ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
DOI: 10.17559/TV-20150311144435

TOWARDS A HIGHLY CUSTOMIZABLE FRAMEWORK FOR RELEASE PLANNING PROCESS

Amir Seyed Danesh, Rodina Ahmad, Shahaboddin Shamshirband, Seyed Mahdi Zargarnataj

Original scientific paper
In software development, release planning is performed to select important features and requirements based on resource and technical constraints and the relationships between requirements. Release planning focuses on finding an optimal solution by seeking various states. This kind of solution finding reveals two remarks. First, it shows that there are various, ambiguous and uncertain parameters that influence the solution. Second, there is not only one solution to any problem. Various solutions can be found that differ in their performance (e.g. time performance, complexity performance, etc.). Consequently, many methods for release planning are often specific to only certain problem domains. This paper examines various current release planning methods to extract the common activities and thoughts in order to establish a customizable framework for release planning. Customization is done by identifying effective parameters, parameter instances and their relationships so that they can affect the selection of the right algorithm or method for each activity. Project characteristics can be specified based on the parameter instances and they are then used to determine the suitable method for achieving each activity within the whole release planning process and the results of which are recorded. This proposed highly customizable process framework with its possible customization features is then validated in several software companies. In 85 % of the cases, the suggested framework for every activity of the process fits the companies' circumstances and helps to hasten the process of release planning.

Keywords: *customizable process; effective parameters; release planning*

U cilju velike prilagodljivosti okvira za planiranje puštanja u promet

Izvorni znanstveni članak
U razvoju softvera, planiranjem puštanja u promet izabiru se važna svojstva i zahtjevi temeljeni na tehničkim ograničenjima i ograničenjima sredstava te odnosima između zahtjeva. Planiranje puštanja u promet usmjereno je na pronalaženje optimalnog rješenja traženjem raznih stanja. Takav način traženja rješenja otkriva dvije stvari. Prvo, pokazuje da postoje različiti nejasni i nesigurni parametri koji utječu na rješenje. Drugo, da ne postoji samo jedno rješenja za neki problem. Mogu postojati različita rješenja koja se razlikuju po svojim karakteristikama (na pr. u odnosu na trajanje, složenost itd.). Stoga su mnoge metode planiranja puštanja u promet često specifične za samo neke aspekte problema. U ovom se radu istražuju razne postojeće metode za planiranje pokretanja u svrhu pronalaženja nekih općih razmišljanja i aktivnosti za uspostavljanje prilagodljivog okvira za planiranje puštanja u promet. Prilagodba se postiže identificiranjem učinkovitih parametara ili primjera parametara i njihovih odnosa tako da se može izabrati pravi algoritam ili metoda za svaku aktivnost. Karakteristike projekta mogu se odrediti na osnovu primjera parametara te se oni tada primjenjuju za određivanje odgovarajuće metode za izvršavanje pojedine aktivnosti u okviru čitavog postupka planiranja puštanja u promet, a čiji se rezultati bilježe. Taj predloženi okvir vrlo prilagodljivog postupka sa svojim mogućim obilježjima prilagođavanja zatim se ocjenjuje u nekoliko softverskih poduzeća. U 85 % slučajeva predloženi okvir za svaku aktivnost u skladu je s uvjetima poduzeća i pomaže u ubrzanju postupka planiranja puštanja u promet.

Ključne riječi: *planiranje puštanja u promet; prilagodljiv postupak; učinkoviti parametri*

1 Introduction

Software development always involves numerous complications and problems among which adjusting the time, costs and resources required to deliver a release are the most important ones. Software companies try to improve their products by understanding new needs and addressing them in future releases [1]. This means that the software is not delivered to customers as a perfect pack in the first place completed in an incremental release manner. This opens up rather a new discipline in software development regarded as release planning. Software release planning is, in principle, assigning a series of features and requirements to a set of sequential releases considering technical and resource constraints [2]. In most cases, releases are developed iteratively, incrementally and eventually leading to a software version because in each release, new features are incorporated and the final release becomes a version. A weak release plan can easily wipe out weekly or even monthly activities of a team, impose heavy expenditures on developers and bring exhaustion to the team.

Generally, there are two main methods of release planning [2]. The first is the manual method which, in fact, relies on human judgments and is used when small numbers of features are available and individuals make decisions on release features through negotiations and liaison. However, considering the increasing number of features and users it is difficult to rely on manual methods to generate proper solutions [3]. The second method is the hybrid release planning approach which relies on both human and computational intelligence to systematically generate release planning solutions. In recent years, various formal models, such as Planning Game [4], Incremental Funding Method (IFM) [5], optimization-based techniques [6], Hybrid Intelligence approach [7, 8], and Lightweight Re-planning [9], have been developed in which systematic methods are able to present several alternative solutions. In addition to these, Mohebzada [10] proposes a recommendations for release planning in which individuals play the main role to follow presented systematic instructions. Reading through several of these models mentioned in Stahlberg's survey [11], one concludes that the process is getting more and more systematic.

Despite the presence of gradual and repetitive methods, the release planning process has complexities due to the possible influence of a number of factors such as the types of requirements, implementation strategies, value for the developing company, and urgency for the client, risk management and personal decisions [12]. In addition, deciding on features to be included in a specific release is a complicated task and is referred to as a wicked problem which is difficult to clearly define and often

there is no clear-cut solution [13, 14]. Bagnall et al.[6] showed the problem of selecting an optimal next release is NP-hard. Svahnberg [11] studied release planning methods and parameterized the process and mentioned some required selection factors and concluded that it may not be straightforward to find a release planning model that suits a company's needs and addresses the desired requirements selection factors. Moreover, a fully systematic approach is never sufficient and needs to be combined with the experience of professional practitioners. For this reason, Ruhe and Saliu [2] have looked at release planning from two viewpoints, art and science. The art refers to human and its capabilities and the science refers to the algorithms and methods.

