Result;
```

4.2 Algorithm Workflow

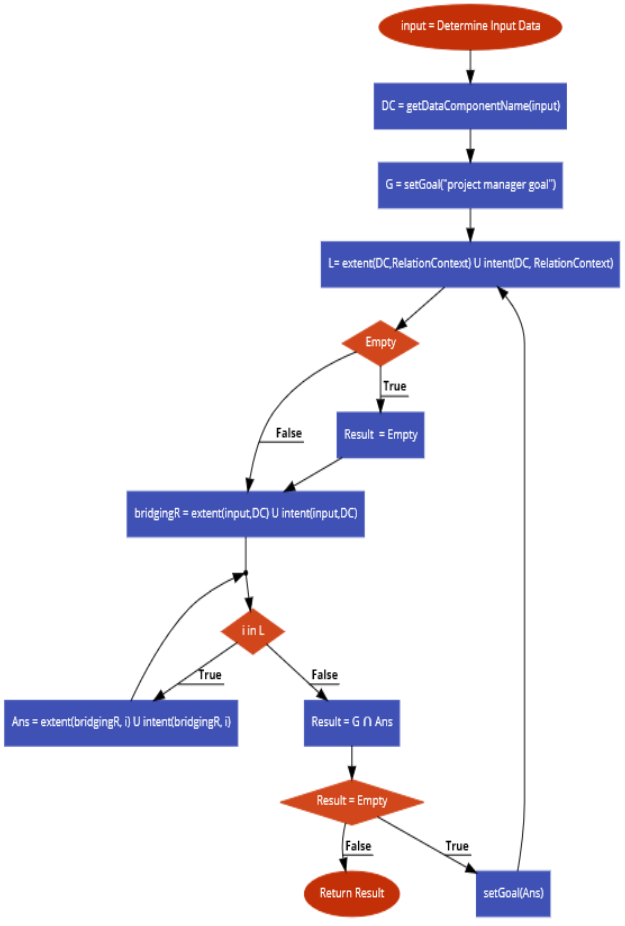

Figure 8 Query Algorithm Workflow

Example

In our running example, let's assume that project manager wants to find the effect of the absence of his "R3" resource on the client needs. Following the steps described above in the search algorithm:

1- $input = "R_3"\ G = "CNC", DC = A_2$

2- $L = Extent(DC, RelationContext) = \{A_1, A_3, A_4\},$

3- $BridgingResult_1 = Extent("R_3", "A_2") \cup Intent("R_3", "A_2") = \{GUI, f_1\ dev, f_2\ dev, f_3\ dev\}$

4- *On each item "i" in "L" run:*

$$Ans = Extent(BridgingResult_1, i) \cup intent(BridgingResult_1, i)$$

5- *Intersect "G" with "Ans" to get the client needs that are affected by the resource "R3" ,*

$$Result = G \cap Ans$$
$$= \{nice\ interface, addition, substraction, multiplication\}$$

Through this example we were able to formally find the affected client needs by the resource "R3" in Project Context"

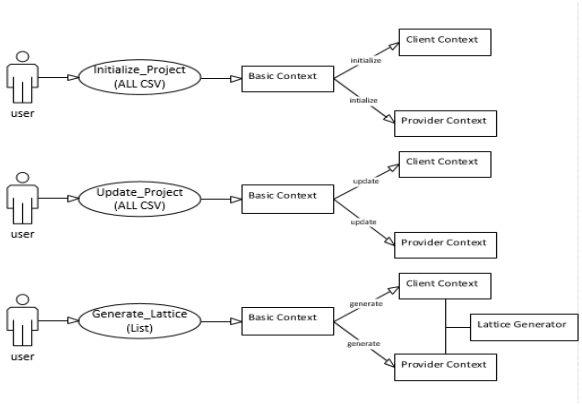

Figure 9 Main classes in our implementation

4 Implementation Tools

The main theoretical method on which our implementation is based is FCA-lattice method. Through its formally well-defined functions and concepts, the implemented program initiates the project and updates its context as the requirements change. In our implementation we use python as the main programming tool. Details of both theoretical method and programming tools are going to be discussed in this section.

