Explaining Product Release Planning Results Using Concept Analysis

Gengshen Du, Thomas Zimmermann, Guenther Ruhe
Department of Computer Science, University of Calgary
2500 University Drive NW, Calgary, Alberta T2N 1N4, Canada
{dug, zimmerth, ruhe}@cpsc.ucalgary.ca

Abstract

Objective: This paper aims to generate explanations from a series of data points obtained from a decision support system called ReleasePlanner® for supporting product release planning and considered to be a black box.
Method: Concept analysis is applied to 1085 data points received from running 10 scenarios of a real world product release planning project with 35 candidate solutions generated by ReleasePlanner®.
Results: Three main results are obtained: (1) patterns between inputs and outputs; (2) impact of individual input parameters on outputs; and (3) sensitivity level of outputs in dependence of inputs.
Conclusion: Concept analysis is shown to be a feasible technique for gaining more insight into the structure of results obtained from a black box input-output system, such as, but not limited to, ReleasePlanner®.

Keywords

Explanations, concept analysis, product release planning

1 Introduction

Product release planning involves decision making on assigning features to different releases for incremental software product development. It must simultaneously consider several aspects, such as conflicting stakeholder priorities and objectives, feature interdependencies, and resource and risk constraints [15]. A decision support system called ReleasePlanner® [13] has been developed to support decision makers in the complex release planning process. It is based on computationally efficient optimization algorithms for the generation of a set of alternatives solution having a proven degree of optimality.

However, the findings from a series of experiments conducted with ReleasePlanner® users revealed that they were reluctant to accept the solutions advised by this tool [6]. Similar observation has also been made on other systems [4] [10]. It was concluded in [1] that the major problems are not technical problems, but people problems in which people have very limited understanding on the support they get from decision support systems. In addition, according to [2], product release planning problem is classified as a wicked problem [14] which is hard to be precisely formulated. Thus the procedure needed to solve product release planning problems (as demonstrated in ReleasePlanner®) is more complex and requires more in depth explanations to achieve good user understanding on the tool support and its solutions.

How can we facilitate better understanding of the ReleasePlanner® solutions? In this paper, a data analysis technique called concept analysis [11] is applied for this purpose. It is applied to investigate data within a specific product release planning problem and identifies hidden relationships between the project inputs and outputs. In particular, three types of relationships are analyzed:

- Patterns between the input and output attributes
- Impact of individual input attributes on the outputs
- Sensitivity level of the outputs to the inputs

The answers to these three research questions provide additional knowledge that is currently unavailable to users of the ReleasePlanner® system. As a result, the users' acceptance and trust level on the tool and its solutions is expected to be improved.

The remainder of this paper is organized as the follows. Section 2 gives an overview of product release planning and the related decision support tool ReleasePlanner®. Section 3 introduces the background of concept analysis. In Section 4, a sample product release planning project is investigated to demonstrate the application of concept analysis. Section 5 analyzes and interprets the results in the context of the three stated questions. Finally, Section 6 summarizes the research and outlines future research.

2 Product Release Planning

Many formal approaches have been proposed for product release planning, such as incremental funding method [5], cost-value approach [9], planning software evolution with risk management [8], and hybrid intelligence (EVOLVE*) [15]. The latter is used in this paper. This section gives a short overview of this approach to the extent necessary to understand and judge the results obtained from concept analysis presented later in this paper. More details on the method are available from [15].

2.1 Technical Formulation

In incremental product development, the goal of product release planning is to select from a set of features $F = \{f_1,\ldots,f_n\}$ and to assign them to one of K possible releases each of them having a weight (relative importance) of ξ_k ($k = 1\ldots K$). A release plan is described by vector x with

decision variables $\{x(1),\ldots,x(n)\}$, where $x(i) = k$ if feature f_i is assigned to release option $k \in \{1,\ldots,K\}$; and $x(i) = K+1$ otherwise (i.e. the feature is postponed).

Two types of feature dependencies are considered: coupling and precedence relationship. A coupling $CC(f_i, f_j)$ indicates that both features f_i and f_j must be released jointly. A precedence $PC(f_i, f_j)$ indicates that feature f_i cannot be released later than f_j. Some features can be fixed to certain release by the pre-assignment $preassign\text{-}x(i)=k$, indicating that f_i is fixed to release k.

