Context-Based Project Management

Ammar Alsaig[1,2(✉)], Alaa Alsaig[1,2], and Mubarak Mohammad[1,2]

[1] Concordia University, Montreal, Canada
aasaig@uqu.edu.sa, mubarak.sami@gmail.com
[2] Jeddah University, Jeddah, Kingdom of Saudi Arabia
aalsaig@uj.edu.sa

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 system · Model · Formal concept analysis FCA · Lattice tree

1 Introduction

Projects and project management are part of every organization or company lives. Since businesses are evolving the requirements related to project management is increasing. That is, project management is not only about time management, team communication, or task tracking, there are, however, other requirements are needed to enhance the management of projects such as task correlation and dynamic update when a change occurs. The context of projects includes important information to enhance accuracy and efficiency of software management tools. By knowing the context of a specific task, the estimation of needed time to complete a specific task and other related tasks will be clearer to the team. Hence, better management and results. The Context-awareness and context-based applications is one of the essential method used recently to enhance most the current systems. Context-awareness plays an important role in Ubiquitous computing [4], cloud computing [5], mobile applications [6], cyber physical systems [7] and others.

P. Cong Vinh et al. (Eds.): ICCASA 2016, LNICST 193, pp. 12–21, 2017.
DOI: 10.1007/978-3-319-56357-2_2

Therefore, 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 using formal data modeling.

There are two essential components of project management: service requester (client), who define 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 vendors, 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 vendors. We refer to client requirements and vendor data as *Project Context*. The high dependency of project context components is illustrated in (Fig. 1). The balance between different components of project management depends on how well project managers are aware of the relationships between different data components within project context. Project managers often need to reconsider their decision regarding the trade-offs they need to commit during the development. Thus, questions like, what's the effect of an absence of certain resources assigned to a project? What are the relations between resources and risks? Or what are relations between tasks and risks? are examples of many other questions that need ready answers for project managers to make their decisions.

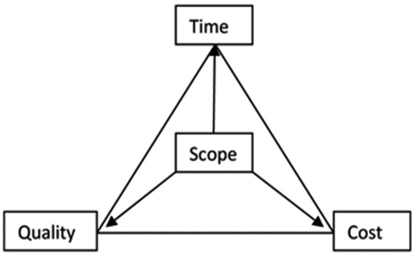


Fig. 1. Main components of projects

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 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, risk management. etc. These tools work as an organizer of project data. However, when it comes to relationships within the data 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*". In the same reference, the suggested solution for this problem should be "*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 management context and keep track of the emerging relationships between its main components. We start from categorizing and classifying project data, formally modeling the data using FCA-lattice, and providing a full implementation of our approach.

2 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 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 definition for the methods used to define the relationship is not provided. This keeps the need to provide a definition for how to build relationship formally. Moreover, defining a class object in project management software that build the relationship among tasks is advantageous. This is because it gives the ability to define relationships among other objects or classes of the software.

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 relationships are built using theoretical formula (FCA) that does build the connections among object based on context information.

3 Data of Project Management

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

(A) Relations Context: 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 what risks are related to it, is defined in the Relations Context. The explanation of relationships among project data is out of the scope of this paper. Interested readers can refer to the following few examples of project management references [1–3] for more information.

(B) Client Context: 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 vendor, which means service vendor can trade-off on this open specification. For example, if the time-frame is left open for service vendor, service vendor can enhance quality, and minimize cost by extending the period of development and by assigning less resources to work on the project.

(C) Vendor Context: data that is defined by project manager of service Vendors. It includes the expected times of deliverables, the resources assigned on the project, 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.

4 Modeling Data

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

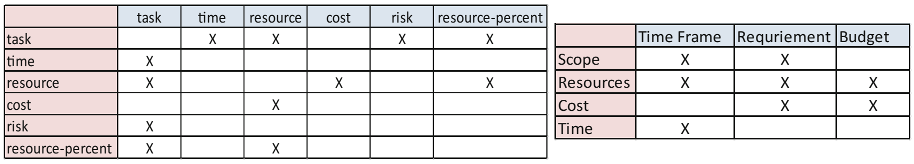
4.1 Formal Concept Analysis (FCA)

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 in the model is FCA lattice. That is, all types of relations are represented as FCA lattice. This means that all data entities we have in our model will be based on objects and attributes.