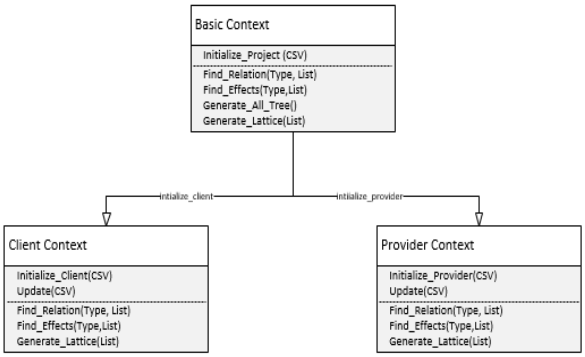

Figure 10 General Use cases

4.1 FCA functions

As explained earlier in Section 3.1, the two formal functions used to find relations between objects and attributes are intent and extent which are both used to achieve a formal concept.

Informally, the extent function returns all attributes that are shared by input object(s), where the intent function returns all objects that are shared by input attribute(s). For example, if Relations Context links a task (T) with four resources (R), the extent function on T gives back R, while the intent of R returns back this T. Based on those two functions, our program can find all attributes shared by list of objects or all objects that are shared by list of attributes.

4.2 Programming Tools

The programming language that we have used to implement our approach is Python 2.7. The reason of picking up python is because of its ready implementation of FCA and its main two methods, namely intent and extent. Also, the graphics libraries in Python come in handy when representing the lattice tree generated to represent the relations between data components. In the following subsections we explain the functions and methods used from ready libraries and the other main customized functions we used in our implementation. Figure 9,10 shows the basic classes in our implementation.

4.2.1 Off-Shelf functions and libraries

The following table gives brief information about the functions and libraries that have been used as.

Library/ Function	Details
Concept	The main FCA library that has a component called Context that implements both intent and extent function in addition to graphic library to represent the lattice tree
Graphviz	The main library to represent the relations in pdf/png file.
Context.FromFile(filename)	Function to read the FCA relations between objects/attributes from CSV file there are other function to define the context inline in the source file.
Context.intension('objects')	Function that takes object/list of objects and returns list of attributes shared by the same object/objects.
Context.extension('attributes')	Function that takes attribute/list of attributes and returns list of objects shared by the same attribute/attributes.

4.2.2 Customized Functions

The following gives brief information about the functions and libraries that we have implemented.

Library/ Function	Details
Initialize_Project(CSV) **Initialize_Client(CSV)** **Initilaize_Provider(CSV)**	These three functions do the same thing. They are called only at the beginning of the project. They take as parameters all csv files that contain the relations between data components and the basic relationships.
Update (type, updateQuery)	Similar to init functions but take either query (key value format) or a csv file that replaces another one.
Find_effects(type, List)	This function will use either intension or extension formal FCA functions depending on the type argument. Type can be object or attribute. List should be list of objects or attributes.
Find_Relations (type, List)	This function find all relations of an item/group of items (objects/attributes).
Generate_All_Tree() **Generate_Lattice(List)**	These functions are similar. One generate a lattice for all project, the other one does this for a specific subset of objects.

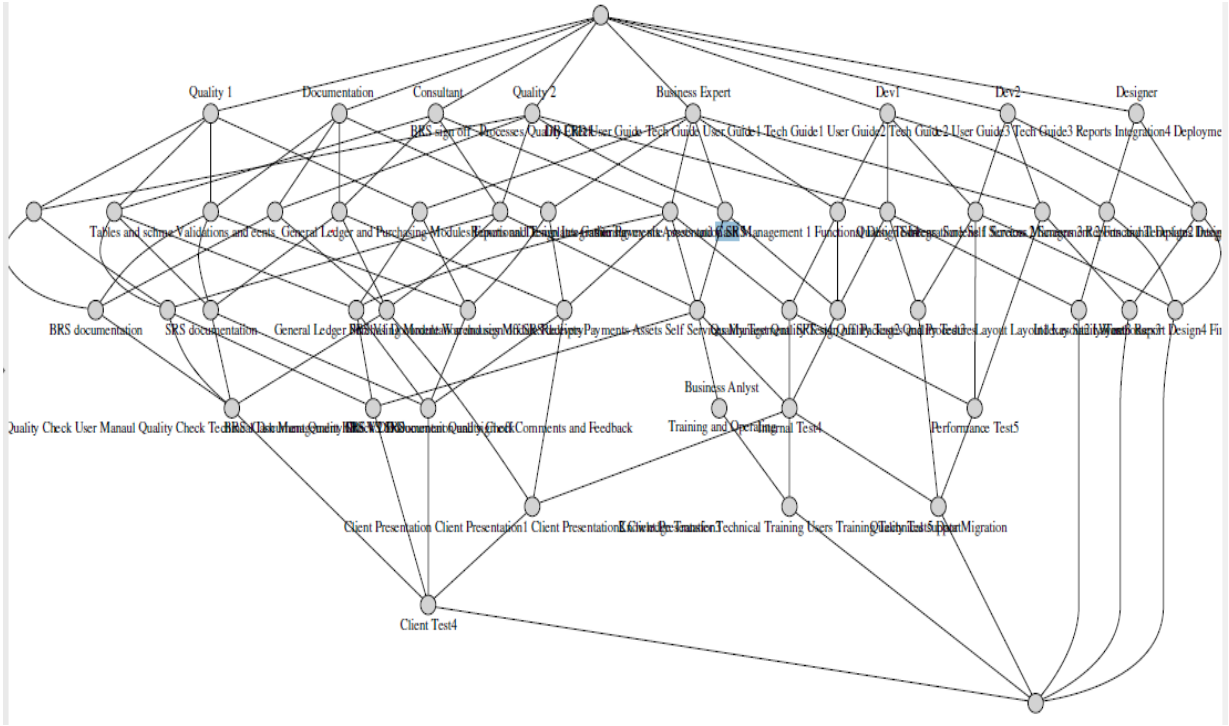

Figure 11 Case study, FCA lattice generated from our program

4.2.3 Real World Case Study

We have deployed our program on a real software Enterprise Resource Planning (ERP) project. This project includes 260 tasks, 9 resources, 20 risks, and had to be implemented in 7 months. We were able to model all project data using the above explained categorization. This data includes the initial data and the evolved data as project progressed. We were able to see the project from different point of view with the help of FCA extension and intension. The following brief test cases show some examples of this information. Figure 11 shows full lattice tree for the project plan.

- Test Case: Find all risks associated with one resource.
    - Result: our program shows all tasks associated with one resource, and shows all risks associated with these tasks.
- Test Case: Find the effects of updating requirements and resources