The planning approach considers T resource types for implementing the features. Capacities $Cap(k, t)$ relate each release k to each resource type $t \in \{1,\ldots,T\}$. Every feature f_i requires an amount of resources of type t $r(f_i, t)$. Thus, each release plan x assigns feature f_i to release k expressed as $x(i) = k$, for all releases k and resource types t, must satisfy $\Sigma_{x(i)=k}\, r(f_i, t) \le Cap(k, t)$.

A set of stakeholders $S = \{s_1,\ldots,s_q\}$ is involved in release planning. Each of them has a relative importance $\lambda \in \{\lambda_1,\ldots,\lambda_q\}$. It is a nine-point ordinal scale that provides differentiation in the degree of importance. The higher the importance value is, the more important the stakeholder is.

In brief, the purpose of release planning is to provide the most attractive features at the earliest releases to the most important stakeholders. For the purpose of this paper, $Value(s, f_i)$, $Urgency(s, f_i)$, and $Competitiveness(s, f_i)$ are the three attributes of a feature's attractiveness. Each feature can be prioritized from these three criteria with the value ranging from 0 to 9. These criteria are associated with the weights μ_1, μ_2, and μ_3, respectively.

The three prioritization criteria are the basis to formulate the objective of product release planning. The objective function $Utility(x)$ is defined as a linear combination of the priority votes of stakeholders related to these criteria:

$$Utility(x) = \sum_{k=1}^{K} (\xi_k \times \sum_{i:x(i)=k} \Pr iority(f_i))$$

where $Priority(f_i)$ is an aggregated priority of f_i defined as:

$$\Pr iority(f_i) = \sum_{s=s_1}^{S_q} (\lambda_s \times (\mu_1 \times Value(s, f_i) + \mu_2 \times Urgency(s, f_i) + \mu_3 \times Competitiveness(s, f_i)))$$

2.2 ReleasePlanner®

ReleasePlanner® [13] is a decision support system that aims at performing systematic product release planning based on computationally efficient optimization algorithms. Users are able to perform what-if analysis to pro-actively explore different scenarios defined by a sequence of inputs of the same project under investigation. In addition, the tool is capable of generating a set of diversified solution alternatives for each instance.

A series of studies on ReleasePlanner® revealed that its users tended to have higher confidence and trust on their manual solutions than the ones generated more efficiently by the tool [6]. The major reason is that the tool works in a black box manner and the rationale of solution generation is hard to understand by the users. This is even more complicated because the users usually investigate multiple scenarios with multiple solutions.

3 Concept Analysis

Concept analysis, firstly introduced in [17], is a theory of data analysis to identify conceptual structures among a set of data. It has been successfully applied to many fields [12], including in software engineering [11].

In this paper, concept analysis is investigated to address the three research questions presented in Section 1. Another two techniques, i.e. rough set analysis and dependency network analysis, have also been applied to explain release planning solutions by ReleasePlanner® [7]. However, they can only deal with the first two research questions and are not the focus of this paper. Detailed applications of these two techniques are available at [7].

3.1 Basic Terminology

Concept analysis investigates the relations R between a set of objects O, and a set of attributes A. The triple $C = (R, O, A)$ is called a formal context.

Def. 1 (Common Attributes and Common Objects): For any set of objects $\mathcal{O} \subseteq O$, the set of common attributes having the same attribute value is called common attributes related to $\mathcal{O}$ and is denoted by $ca(\mathcal{O}) = \{a \in A \mid \forall\, o \in \mathcal{O} : (o, a) \in R\}$. For a set of attributes $\mathcal{A} \subseteq A$, their common objects are $co(\mathcal{A}) = \{o \in \mathcal{O} \mid \forall\, a \in \mathcal{A} : (o, a) \in R\}$.

Def. 2 (Formal Concept): Each pair $c = (\mathcal{O}, \mathcal{A})$ with $\mathcal{O} = co(\mathcal{A})$ and $\mathcal{A} = ca(\mathcal{O})$ is called a formal concept. It demonstrates a pattern, i.e. relation, between $\mathcal{O}$ and $\mathcal{A}$.