4.2 Relations Context

This type of data defines the different relationships between the data entities of both Vendor Context and Client Context. The relationships in Relations Context are illustrated in (Fig. 2(A), (B)). The relations defined in the Figure are not exhaustive, many other components can appear in both Vendor and Client Context. Nonetheless, all information can be represented as shown in (Figs. 2 and 3). These relations are defined as FCA lattice. Thus, some relations can be inferred. For example, there is a relationship between resources and risks through requirements. The explanation of how this data is used in the system will be discussed in the implementation section.



	task	time	resource	cost	risk	resource-percent
task		X	X		X	X
time	X					
resource	X			X		X
cost			X			
risk	X					
resource-percent	X		X			

(A)

	Time Frame	Requriement	Budget
Scope	X	X	
Resources	X	X	X
Cost		X	X
Time	X		

(B)

Fig. 2. (A) Relations context for vendor context, (B) Relations context for vendor/client contexts

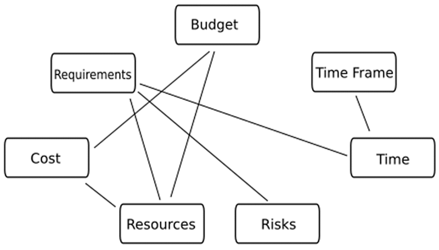


Fig. 3. Example of relations context relations between provider context

4.3 Client Context

Client Context is defined by the service requester. It includes all client requirement and specifications for the project. This includes but not limited to, time frame, budget, requirements and any other specifications regarding the project. In our model, Client Context consist of data entities. For example, time frame is an entity, budget is a different entity, and requirements list is another entity and so on. The relationship between those entities is defined in the Relations Context.

4.4 Vendor Context

Vendor Context is similar to Client Context. However, it is defined by the project executing team. In fact, it is the responsibility of project managers. The entities in this context includes but not limited to, project plan, cost, and resources…etc. Like Client Context, the interrelations between different data entities are defined in the Relations Context.

5 Implementation Tools

The main theoretical method on which our implementation is based is the 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.

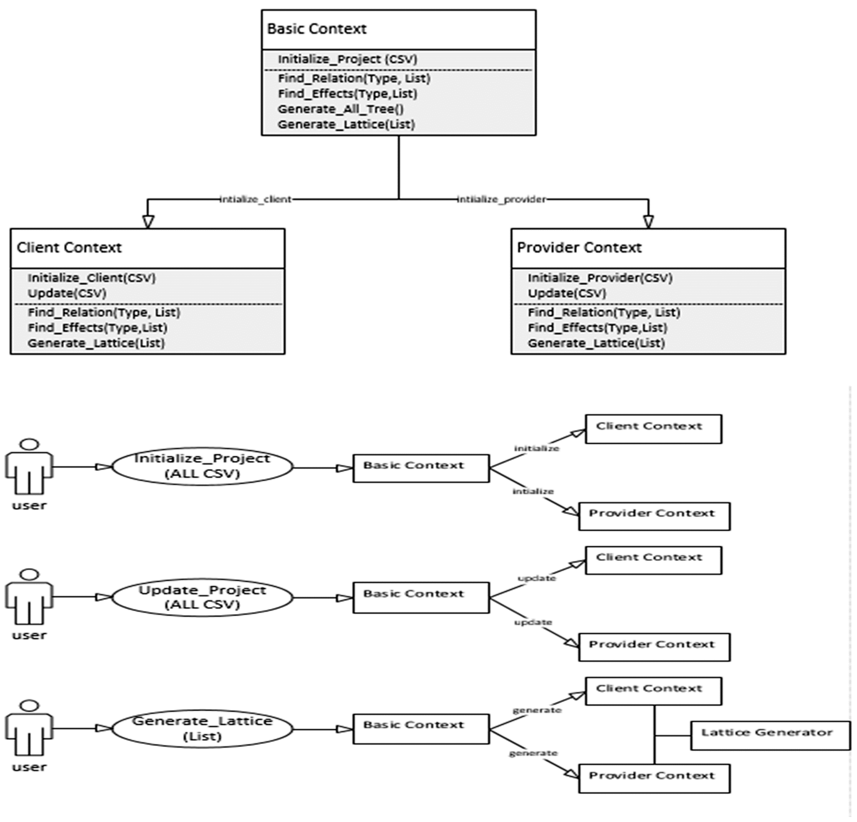


Fig. 4. Main classes in our implementation and general use cases

5.1 FCA Functions

The two formal functions used to find relations between objects and attributes are intention and extension functions. These two functions are formally defined as:

Extent and Intent Formula
Concepts Extent = Ext (X, Y, I) = {A ∈ 2x | (A, B) ∈ B (X, Y, I) for some B}
Concepts Intent = Int (X, Y, I) = {B ∈ 2y | (A, B) ∈ B (X, Y, I) for some A} [6]

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 with four resources, the extent function on that tasks gives back the four resources, while the intent of the four resources returns back this task 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.