    - Result: through the Relations Context and Project Context, system was able to find all relations shared by requirements and resources, and relations associated with resources and ones associated with requirements.
- Test Case: Find effects of Increase in resource cost per month
    - Result: our program showed all tasks associated with that resource, and all risks.

5 Overall Observation

After full implementation of our approach on a real-world case study, we observed interesting findings. First, our approach does not only help in formalizing the relations among data entities but also to infer some relations based on the dependencies between data entities. Also, we were able to find any linked information to any piece of information in the project context which was not available with traditional PM tools. For example, finding assigned resources on a specific task, which risks linked to it, what is the time assigned to the task, and so on. This allows us to find the echo of any change in the project context. However, it is worth saying that our current implementation cannot replace the project management tools but can be deployed side by side to these tools to empower them and minimize the effort spent by project managers to update project context. We also give brief comparison table between our method and the standard project management tools in Table 1.

Criteria	Our method	Standard PM tools
Based on Formal Approach	YES	NO
Capture all types of relations	YES	NO
Easy update (no multiple places)	In Most cases	NO
Provide Gantt charts and other PM charts	NO	YES
Has easy-user friendly interface	NO	YES
can represent relationships in a figure between all data entities	YES	NO

Table 1 Context-based approach VS. Traditional tools

6 Literature Review

In [11], a comparative study is done on twenty project management tools to view their features and summarize them. In the study, none of the mentioned software was characterized with the ability to show the interconnection between one task and another. That is, the change in one task could be tracked by the available software but not the affected tasks by these changes.

Adding the features of tracking the changes and building the dependencies among tasks using programming languages is possible as XPSuite did in their research [12]. However, it is not based on theoretical and formal methods. Formalism is required to provide an authorized definition for the relationships that exist among tasks within a specific project. Some researchers work to define the relationship among organizations [13]. The relationship is between an organization and another is based on sharing the work on specific tasks and the dependencies among these tasks. The research [13] suggesting to build a complete platform that is shared by all participating companies to have the ability to track tasks and notify others about progress that has been done on specific tasks. However, the methods that is used to define the relationship is not defined that keeps the need to define how to build relationship formally. Moreover, by defining a class object in project management software that build the relationship among tasks, the relationships among other objects or classes of the software can be defined using the same method.

According to [15], the writer introduced SMIT, which is a project management software that is able to plan and re-plan tasks within a project. That is, SMIT structured the tasks in hierarchy tree that is built on the relationship and dependencies among objects, attributes, and tasks. This is to give the ability to SMIT to plan or re-plan again whenever some changes that happen during project life cycle. SMIT is a great work and has very powerful features. However, the relationships among tasks were not defined formally. That is, SMIT software is not intelligent enough to realize the interconnection between tasks. The relationship was based on input given by the user, however, SMIT software is not aware of it. In our research, we introduced a software that is intelligent enough to build relationship between existing tasks or added tasks during project life cycle. These relationship is built using theoretical formula (FCA) that does build the connections among object based on context information.