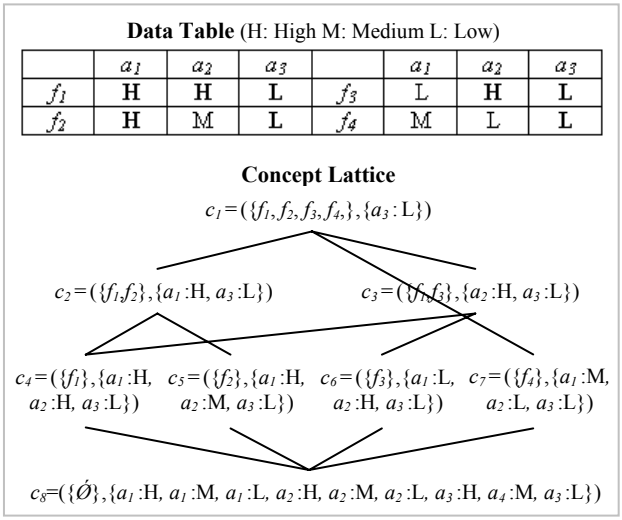
Def. 3 (Concept Lattice): All formal concepts for a given context C are called a complete concept lattice in which concepts can be partially ordered. If $c_1 = (\mathcal{O}_1, \mathcal{A}_1)$ and $c_2 = (\mathcal{O}_2, \mathcal{A}_2)$ are two concepts in the context C, a partial order $c_1 \le c_2$ can be defined whenever $\mathcal{O}_1 \subseteq \mathcal{O}_2$ and $\mathcal{A}_1 \supseteq \mathcal{A}_2$.

Def. 4 (Greatest Lower Bound and Least Upper Bound): The greatest lower bound of c_1 and c_2 is the concept with objects $\mathcal{O}_1 \cap \mathcal{O}_2$ and attributes held by all objects in $\mathcal{O}_1 \cap \mathcal{O}_2$. The least upper bound of c_1 and c_2 is the concept with attributes $\mathcal{A}_1 \cap \mathcal{A}_2$ and objects which have all attributes in $\mathcal{A}_1 \cap \mathcal{A}_2$.

3.2 An Illustrate Example

Applied to planning product releases, O constitutes the set of features F to be assigned to different releases. The input to and output from product release planning using ReleasePlanner® form set A. Figure 1 shows a simple example of concept analysis in this domain. The upper part is a data table of feature set $F = \{f_1,\ldots,f_4\}$ defined with the attribute set $A = \{a_1,\ldots,a_3\}$. In this table, the value of each attribute for each feature can be H, M, or L.

These values represent different value ranges. The lower part of this figure shows the corresponding concept lattice with all the concepts $\{c_1,\ldots,c_8\}$ and their order relations. In this lattice, the values of the attributes in each concept are also highlighted. Among all the concepts, c_1 is the most general one and c_8 is the most specific one. Some of the order relations among the concepts are $c_2 \leq c_1$, $c_3 \leq c_1$, and $c_7 \leq c_1$. From this figure, we can also identify the least upper bound and greatest lower bound of a set of concepts. For example, c_1 and c_4 are the least upper bound and greatest lower bound of c_2 and c_3, respectively.



	a_1	a_2	a_3		a_1	a_2	a_3
f_1	H	H	L	f_3	L	H	L
f_2	H	M	L	f_4	M	L	L

Figure 1: Example concept analysis

4 Applying Concept Analysis to Explain Product Release Planning Results

4.1 Sample Project

To illustrate the application of concept analysis to explain results generated by ReleasePlanner®, we investigate on a sample project based on the data from a real life product release planning problem. As a summary, this project is defined with the following inputs:

- $F = 31$ features $\{f_1, \ldots, f_{31}\}$ to be assigned
- $K = 2$ releases
- $S = 18$ stakeholders $\{s_1,\ldots,s_{18}\}$ with weights $\{\lambda_1,\ldots, \lambda_{18}\}$
- Prioritization criteria $Urgency(s,f_i)$, $Value(s,f_i)$, and $Competitiveness(s,f_i)$
- $T = 3$ types of resources $\{Res1, Res2, \text{and } Res3\}$