This paper is structured as follows: Section 2 is about how to generate a framework base on their definitions and effective parameters and their values. In section 3 we introduce a tool called General Process Release planning (GPR) to support our framework. Next in section 4, the case studies carried out are described. In section 5, data results gathered from case study projects are reported and analysed. The paper ends with the conclusions and suggestions for future work.

2 Customizable framework

In order to define the framework for the release planning, a few steps needs to be undertaken; The first step is customizing release planning process. It means we need to know in each product of company what are the main steps or common activities for releasing based on the policy of companies. After that we need to know the relation of these activities in terms of input and output parameters. So, the next step is finding effective parameters in each activity and values of each parameter with relation to other parameters. Once these steps have been identified it is possible to define a framework for each company. GPR tool is developed in order to perform these steps to get bespoke framework for each company that will be explained in future sections.

2.1 Customization of release planning

The main objective of release planning is choosing a set of high priority requirements in order to form a new release. A set of input requirements is often prioritized by stakeholders and then estimation of resources for each requirement is added up. Finally, high priority requirements that meet resource constraints are selected to form a release [13]. These simple activities can be considered the basic activities of almost all release planning methods. Various methods often differ in the order of activities and, except for the Ad-hoc method [1] whose activities are not clearly specified, the activities are executed in all release planning methods such as the Quantitative WinWin [15], Quality Improvement Paradigm [16], Release Planning under Fuzzy Effort Constraints [17], Planning Game, Optimization-based Techniques [18], Hybrid Intelligence approach (EVOLVE Family) [19], Bi-Objective Release Planning [20]. In the Release Planning under Fuzzy Effort Constraints method [17] and Planning Game [1], the requirements prioritization is combined with the higher priorities requirements selection activity. Moreover, the "Release Planner" tool is used in the Quality Improvement Paradigm [16] and the Release Planning under Fuzzy Effort Constraints [17] to accomplish these activities. Most of the release planning methods incorporate the stakeholders in the prioritization of requirements. Although inputs, outputs and the algorithm used in each activity of release planning methods may differ in various methods, the common activities of release planning are the same.

The activities of release planning are based on the common activities of current release planning methods and contain the following:

- Requirements prioritization
- Resource estimation
- Pre-release planning
- Trade-off analysis of plans.

2.2 The relation of common activities of release planning

Based on the definition of each of the common activities in release planning, it can be observed that there is a conceptual dependency between the outputs and inputs of these activities which will lead to the pre-planning activity and the final release plan. Regardless of the release planning algorithm used in the activities and how each of the activities is implemented, these activities can be called the "release planning process activities" which receive a set of inputs in various phases and produce a release plan as the output (Fig. 1).

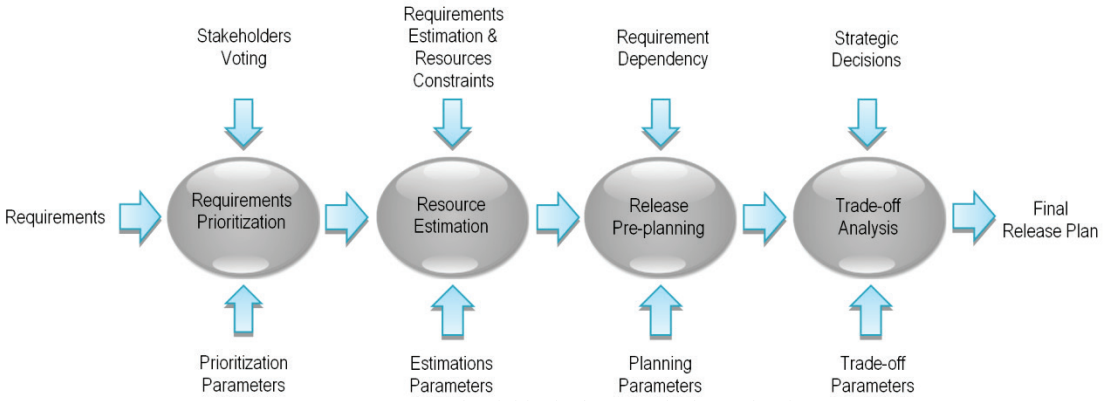

Figure 1 Inputs, outputs and activities in the general release planning process

In addition, Shafique and Saleem [21] investigated the parameters of the release planning methods. They concluded that requirements dependencies, resource consumption, constraints factors, effort constraints and stakeholders' influence factors, respectively, are the most important common parameters in the methods they studied. These were followed by other parameters such as budget and time. Since the number and type of constraints and estimations vary in the common activities, they do not face any difficulties in receiving other inputs, hence are considered to be "perfect" in terms of receiving inputs. With regards to the output, most of the releases planning methods produce one or more primary plans where the best one is then selected and this is also supported by the common activities.

Therefore, the common activities of the release planning methods can be used as the general process of release planning considering the fact that they receive various inputs and produce the needed outputs. Moreover, it has to be noted that the activities of the process must be customized and the implementation procedures have to be precise to achieve the proper plan. In other words, the general process explains a series of required activities to achieve release planning but every activity and its implementation mode must be described. Fig. 2 presents the common activities in the general process of release planning, along with their inputs and outputs. Note that some common activities which solely receive inputs are not considered as an independent activity and their key input is abandoned to the key common activities. In addition, the figure tries to show all of the input data that are effective in making the best decisions on requirements for an effective release.