In addition, the study [19], provides the most powerful project management tools, among 119 tools, in terms of scheduling and planning, which are Workfront, Genius Project, Oracle Primavera, LibrePlan, Sciforma, JIRA, and Microsoft Project. Although these methods are offering a great job for project managers in the beginning of a project, they cannot dynamically adapt the schedule to the changes through the life cycle of a project [19]. That is, there is not a formal method that can define the dependent tasks on a specific task to provide the ability to reschedule the tasks and the time table according to any new decisions or update. This shows the need to the provision of a method that could formally define the relationship and dependencies among projects tasks and elements.

7 Conclusion

Current Project Management practices need high-level human involvement to initiate and update projects as they evolve. Losing track of interrelations within project data due to frequent updates is one of the main causes of project failures. Our approach formally defines relations between different data entities. This keeps all relations captured along projects life cycle. This also provides easy update to project plan with clear view on effect on other part of the project. After implementing real world software project, we found that our approach was able to model all data of project successfully. Also, with the help of FCA intention and extension we were able to find any relationship that is direct or inferred from the project context. Finally, we have provided a brief comparison between our method and tradition project management tools. Although our method cannot replace traditional project management tools, it surely can strengthen the data integration, simplify context update and minimize human intervention. As a conclusion, our work is considered to be the first step towards a full automation of project updates and automatic track of changes.

8 References

1. Lewis, James P. (2005). Project Planning, Scheduling & Control, 4E. McGraw Hill. ISBN 978-0-07-146037-8.
2. Michael W. Newell, Marina N. Grashina (2004). The Project Management Question and Answer Book. p.8.
3. PMBOK Third Edition 2004 p.165
4. Weiser, M. (1993). Some computer science issues in ubiquitous computing. Communications of the ACM, 36(7), 75-84.
5. Mell, P., & Grance, T. (2011). The NIST definition of cloud computing.
6. Charland, A., & Leroux, B. (2011). Mobile application development: web vs. native. Communications of the ACM, 54(5), 49-53.
7. Baheti, R., & Gill, H. (2011). Cyber-physical systems. The impact of control technology, 12, 161-166.
8. Lowery, G. (1997). Managing Projects With Microsoft Project 4.0: For Windows and MacIntosh. John Wiley & Sons, Inc.
9. Welti, N. (1999). Successful SAP R/3 implementation: Practical management of ERP projects. Addison-Wesley Longman Publishing Co., Inc.
10. Lang, J., & Davis, E. (2010). Redmine-open source project management web-application.
11. Schiff, J. L. (2012, September 26). 12 Common Project Management Mistakes--and How to Avoid Them. Retrieved from CIO: http://www.cio.com/article/2391872/project-management/12-common-project-management-mistakes--and-how-to-avoid-them.html
12. Mishra, A., & Mishra, D. (2013). Software project management tools: a brief comparative view. ACM SIGSOFT Software Engineering Notes, 38(3), 1-4.
13. Angioni, M., Carboni, D., Melis, M., Pinna, S., Sanna, R., & Soro, A. (2004, November). XPSuite: tracking and managing XP projects in the IDE. InProceedings of the 2004 workshop on Quantitative techniques for software agile process (pp. 46-52). ACM.
14. Aoyama, M., Yabuta, K., Kamimura, T., Inomata, S., Chiba, T., Niwa, T., & Sakata, K. (2014, June). A Resource-Oriented Services Platform for Managing Software Supply Chains and Its Experience. In Web Services (ICWS), 2014 IEEE International Conference on (pp. 598-605). IEEE.
15. Mihajlovic, Z., & Velasevic, D. (2001). Tracking software projects with the integrated version control in SMIT. ACM SIGSOFT Software Engineering Notes, 26(2), 38-43.
16. Munns, A. K., & Bjeirmi, B. F. (1996). The role of project management in achieving project success. International journal of project management, 14(2), 81-87.
17. Cimiano, Philipp, Andreas Hotho, and Steffen Staab. "Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis." J. Artif. Intell. Res.(JAIR) 24 (2005): 305-339.
18. Seiler, M., & Paech, B. (2017, February). Using Tags to Support Feature Management Across Issue Tracking Systems and Version Control Systems. In International Working Conference on Requirements Engineering: Foundation for Software Quality (pp. 174-180). Springer, Cham.
19. Farré, C., Franch, X., & Ionescu, T. (2017). State of the Practice on Software Release Planning.