The full details of this project setting can be referred to http://pages.cpsc.ucalgary.ca/~dug/ConceptAnalysis. This project setting and the results obtained from it are taken as the baseline scenario. From this baseline, the tool user also generates another 9 scenarios that the user thinks to be the most important (but not necessarily the complete) scenarios for investigation. Together these 10 scenarios are used for what-if analysis which is useful and important for product release planning, as discussed in Section 2.2. For all these scenarios, in total 35 solutions are generated by ReleasePlanner® for later analysis.

4.2 Data for Concept Analysis

With the above project settings, each feature f_i in each solution is associated with a data point used for concept analysis (see Table 1). The selection of the attributes is based on the experience of manual analysis of several product release planning projects [15]. The first six input attributes are relevant to stakeholder votes which are considered in the objective function for planning. In particular, $ConfUrgency(f_i)$, $ConfValue(f_i)$, and $ConfComp(f_i)$ are the standard deviation between different stakeholders' votes for each feature f_i from the three criteria, respectively. They indicate the degree of disagreement among stakeholder opinions. The other six input attributes address resource utilization and criticality of features.

Table 1: Data defined for concept analysis

Input Attribute	Definition
$AverageUrgency(f_i)$	$AverageUrgency(f_i) = \frac{\sum_{S=S1}^{S18} \lambda_S \times Urgency(s, f_i)}{\sum_{S=S1}^{S18} \lambda_S}$
$AverageValue(f_i)$	(similar for $AverageValue(f_i)$, $AverageComp(f_i)$)
$AverageComp(f_i)$	
$RelConfUrgency(f_i)$	$\mathrm{Re}lConfUrgency(f_i) = \frac{ConfUrgency(f_i)}{AverageUrgency(f_i)}$
$RelConfValue(f_i)$	(similar for $RelConfValue(f_i)$, $RelConfComp(f_i)$)
$RelConfComp(f_i)$	
$Res_tUtiRatio(f_i)$ (t = 1, 2, 3)	$\mathrm{Re}s_tUtiRatio(f_i) = \frac{r(f_i, \mathrm{Re}s_t)}{\sum_{i=1}^{31} r(f_i, \mathrm{Re}s_t)}$
$Res_tCriticality(f_i)$ (t = 1, 2, 3)	$\mathrm{Re}s_tCriticality(f_i) = \mathrm{Re}s_tUtiRatio(f_i) - \frac{r(f_i, \mathrm{Re}s_t)}{\sum_{k=1}^{2} Cap(k, \mathrm{Re}s_t)}$
Output Attribute	**Definition**
$Release(f_i)$	$Release(f_i)$

Based on our previous experience on the analysis of these attributes, each attribute is discretized according to Table 2. The purpose of discretization is to scale attributes with continuous values to a nominal or ordinal scales.

Table 2: Discretization of the defined attributes

Input Attribute	Value Range	Discretization	
$AverageUrgency(f_i)$	A real number in [0, 9]	High	[6, 9]
$AverageValue(f_i)$		Medium	[4, 6)
$AverageComp(f_i)$		Low	[0, 4)
$RelConfUrgency(f_i)$	A real number in [0, 1]	High	[0.7, 1.0]
$RelConfValue(f_i)$		Medium	[0.4, 0.7)
$RelConfComp(f_i)$		Low	[0.0, 0.4)
$Res_tUtiRatio(f_i)$	A real number in [0, 1]	High	[0.10, 1.00]
		Medium	[0.05, 0.10)
		Low	[0.00, 0.05)
$Res_tCriticality(f_i)$	A real number in [-1, 1]	No	[0.00, 1.00]
		Low	(-0.01, 0.00)
		Medium	[-1.00, -0.01]
Output Attribute	**Value Range**	**Discretization**	
$Release(f_i)$	An integer in [1, 3]	Not necessary	

Based on the above definition and discretization, a table with 1085 data points (35 solutions with each containing 31 features) is obtained for later concept analysis. The complete table is available at the website provided earlier.