This type of presentation separates the input data from the effective parameters in every certain activity. The parameters of each activity help in the proper and precise implementation of the activities. Moreover, they can be used to customize the planning process for a certain project or company (to do this, they have to have specific values). As mentioned earlier, the parameter value can be identified and classified using past experiences in release planning and reviewing release planning literature. The next section describes how this is done in every activity.

2.3 Process model for release planning

By analysing existing release planning methods, a series of common activities was identified, each containing their own definitions, inputs and outputs. However, the explanation on how every activity is implemented is lacking. Nevertheless, from these common activities, a general framework of release planning can be defined. The process is able to show the activities needed to be accomplished in order to implement an optimal release planning method. Effort is made to define the specifications and parameters of each activity so that most methods can be covered by simply altering the inputs and outputs. Covering the various release planning activities highly depends on the customization of the general release planning process. Customization here means to correctly assign values to the effective parameters in every activity in order to choose the most efficient method or algorithm, e.g. 'ranking', 'numerical assignment', or AHP in the requirement prioritization activity. To achieve this, the effective parameters must first be identified, described and be assigned values. It is also necessary to determine their effect in the selection of the right method or algorithm in every activity.

Below, we first describe the effective parameters in every activity of the general release planning process. Then, for each activity, we investigate how these parameters can influence the selection of the right method or algorithm. Finally, we examine the influence of the correlations of these parameters.

2.4 Customization of requirements prioritization activity

Requirements prioritization is the first activity in the release planning process, which facilitates requirement analysis in the next activity. The various tasks in this activity are presented below.

Effective parameters in requirements prioritization

In its simplest form, this task is accomplished in an ad hoc manner regardless of any given parameters. In its current form, which is adopted in most release planning methods, priority is taken from the stakeholders' viewpoint and then integrated with the developers' vote, hence, requirement priority is determined [11]. The most well- known prioritization method under this category is the Analytical Hierarchy Process (AHP) [22]. The process in this method is designed so that it is consistent and can interact with human intellect and nature. From a technical viewpoint, AHP is one of the most comprehensive systems designed for decision-making with multiple measures since it allows the formulation of the problem in a hierarchical manner and enables one to consider various quantitative and qualitative measures [23]. Besides this, other methods for requirements prioritization have been introduced, some of which are mentioned below [20, 22]:

- Cumulative Voting or the 100-Dollar test
- Numerical Assignment (Grouping)
- Ranking
- Top-Ten Requirements
- Quality Function Development (QFD)
- Cost-Value Approach.

In addition to these methods, other prioritization techniques exist among which are B-Tree based methods, Quality-based methods, genetic Algorithm or Value-based methods [24, 25]. These methods try to increase the requirements prioritization quality by decreasing the number of comparisons, considering certain specifications for prioritization and lessening the complexity. Nevertheless, the question facing many software development teams is how to select the proper method to prioritize the software. Most teams seek a simpler prioritization method, hence the use of methods such as Ad Hoc and Numerical Assignment [26]. Answering the above question can clarify some requirements prioritization parameters relevant to method selection.

Aasem et al. [27] compared existing requirements prioritization methods and presented measures to evaluate them. Some of these measures include scale, granularity, considered parameters for prioritization and

implementation type as shown in Tab. 1. This classification helps to find best fitted requirements prioritization methods based upon specific parameters. Requirement manager can determine values of each parameter and find out suitable method and speed up method selection. According to these measures, every prioritization method is only suitable for certain cases. For example, since AHP, B-Tree and 100-Dollar Test are complicated methods, small-size companies with a limited number of stakeholders are not expected to be able to use them. Therefore, the number of stakeholders involved in requirements prioritization and the size of the software development team or company can partly determine or limit a certain prioritization method. This is also true about granularity which shows the precision of every certain output, e.g. fine, medium, coarse or extremely coarse. If the requirements are to be prioritized carefully, then the methods with a fine granularity are used but when the precision is less important, the methods with a coarse granularity can be employed. The identified parameters can be used as a classifier for requirement prioritization method. In fact, finding such specifications that can limit the selection of the prioritization method and identifying the parameters to be used in the prioritization can help the customization of the requirements prioritization activity and therefore, can determine various customization parameters. In addition, similar to Aasem et al. [27] in which classifications were made for every parameter and methods were placed in these classes, primary classifications must also be made for every requirement prioritization parameter.

Table 1 Classification of Prioritization methods [27]

Technique	Scale	Granularity	Sophistication	Aspect	Perspective	Type
AHP	Ratio	Fine	Very Complex	Strategic Importance, Penalty	Product Manager	Algorithmic
B-Tree	--	Fine	Complex	-	-	Algorithmic
100-Dollars Test	Ratio	Fine	Complex	Customer importance	Customers	Manual
Ranking	Ordinal	Medium	Easy	Volatility	Requirements Specialist	Manual
Numerical Assignment	Ordinal	Coarse	Very Easy	Time, Risk	Project Manager, Requirements Specialist	Manual
Top 10	---	Extremely Coarse	Extremely Easy	Customer importance	Customers	

Using the results of these studies and those of other corresponding literature [29], a list of specifications can be obtained in which the best prioritization methods are described. To achieve this goal, an investigation of these specifications and their correlated objectives was performed which has yielded a set of such specifications with their allowed values. Tab. 2 shows one of the parameters along with their instances. An instance of every parameter represents allowed values for that parameter and can be added later to expand the method.