5.2 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 4 shows the basic classes in our implementation.

5.2.1 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

5.2.2 Customized Functions

The following gives brief information about the functions and libraries that we have implemented (Table 1).

5.2.3 Case Study and Test Cases

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

Table 1. Implemented information of function and libraries

Library/function	Details

Initialize_Project(CSV)
Initialize_Client(CSV)
Initilaize_Vendor(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

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 5 shows full lattice tree for the project plan.

Test Case: Find all risks associated with one resource.

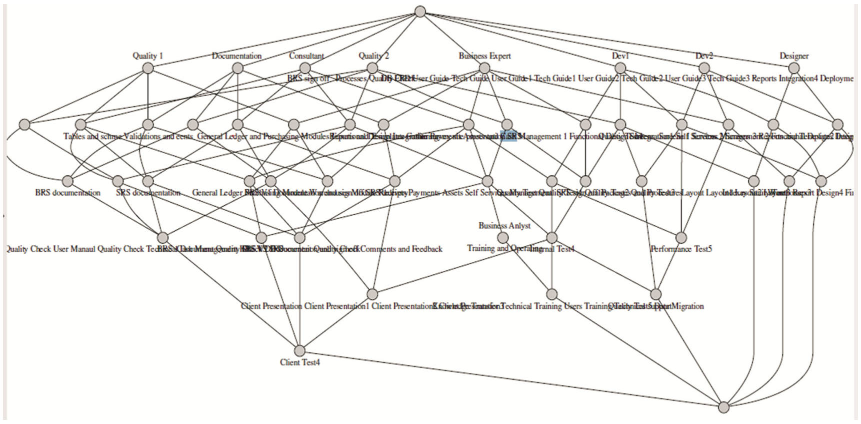


Fig. 5. Case study, FCA lattice generated from our program

– 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 Vendor 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.

6 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 2.

Table 2. Context-based approach VS. Traditional tools

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

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.

References

1. Lewis, J.P.: Project Planning, Scheduling & Control, 4E. McGraw Hill, New York (2005). ISBN 978-0-07-146037-8
2. Newell, M.W., Grashina, M.N.: The Project Management Question and Answer Book, p. 8 (2004)
3. PMBOK, Third Edn., p. 165 (2004)
4. Weiser, M.: Some computer science issues in ubiquitous computing. Commun. ACM **36**(7), 75–84 (1993)
5. Mell, P., Grance, T.: The NIST definition of cloud computing (2011)
6. Charland, A., Leroux, B.: Mobile application development: web vs. native. Commun. ACM **54**(5), 49–53 (2011)
7. Baheti, R., Gill, H.: Cyber-physical systems. In: The Impact of Control Technology, vol. 12, pp. 161–166 (2011)
8. Lowery, G.: Managing Projects With Microsoft Project 4.0: For Windows and MacIntosh. Wiley, New York (1997)
9. Welti, N.: Successful SAP R/3 Implementation: Practical Management of ERP Projects. Addison-Wesley Longman Publishing Co. Inc., Boston (1999)
10. Lang, J., Davis, E.: Redmine-open source project management web-application (2010)
11. Schiff, J.L.: 12 common project management mistakes–and how to avoid them, 26 September 2012. http://www.cio.com/article/2391872/project-management/12-common-project-management-mistakes–and-how-to-avoid-them.html. Accessed from CIO
12. Mishra, A., Mishra, D.: Software project management tools: a brief comparative view. ACM SIGSOFT Softw. Eng. Notes **38**(3), 1–4 (2013)
13. Angioni, M., Carboni, D., Melis, M., Pinna, S., Sanna, R., Soro, A.: XPSuite: tracking and managing XP projects in the IDE. In: Proceedings of the 2004 Workshop on Quantitative Techniques for Software Agile Process, pp. 46–52. ACM, November 2004
14. Aoyama, M., Yabuta, K., Kamimura, T., Inomata, S., Chiba, T., Niwa, T., Sakata, K.: A resource-oriented services platform for managing software supply chains and its experience. In: 2014 IEEE International Conference on Web Services (ICWS), pp. 598–605. IEEE, June 2014
15. Mihajlovic, Z., Velasevic, D.: Tracking software projects with the integrated version control in SMIT. ACM SIGSOFT Softw. Eng. Notes **26**(2), 38–43 (2001)
16. Munns, A.K., Bjeirmi, B.F.: The role of project management in achieving project success. Int. J. Project Manage. **14**(2), 81–87 (1996)
17. Cimiano, P., Hotho, A., Staab, S.: Learning concept hierarchies from text corpora using formal concept analysis. J. Artif. Intell. Res. (JAIR) **24**, 305–339 (2005)