4.3 Concept Analysis of the Data

We used an open source library called Colibri/Java [3] to perform concept analysis. It builds a concept lattice which contains all patterns (concepts) for the data prepared in Section 4.2. We then implemented a tool to traverse the concept lattice to select only those patterns where the distribution of the output values significantly changed along the (subset) relations between these patterns. To test significance, we used Fisher Exact Value and Chi Square tests (significance level of p=0.01) [16]. Using our tool, two filtered lattices were generated that contain the patterns which affect the distribution of releases the most:

- Concept lattice #1 contains 64 patterns where the likelihood of assigning a feature f_i to release 1 is increased by at least 45%.
- Concept lattice #2 contains 1093 patterns where the likelihood of assigning a feature f_i to any release, i.e. 1, 2 or 3 (postponed), is increased or decreased by at least 30%.

The first lattice is essentially a part of the second one. The details of these lattices are available at the website given earlier and will be analyzed in depth in Section 5.

Table 3: Example record in the generated concept lattices

Context			*Res3Criticality(f_i)*_Low		
Var			*AverageComp(f_i)*_High		
Δ_R1	0.49	**Context_R1**	548	**Context+Var_R1**	118
Δ_R2	-0.3	**Context_R2**	339	**Context+Var_R2**	1
Δ_R3	-0.18	**Context_R3**	198	**Context+Var_R3**	0

Each filtered lattice consists of a number of patterns and transitions between these patterns in the form shown in Table 3. This table is read as, for all the 1085 cases in the dataset, the distribution of features f_i following the pattern of "*Res3Criticality(fi)* = Low" ("context" part) is 548 data points for release 1 ("context _R1"); 339 for release 2 ("context_R2"); and 198 for release 3 ("context_R3"). The pattern of "*Res3Criticality(fi)* = Low AND *Average Comp(fi)* = High" ("context" and "var") is supported by 118 data points for release 1 ("context+var_R1"); 1 for release 2 ("context+var_R2"); and 0 for release 3 ("context+var_R3"). The transition between these two patterns can be understood as a rule: adding "*Average Comp(fi)* = High" ("var") to the "context" part increases the likelihood of assigning a feature f_i to release 1 by 49% ("Δ_R1"), and decreases the likelihood to release 2 and 3 by 30% ("Δ_R2") and 18% ("Δ_R3"), respectively.

5 Analysis and Interpretation of Results

In this section, we analyze the two lattices generated in Section 4.3 from three perspectives: similarity of patterns, importance of input attributes, and sensitivity of outputs. The findings from these aspects contain new knowledge that reveals the underlying relationships between the inputs to ReleasePlanner® and its outputs, for the studied sample project. They can be used as explanations for this decision support system and its solutions.

5.1 Pattern Transitions and Data Similarity

Each generated concept lattice covers the most significant patterns discovered from the product release planning data used for concept analysis. These patterns are presented in the "context" part and of the different granularities, i.e. from the most general to the most specific. A general pattern can be transformed to more specific ones, and vice versa. By examining the generalization or specification relationships among these patterns, the transitions among the patterns become visible. In addition, the discovered patterns demonstrate the similarities shared among the data used for analysis. Data that are categorized under a same pattern are of the similarity as demonstrated by the pattern. For any two patterns that can be generalized to the same more general pattern, the two data sets supporting these patterns must be similar to each other in the way that is represented from the general pattern.

To illustrate the pattern transition and data similarity in this sample project, the concept lattice #1 is analyzed for simplicity. Any other lattices can be analyzed similarly.

Figure 2 shows the top four levels of patterns within this concept lattice and the transitions of these patterns. The complete transitions of all the patterns in this lattice can be referred to the website provided earlier. In this lattice, the most general pattern is #1, as shown in the very top of the figure. It can be specified to pattern #2, #3, and #4 at the second level. In this case, we say pattern #1 is the generalization of pattern #2, #3, and #4. On the other hand, pattern #2, #3, and #4 are the three specifications of pattern #1. Each of these three patterns can be further specified to other patterns until no more specific pattern can be found. For example, one of the most specific patterns is pattern #60. It follows the specification path of pattern #1→#3→#6→#41→#50→#60.