Table 2 One parameter of requirements prioritization

Parameter	Allowed value	Description
Market type (MT)	Customized (MT_1)	The software is designed and developed for a certain costumer
	Limited customer (MT_2)	The number of customers is limited and every customer can have different views about each requirement
	Unlimited customer (MT_3)	The number of customers is unlimited and unlimited number views are available about each requirement

These presented parameters can still be expanded with more validations and tests. Despite some release planning methods in which requirement prioritization and resource estimations are done simultaneously [22], in this context, requirements are prioritized regardless of their resource estimations.

2.4.1 The relationships between effective parameters in requirements prioritization

For each parameter in the requirements prioritization activity, the existence of a relation between its instances and other parameter's instances forms an ordered pair. The relationship between "market type" and other effective parameters on requirements prioritization is defined as the following ordered pair:

$$K_{MT} = \{x \in MT, y \in \{DM, TS, RN, RG, PI, TE, DE\} | (x, y)\}$$

According to this definition, members of every K set are as follows:

K_{MTDM} = {(MT_1, DM_1), (MT_1, DM_2), (MT_1, DM_3), (MT_1, DM_4), (MT_2, DM_2), (MT_2, DM_3), (MT_2, DM_4), (MT_3, DM_2), (MT_3, DM_4)}

K_{MTTS} = {(MT_1, TS_1), (MT_1, TS_2), (MT_1, TS_3), (MT_2, TS_2), (MT_2, TS_3), (MT_3, TS_2), (MT_3, TS_3)}

K_{MTRN} = {(MT_1, RN_1), (MT_1, RN_2), (MT_1, RN_3), (MT_2, RN_2), (MT_2, RN_3), (MT_3, RN_2), (MT_3, RN_3)}

K_{MTRG} = {(MT_1, RG_1), (MT_1, RG_2), (MT_1, RG_3), (MT_2, RG_1), (MT_2, RG_2), (MT_2, RG_3), (MT_3, RG_1), (MT_3, RG_2), (MT_3, RG_3)}

K_{MTPI} = {(MT_1, PI_1), (MT_1, PI_2), (MT_1, PI_3), (MT_2, PI_1), (MT_2, PI_2), (MT_3, PI_1)}

K_{MTTE} = {(MT_1, TE_1), (MT_1, TE_2), (MT_2, TE_1), (MT_2, TE_2), (MT_3, TE_1), (MT_3, TE_2), (MT_3, TE_3)}

K_{MTDE} = {(MT_1, DE_2), (MT_1, DE_3), (MT_2, DE_1), (MT_2, DE_2), (MT_3, DE_1)}

In the next activity, the relationship between the "development methodology" parameter and the other requirement prioritization parameters is examined, except the "market type" which was investigated before. The same procedure is performed for all other parameters i.e. "team size", "number of requirements", "requirements granularity", "number of prioritization inputs" and "team experience". Hence, these sets show the possible relations between instances of two parameters in the requirement prioritization activity and make all possible sets of ordered pairs. When the number of combinations increases exponentially, the size of problem increases [28].

Every set of ordered pairs is considered as a certain or definite state. Each set is combined with other sets of ordered pairs with which it has a common point and forms a set of common ternaries. Every common ternary represents a combination of an instance of three parameters, the accuracy of which must be determined like the relationship between two instances. Moreover, it is necessary to omit improbable or less likely states which can be neglected. It must be noted that integrating all states will generate other new common states having all three parameters of an instance, so these must be omitted. Having the states of the three parameters generated, it is necessary to combine them to achieve a four-parameter state and this is repeated until an *N*-parameter state is obtained. A tool is developed to record and generate these different states, to perceive the inter-relationships of these states and to determine common and uncommon states. This tool is capable of perceiving every parameter's instances and can generate the relationship status between instances of two, three or more parameters. Having identified the inter-relationships between parameters' instances in every stage, the tool automatically generates new multi-parameter relationship states and allows users to recognize the likelihood of new relationships. Fig. 2 displays the relationship between requirements prioritization parameters instances.

	DE1	DE2	DE3	DM1	DM2	DM3	DM4	MT1	MT2	MT3	PI1	PI2	PI3	RG1	RG2	RG3	RN1	RN2	RN3	TE1	TE2	TE3	TS1	TS2	TS3
DE1					X		X		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
DE2				X	X	X	X	X	X		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
DE3				X	X	X	X	X			X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
DM1		X	X					X			X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
DM2	X	X	X					X	X	X	X	X	X	X	X		X	X		X	X	X	X	X	
DM3		X	X					X	X		X	X	X	X	X	X	X	X	X	X	X	X		X	X
DM4	X	X	X					X	X	X	X	X	X	X	X		X			X	X	X	X	X	
MT1		X	X	X	X	X	X				X	X	X	X	X	X	X	X	X	X	X		X	X	X
MT2	X	X			X	X	X				X	X		X	X	X		X	X	X	X			X	X
MT3	X				X		X				X			X	X	X		X	X	X	X	X		X	X
PI1	X	X	X	X	X	X	X	X	X	X				X	X	X	X	X	X	X	X	X	X	X	X
PI2	X	X	X	X	X	X	X	X	X					X	X	X	X	X	X	X	X	X	X	X	X
PI3	X	X	X	X	X	X	X	X						X	X		X	X		X	X	X	X	X	X
RG1	X	X	X	X	X	X	X	X	X	X	X	X	X				X	X	X	X	X	X	X	X	X
RG2	X	X	X	X	X	X	X	X	X	X	X	X	X				X	X	X	X	X	X	X	X	X
RG3	X	X	X	X		X		X	X	X	X	X					X	X		X	X	X	X	X	X
RN1	X	X	X	X	X	X	X	X			X	X	X	X	X	X				X	X	X	X	X	X
RN2	X	X	X	X	X	X		X	X	X	X	X	X	X	X	X				X	X	X		X	X
RN3	X	X	X	X		X		X	X	X	X	X		X	X					X	X	X			X
TE1	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X				X	X	X
TE2	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X				X	X	X
TE3	X	X	X	X	X	X	X			X	X	X	X	X	X	X	X	X	X				X	X	X
TS1	X	X	X	X	X		X	X			X	X	X	X	X	X	X			X	X	X			
TS2	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X		X	X	X			
TS3	X	X	X	X		X		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X			

Figure 2 The relationship between instances of requirements prioritization parameters

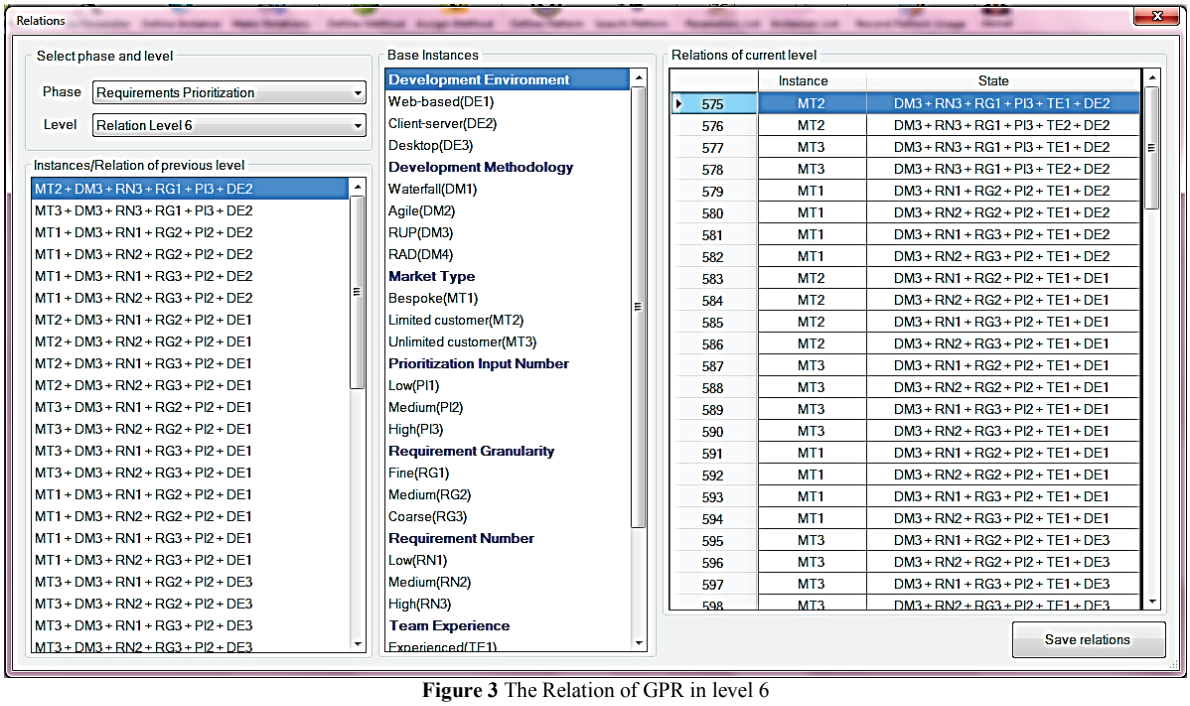

Figure 3 The Relation of GPR in level 6

3 Tool support

In every activity of the general release planning, in its simplest form, a table that contains the mapping between different parameter instances and methods can be used. By using the table, the software team or developer organization can select parameter instances in order to identify a suitable method to perform in each activity of the general process based on past experiences. As observed in the customization of every activity of the general release planning process, accomplishing such a mapping requires determining the effect of parameters and their instances on each other and making use of past experiences in order to select the best method for every activity. The main disadvantage of using this type of mapping is the large number of states in the mapping table which in turn, will lead to the generation of numerous new states each time a definite parameter or instance is added. To overcome this problem, a software tool called the "General Process Release planning (GPR)" was developed that can speed up the process.

The "General Process Release planning (GPR)" tool is used to optimize the customization process and to remove the main weakness of this method. By using the GPR tool, a release planner is allowed to enter his/her experiences of inter-relationships between parameters and their instances and make the mapping table more complete and accurate. The GPR tool is designed and implemented to enter and record all states of the parameters and their instances and to trace their inter-relationships as demonstrated in Fig. 3.

The tool allows the release planner to add new parameters and instances in order to facilitate the establishment of a relationship between them. The release planner or the project manager can use the already entered list of parameters and instances to establish a relationship between two different instances and create a new ordered pair. Having established the relationship between all allowed instances of two definite parameters, the tool can now be used to enter the relationship between three parameters. It allows the formation of an ordered ternary which includes three instances of three different parameters and removes non-ordered pair parameters automatically. Thus, if instances of two parameters are not related, they are never placed in an ordered tuple.

By using this method, it is possible to simply add new parameters and instances to every activity of the release planning process in addition to being able to define the inter-relationships between parameters and to remove unrelated ones. The final activity of the tool generates an ordered tuple of parameters of a certain activity. These sets differ only in their instances. A proper method can be specified for each set in every activity of the general release planning process. For example, a requirement prioritization method can be selected in accordance to the set of requirement prioritization parameters and the tool is capable of recording the selected method.