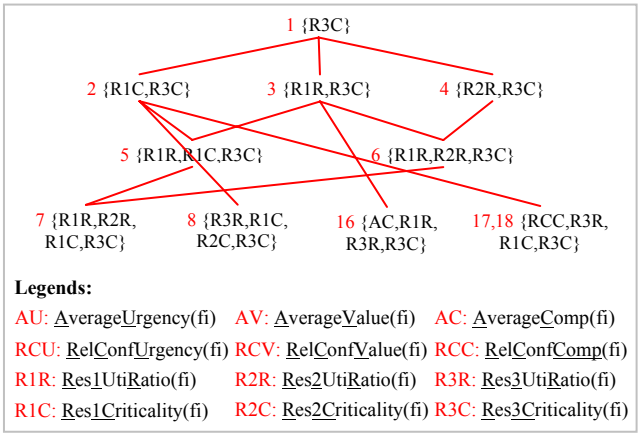


Figure 2: Transition of patterns (concept lattice #1)

Regarding the similarities shared among all the 1085 data used for the analysis, all these data are the same in terms of their values on the input R3C (*Res3Criticality(f_i)*), as illustrated through the most general pattern, i.e. pattern #1. More similarity is discovered from the data #1 to #715 and #869 to #930 because these data points also share the same value on the input attribute R1C (*Res1Criticality(f_i)*), besides on R3C. As a result, these data form a more specific pattern, i.e. pattern #2.

These results can be interpreted as a type of explanations on ReleasePlanner® solutions. If, in a solution, the release assignment of a feature is supported by general pattern(s) that are supported by a large number of data points, the users are more likely to accept such result. Otherwise, they might want to further improve the solution.

5.2 Importance Level of Inputs on Outputs

By examining all the found patterns ("context" part), we can identify each input attribute's importance level to the output attribute, in our case the release. The assumption is that the higher the number of the occurrence of an input in the patterns is, the more important this input is in determining the release value. However, an exception to this assumption is that this number cannot be as high as the total number of data used for analysis. The rational for this exception is given later.

For this purpose, we investigate the second concept lattice which provides more coverage than the first one on the patterns inherent in the data. Figure 3 summarizes the number of occurrence of each input in this concept lattice. *Res3Criticality(f_i)* appears to be the most important attribute. It is in all the patterns and has the same value. In other words, it has no influence at all on the distribution of release. *Res1Criticality(fi)* and *Res1UtiRatio(fi)* are important attributes which have different values. The least important ones are *AverageComp(fi)*, *AverageUrgency(fi)*, and *RelConfComp(fi)*. Other input attributes have medium level of importance.

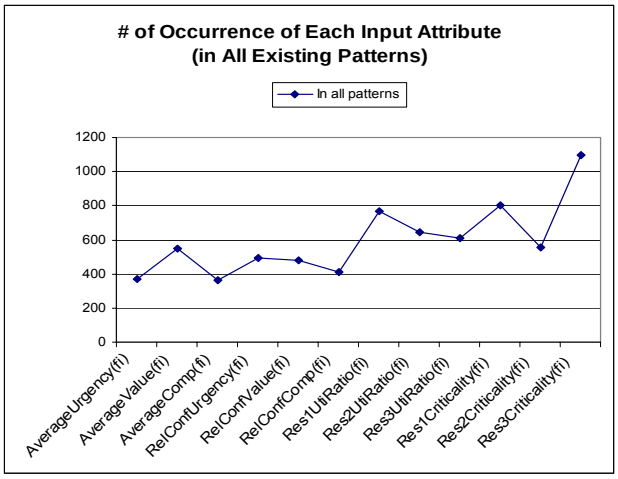


Figure 3: Importance level of each input on the output (concept lattice #2)

This part of the results provides the ReleasePlanner® users with the explanations by identifying a subset of all the defined inputs that play the most significant impacts on the tool when it generates solutions.