The tool seeks all proper methods pertaining to a certain parameter by receiving data concerning the parameter instances. The more considerable the size of the data and the number of instances are, the more accurate the tool will search and a more proper method will be suggested to the release planner or project manager. Since the tool is based upon the customizable release planning process, it supports all of the activities of release planning. Moreover, it provides the release planner with the ability to customize the release planning activities. In this way, the planner can enter arbitrary parameters and instances, if necessary. This leads to a better applicability of the tool in different software development projects.

4 Results

The process framework, with the support of the software tool, was employed in three software companies in order to examine the applicability of the customizable release planning process and the release planning tool. The chosen companies have different characteristics and are in different fields. Effort was made to select companies that, firstly, have a development methodology and, second, their teams' members are more than four people and finally, release planning is accomplished in each of their teams independently. All the studied companies have different software teams. Every project manager was asked to extract his/her release planning method in every activity of the GPR tool using the initial data available in the software database. Characteristics of every company are summarized in Tab.3.

Number of 14 projects from 3 companies is selected to use process model, and some parameter distributions of projects are presented in Fig. 4.

Table 3 The companies' descriptions

Name	Description
A	The company has been developing banking and insurance software since 7 years ago and now is known as one of the pioneers on national level. The company is in charge of developing and maintaining "Core Banking" software in two state banks and one private bank in Iran and is considered to be one of the main developers of comprehensive financial guidelines of the country.
B	The company has been working on software development for more than 25 years. This is the first software company in Iran which supplied Windows-based systems as an integrated one in 1997. Nowadays, having more than 9500 customers in big, medium and small businesses and more than 1100 labour, the company is one of the biggest software developing companies in Iran and is almost dominant in the field of financial software.
C	The company was founded in 2005 by a combination valid IT companies and support and investment of active companies in the capital market with the goal of presenting the first "total online guideline" in the field of capital market. With less than 8 years of activity background the company hosts 51 agents in the Stock Exchange, 41 agents in Goods Exchange and 62 investment funds with more than 1500 branches in Iran.

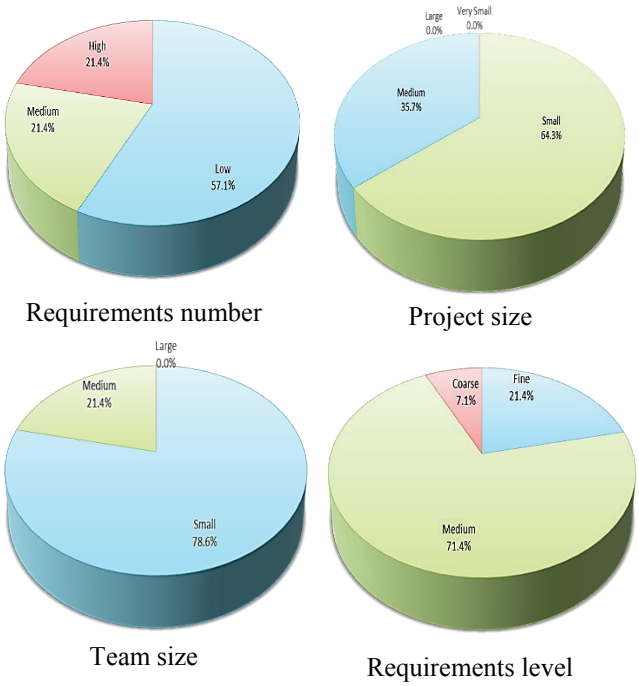

Figure 4 Case studies project parameters properties

4.1 Evaluation results

Team managers and release planners were provided with the GPR tool where they used the proposed method of the tool in at least two software releases using every activity in the general process of release planning. The results showed that this method was adopted more in companies that already have their own release planning processes. Furthermore, the success of the method is evaluated based on the votes of the project managers and release planners. Referring to Fig. 5, it can be concluded that around 85 % of the companies agree that the method and use of the GPR tool is better than their previously employed method of release planning. It should be noted that most of the companies use simple methods to plan a release.

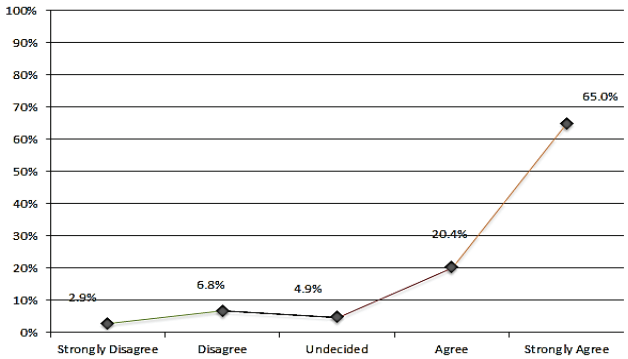

Figure 5 Evaluation results of release planning process model in the studied companies

In addition, a survey was performed among project managers and release planners on every definite stage (results of which are shown in Fig. 6) in order to compare the quality of the presented method in all activities of the customizable process. They mostly agree on the requirements prioritization activity since the proposed methods were more accurate and applicable for that stage but they only provided an overall view for the resource estimation activity. Although this depends on the resource estimation methods and the challenges of method classification, project managers tend to pay more attention to a method's applicability rather than the sole task of method selection.

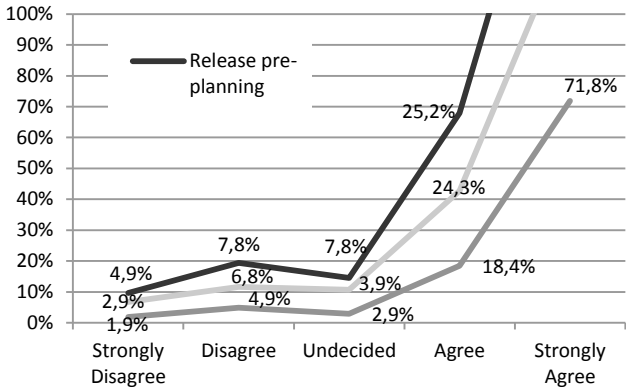

Figure 6 The evaluation results of the activities of the release planning process model in studied companies

The results of using this method along with the GPR tool in studied companies showed that most companies do not employ any release planning method and those that do, employ the simplest methods and they are the ones that tend towards making use of the proposed method. On the other hand, those companies that have been using a more mature release planning process mostly identified with the proposed software methods as convergent with their current ones. However in some cases, the project managers did not use the method proposed by the software tool because of reasons originating from a project's implicit characteristics. More importantly, most companies considered the GPR tool's suggestions as well-suited and more accurate.

5 Conclusion

The framework of release planning is based on a set of common features and activities in different planning methods and covers all activities of release planning. Customizing this process and preparing effective parameters and their instances enables its application in various software development projects. The process helps release planners perform their tasks in an activity-by-activity manner and use the best experiences available in employing the method in a certain way. The results of using the proposed framework in software companies and software projects showed that it is suitable for every field and most companies can optimize their own release planning processes through customization of the proposed method. The customizability of the method and the ability to add various parameters can help companies considerably in adapting the proposed method with their current experiences.

The summary of result of using release planning framework is as follows:

- ***Reduction of time spent in release planning***

Method selection in release planning and its various steps is a time consuming process in most companies and hence they tend to use simple method or equipped tools

which may even be improper for the company or its project. In fact, identifying different parameters of process model steps such as resource estimation or requirements prioritization requires time and expert labour and resource allocation is not always possible in most projects especially small ones. The process model provides rapid and simple way for the release planner or project manager with successful experiences of other projects to determine methods suiting his or her considered project.

- ***Using project characteristics in the release planning***

Absolutely, one of the most important issues in release planning is to select a method which best suits the project. Most release planning methodologies solely try to present a planning method regardless of its fitness to the project. Besides, in such methods the project manager or release planner has to spend much time or rely on his/her experience to select a method relative to project characteristics. But this is usually challenged in different levels and the fitness is not ensured. But, Process model of release planning uses a project's specific features and the best of past experiences to present a method suiting the project.

- ***Highly adoptability of frameworks***

Using the various properties to select the method of each step of process model causes the process model can be used in different project domain. Also, project properties of a similar project can be used as experience for other similar projects.

Release planning is considered as one of the most important sections of software engineering and plays a significant role in time and cost optimization and in achieving a software product. The customizable release planning process and the GPR tool presented in this paper aim to make different activities of release planning parametric and accurate in order to facilitate the task for software companies and make results more reliable through the use of this method. Thus, the experiences of different companies can be recorded and published by using this method in order to present a more precise analysis. Moreover, optimizing the proposed method using method parameters and results can be considered as one of the tasks to be accomplished in the future.

6 References

[1] Danesh, A. A Survey of Release Planning Approaches in Incremental Software Development. // Computational Intelligence and Information Technology. 250, (2011), pp. 687-692.DOI:10.1007/978-3-642-25734-6_119

[2] Ruhe, G.; Saliu, M. O. The art and science of software release planning. // Software, IEEE. 22, (2005), pp. 47-53. DOI: 10.1109/MS.2005.164

[3] Dybå, T.; Dingsøyr, T. Empirical studies of agile software development: A systematic review. // Information and Software Technology. 50, 8(2008), pp. 833-859. DOI: 10.1016/j.infsof.2008.01.006

[4] Ruhe, G.; Saliu, O. The Science and Practice of Software Release Planning, University of Calgary, 2005.

[5] Denne, M.; Cleland-Huang, J. The incremental funding method: data-driven software development. // Software, IEEE. 21, (2004), pp. 39-47 DOI: 10.1109/MS.2004.1293071

[6] Bagnall, A. J.; Rayward-Smith, V. J.; Whittley, I. M. The next release problem. // Information and Software Technology. 43, 14(2001), pp. 883-890. DOI: 10.1016/S0950-5849(01)00194-X

[7] Przepiora, M.; Karimpour, R.; Ruhe, G. A hybrid release planning method and its empirical justification. // Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement, Lund, Sweden, 2012. DOI: 10.1145/2372251.2372271

[8] Ruhe G.; Ngo, A. Hybrid Intelligence in Software Release Planning. // Int. J. Hybrid Intell. Syst. 1, (2004), pp. 99-110. DOI: 10.3233/HIS-2004-11-212

[9] Albourae, T.; Ruhe, G.; Moussavi, M. Lightweight Replanning of Software Product Releases. // Software Product Management, 2006. IWSPM '06. International Workshop on. 2006, pp. 27-34 DOI: 10.1109/IWSPM.2006.5

[10] Mohebzada, J. G. A Recommendation System for Planning Software Releases. // MS, Engineering, Electrical and Computer, University of Calgary, 2012.