5.3 Sensitivity Level of Outputs to Inputs

The generated concept lattices also check if adding a new input attribute ("var" part) to the existing patterns ("context" part) would change the distribution of release. If the change is significant, this input is likely responsible for such change, i.e. the output is sensitive to this input. To observe the sensitivity of the output on each input, the second concept lattice is used for analysis again. In particular, we examine six types of how the "var" part may impact on the distribution of releases:

- R1/R2/R3 +0.30: increase by at least 30% in the distribution of assigning a feature f_i to release 1, 2, and 3, respectively
- R1/R2/R3 -0.30: decrease by at least 30% in the distribution of assigning a feature f_i to release 1, 2, and 3, respectively

For each input, we first count its number of occurrence in the "var" part of each record. For example, in the record in Table 3, the input attribute *AverageComp(f_i)* is in the "var" part with 118 cases supporting the impact of R1 +0.30. Therefore its number of occurrence in this record, for this type of impact, is 118. Then, by summing up such numbers for all the records of the same impact type, we obtain the total number of occurrence of this input. Figure 4 shows this number for each input based on the above calculation. We assume that the higher this number is, the more sensitive the output is to this input.

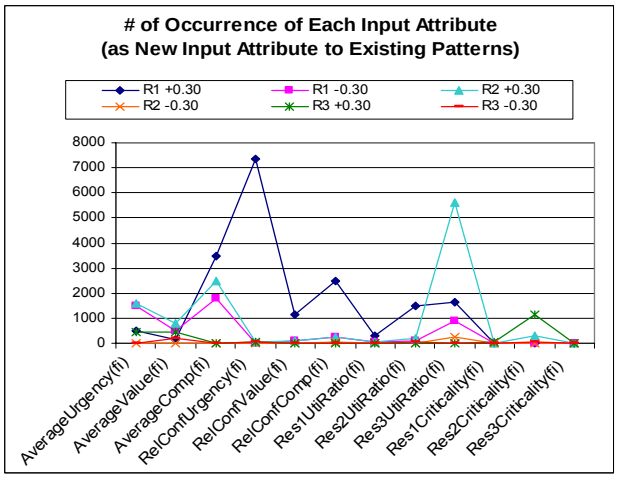


Figure 4: Sensitivity level of the output to each input (concept lattice #2)

From this figure, *RelConfUrgency(fi)*, *Res3UtiRatio(fi)* and *AverageComp(fi)* have the most significant impacts on the distribution of release. Specifically, release 1 is most sensitive to *RelConfUrgency(fi)* and *AverageComp(fi)* for at least 30% of increased distribution, and to *AverageComp(fi)* for at least 30% of

decreased distribution. Release 2 is most sensitive to *Res3UtiRatio(fi)* and *AverageComp(fi)* for at least 30% of increased distribution. But they usually have no impact as R2 -0.30 or R3 ±0.30. On the other hand, *Res1UtiRatio(fi)*, *Res1Criticality(fi)*, and *Res3Criticality(fi)* almost never contribute to any change by at least ±30% in any release. Although they occur the highest times in the patterns in Figure 3, their sensitivity levels are not as significant as at least ±30% and cannot be reflected in Figure 4. The other input attributes in general have medium level of sensitivity on the output attribute.

The above results explain some sensitivity aspects of the solutions generated by ReleasePlanner®. This kind of knowledge reveals the degree of impact from changing different input parameters. In case of uncertain data, the rule of thumb is that the more robust a solution, the higher chance of acceptance by the user.

6 Conclusions and Future Work

In this paper, a formal data analysis method called concept analysis is combined with statistical hypothesis testing to explain complex solutions recommended by ReleasePlanner®, a decision support system for product release planning. The results of our analysis of the data of individual release planning projects contain additional knowledge that is currently unavailable to the tool users. Specifically, such knowledge explains the underlying relationships inherent in the investigated data, in terms of the underlying patterns between the input and output data, as well as the importance and sensitivity levels of inputs on outputs. These explanations intend to achieve better understanding on the solutions of ReleasePlanner®, and therefore higher acceptance level from the user side. To demonstrate the application of concept analysis and statistical hypothesis testing, a sample product release planning project was investigated. Although the findings presented in this paper are specific to the sample project, the methodology of applying such analysis is generic (since it treats the decision support system as a black box) and can be applied to any other product release planning projects, or other software systems in which explaining complex solutions to users is necessary.