[11] Svahnberg, M.; Gorschek, T.; Feldt, R.; Torkar, R.; Saleem, S. B.; Shafique, M. U. A systematic review on strategic release planning models. // Information and Software Technology. 52, 3(2010), pp. 237-248. DOI: 10.1016/j.infsof.2009.11.006

[12] Ruhe, G. Software Release Planning vol. 3: Handbook of Software Engineering and Knowledge Engineering, 2005.

[13] Carlshamre, P. Release Planning in Market-Driven Software Product Development: Provoking an Understanding. // Requirements Engineering. 7, 3(2002), pp. 139-151. DOI: 10.1007/s007660200010

[14] Ngo-The, A.; Ruhe, G. A systematic approach for solving the wicked problem of software release planning. // Soft Computing. 12, 1(2008), pp. 95-108. DOI: 10.1007/s00500-007-0219-2

[15] Ruhe, G.; Eberlein, A.; Pfahl, D. Trade-off Analysis for Requirements Selection. // International Journal of Software Engineering and Knowledge Engineering. 13, 4(2003), pp. 345-366. DOI: 10.1142/S0218194003001378

[16] Amandeep; Ruhe, G.; Stanford, M. Intelligent Support for Software Release Planning. // Product Focused Software Process Improvement. vol. 3009, F. Bomarius and H. Iida, Eds., ed: Springer Berlin Heidelberg, 2004, pp. 248-262. DOI: 10.1007/978-3-540-24659-6_18

[17] An, N.-T.; Saliu, M. O. Fuzzy Structural Dependency Constraints in Software Release Planning. // Fuzzy Systems, 2005. FUZZ '05. The 14th IEEE International Conference on. 2005, pp. 442-447. DOI: 10.1109/FUZZY.2005.1452434

[18] Freitas, F. G.; Coutinho, D. P.; Souza, J. T. Software Next Release Planning Approach through Exact Optimization. // International Journal of Computer Applications. 22, (2011), pp. 1-8. DOI: 10.5120/2607-3636

[19] Greer, D.; Ruhe, G. Software release planning: an evolutionary and iterative approach. // Information and Software Technology. 46, (2004), pp. 243-253. DOI: 10.1016/j.infsof.2003.07.002

[20] Saliu, M. O.; Ruhe, G. Bi-objective release planning for evolving software systems. // Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering, Dubrovnik, Croatia, 2007. DOI: 10.1145/1287624.1287641

[21] Bin Saleem, Saad; Usman Shafique, Muhammad. A Study on Strategic Release Planning Models of Academia and Industry. // Master Thesis no: MSE-2008-24, School of Engineering Blekinge Institute of Technology, 2008.

[22] Berander, P.; Andrews, A. Requirements Prioritization. // Engineering and Managing Software Requirements, A. Aurum and C. Wohlin, Eds., Springer Berlin Heidelberg, 2005, pp. 69-94. DOI: 10.1007/3-540-28244-0_4

[23] Saaty, T. L. The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation (Decision Making Series), McGraw-Hill, 1980.
[24] Perini, A.; Susi, A.; Avesani, P. A Machine Learning Approach to Software Requirements Prioritization. // Software Engineering, IEEE Transactions on. 39, (2013), pp. 445-461. DOI: 10.1109/TSE.2012.52
[25] Iqbal, M. A.; Zaidi, A. M.; Murtaza, S. A New Requirement Prioritization Model for Market Driven Products Using Analytical Hierarchical Process. // Data Storage and Data Engineering (DSDE), 2010 International Conference on, 2010, pp. 142-149.
[26] Svensson, R. B.; Gorschek, T.; Regnell, B.; Torkar, R.; Shahrokni, A.; Feldt, R. et al. Prioritization of quality requirements: State of practice in eleven companies. // Requirements Engineering Conference (RE), 19th IEEE International, 2011, pp. 69-78 DOI: 10.1109/RE.2011.6051652
[27] Aasem, M.; Ramzan, M.; Jaffar, A. Analysis and optimization of software requirements prioritization techniques. // Information and Emerging Technologies (ICIET), International Conference on, 2010, pp. 1-6. DOI: 10.1109/ICIET.2010.5625687
[28] Hosseinabadi, A. A. R.; Kardgar, M.; Shojafar, M.; Shamshirband, S.; Abraham, A. GELS-GA: Hybrid metaheuristic algorithm for solving Multiple Travelling Salesman Problem. // Intelligent Systems Design and Applications (ISDA), 14th International Conference on, 2014, pp. 76-81, DOI: 10.1109/ISDA.2014.7066271
[29] Marjaie, S. A.; Kulkarni, V. Recognition of Hidden Factors in Requirements Prioritization Using Factor Analysis. // Computational Intelligence and Software Engineering (CiSE), International Conference on. 2010, pp. 1-5 DOI: 10.1109/CISE.2010.5676794

Authors' addresses

Amir Seyed Danesh
Department of Software Engineering,
Faculty of Computer Science and Information Technology,
University of Malaya,
50603, Kuala Lumpur, Malaysia
E-mail: a.seyeddanesh@gmail.com

Rodina Ahmad
Department of Software Engineering,
Faculty of Computer Science and Information Technology,
University of Malaya
50603, Kuala Lumpur, Malaysia
E-mail: rodina@um.edu.my

Shahaboddin Shamshirband
Department of Computer System & Technology,
Faculty of Computer Science & Information Technology,
University of Malaya,
50603, Kuala Lumpur, Malaysia
E-mail: shahab1396@gmail.com

Seyed Mahdi Zargarnataj
Shahid Beheshti University, Tehran, Iran
E-mail: m.zargarnataj@gmail.com