As a very important future work, empirical studies will be conducted with ReleasePlanner® users in order to justify the usefulness and effectiveness of the proposed method for explaining the tool's solutions. In addition, the results obtained from the concept analysis, as presented in this paper, only provides one type of explanations on ReleasePlanner® solutions and it is by no means complete. The explanations generated from this method are better to be used with other types of explanations (e.g. the ones discussed in [7]) that address the solutions from different aspects. Therefore we will also investigate on how these different types of explanations obtained from different techniques are complimentary to each other so that they can together provide the tool users with a more complete view of explanations on the tool and its solutions.

Acknowledgement

The authors would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Alberta Informatics Circle of Research Excellence (iCORE) for their financial support of this research. Many thanks are due to Daniel Götzmann and Christian Lindig who provided the Colibri/Java implementation.

References

[1] M. J. A. Berry, G. S. Linoff: Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management (2nd Edition), Wiley, 2004.

[2] P. Carlshamre: Release Planning in Market-Driven Software Product Development: Provoking an Understanding. Journal of Requirements Engineering, Vol. 7, 2002, pp. 139-151.

[3] Colibri/Java, available at http://code.google.com/p/ colibri-java/, last accessed February 2008.

[4] F. Davis, J. Kottemann: Determinants of Decision Rule Use in a Production Planning Task. Journal of Organizational Behavior and Human Decision Processes, Vol. 63, No. 2, 1995 pp. 145-157.

[5] M. Denne, J. Cleland-Huang: The Incremental Funding Method: Data Driven Software Development. IEEE Software, Vol. 21, No. 3, 2004, pp. 39-47.

[6] G. Du, J. McElroy, G. Ruhe: A Family of Empirical Studies to Compare Informal and Optimization-based Planning of Software Releases. Proceedings of the 5th International Symposium on Empirical Software Engineering, Rio de Janeiro, Brazil, 2006, pp. 212-221.

[7] G. Du, G. Ruhe: Comparison of Two Machine Learning Techniques for Explaining Results in Product Release Planning. Submitted to Journal of Information Sciences, Special Issue on Applications of Computational Intelligence and Machine Learning to Software Engineering, 2008, 36 pages.

[8] D. Greer: Decision Support for Planning Software Evolution with Risk Management. Proceedings of the 16th International Conference on Software Engineering and Knowledge Engineering, Banff, Canada, 2004, pp. 503-508.

[9] J. Karlsson, K. Ryan: A Cost-Value Approach for Prioritizing Requirements. IEEE Software, Vol. 14, No. 5, 1997, pp. 67-74.

[10] L. Lethola, M. Kauppinen, S. Kujala: Requirements Prioritization Challenges in Practice. Proceedings of the 4th International Conference on Product Focused Software Process Improvement, Vol. 3009, 2004, pp. 497-508.

[11] C. Lindig, G. Snelting: Assessing Modular Structure of Legacy Code Based on Mathematical Concept Analysis. Proceedings of the 19th International Conference on Software Engineering, Boston, USA, 1997, pp. 349-359.

[12] U. Priss: Formal Concept Analysis in Information Science. Annual Review of Information Science and Technology, Vol. 40, 2006, pp. 521-543.

[13] ReleasePlanner®, available at www.releaseplanner.com, last accessed April 2008.

[14] H. W. J. Rittel, M. M. Webber.: Dilemmas in a General Theory of Planning. Policy Sciences, Vol. 4, 1973, pp. 155-169.

[15] G. Ruhe, A. Ngo-The: Hybrid Intelligence in Software Release Planning. Journal of Hybrid Intelligent Systems, Vol. 1, No. 2, 2004, pp. 99-110.

[16] L. Wasserman: All of Statistics: A Concise Course in Statistical Inference (2nd Edition), Springer, 2004.

[17] R. Wille: Restructuring Lattice Theory: an Approach based on Hierarchies of Concepts. In: Ordered Sets (Ed. I. Rival), Reidel, Dordecht-Boston, 1982, pp. 445-470.