# Toward a Pattern-Based Release Planning

Amir Seyed Danesh, Rodina Ahmad, and Seyed Mahdi Zargarnataj

***Abstract*—Release planning is an important activity in the software development that can adjust budget and time over various releases. Most software companies always concern about the best delivery of a new release although new features and needs are available. Effective parameters on release planning adopt certain values in a project but as a result of their ambiguity and diversity they take different values in a same project. Therefore, despite existence of various release planning methods, release planning process is deficient. This means that every method performs planning task by the help of defined algorithms and specific parameters. The present paper aims to develop a general process for release planning which result from investigation of common tasks and steps of current release planning method. Then, certain patterns are obtained by customizing the common tasks and steps. It also explains how to generate the patterns and achieve them in order to plan for a release.**

***Index Terms*—Pattern-based, software, release planning.**

## I. Introduction

One of the most important problems of most companies in developing and maintaining huge and complicated software systems is to determine functionality and changes to be exerted on different software releases[1]. Companies try to improve their products through an understanding of new needs and considering them in future release [2]. These new needs and changes lead the software toward higher commercial value and applicability. Release planning, in fact, is choosing an optimal set of features for every certain release [3].

Another problem rises to the extent that some of the organizations do not see release planning activity as independent, and they think basic decisions in theirs organization are based on business rule and needs [4] and sometimes, a very successful method of release planning in a certain project is not efficient in another.

A variety of methods are developed each of which influence and improve release planning in a different way. Effective parameters on release planning adopt certain values in a project but as a result of ambiguity and variety of the parameters they take different values in a same project or firm [5]. Thus, it is claimed that there is a need to develop a general and comprehensive process based upon current methods. This means that studying common tasks and steps leads to a general process. Then, they are employed along with customization procedure to obtain certain patterns.

Manuscript received November 7, 2013; revised January 23, 2014.

Amir Seyed Danesh and Rodina Ahmad are with the Department of Software Engineering, Faculty of Computer Science and Information Technology, University of Malaya, 50603, Kuala Lumpur (e-mail: amir_s_d@siswa.um.edu.my, rodina@um.edu.my).

Seyed Mahdi Zargarnataj is with the Information System Architecture Research Center,Shahid Beheshti University, Tehran, Iran (e-mail: m.zargarnataj@gmail.com).

## II. Software Release Planning

Improving software development processes is a significant step for software companies. A strong tendency is to follow the so-called iterative and incremental development.

Release planning is, in fact, assigning certain features in the next release respecting technical and resource constraints [6]. Iterative and incremental software development is always employed when a huge software project (having numerous features) aims to produce a software product. It allows developers to provide various functionalities for users in incremental releases (with an increase in the number of available features)[7]. Various methods and models are proposed for release planning and Svahnberg presented a systematic review of release planning methods in [8]. Therefore, different factors influence this process and current methods are not fully support release planning details in the area of software development.

Hence, the questions emerge that how can you achieve a generality in release planning or how can you define certain patterns for release planning through investigation of common tasks and steps of these methods. Therefore, the first step is to conduct the general process of release planning that resulted from examination of different release planning methodologies and the second is to develop patterns for every certain step of the general process.

## III. Objective of Study

The present paper aims to develop a general and comprehensive process based upon current methods. This means that studying common tasks and steps leads to a general process. Then, they are employed along with customization procedure to obtain certain patterns.

Conducting the general process of release planning is the result of examining a quite complete set of release planning methods. Studying release planning methods and their different stages changes the steps and common tasks to the process.

## IV. Conducting the General Process of Release Planning

Release planning is, in fact, choosing a set of requirements in order to generate new release of software. This means that the set of received requirements must first be prioritized and then estimations and forces related to various resources are added up. Finally, highly prior requirements or properties evaluating estimations and forces

DOI: 10.7763/IJCCE.2014.V3.290

are selected to develop the release [3]. Similar to the three mentioned basic steps (requirements prioritization, resource estimation and Pre-release planning) these cases can be considered as common features among release planning methodologies. However, they are not considered as key steps since some methodologies do not respect them or acquire or generate their required data in a certain manner. Nevertheless, the following common steps are considered for release planning process, as presented in Fig. 1.

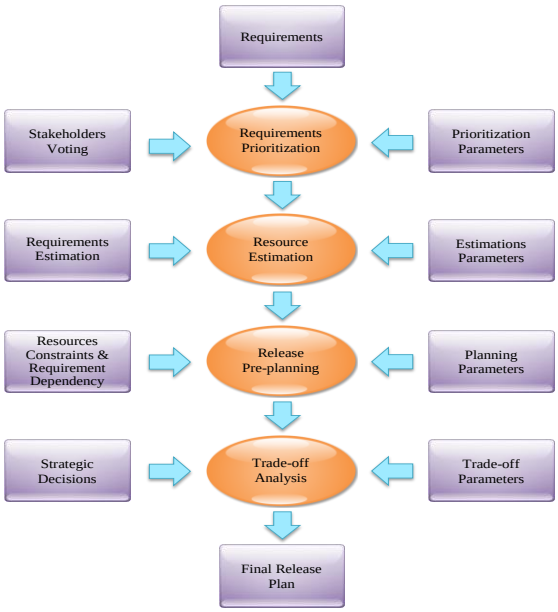


Fig. 1. Inputs, outputs and activities in release planning general process.

### A. Requirements Prioritization

The primary priority of requirements (from stakeholders' viewpoint) and requirement prioritization parameters are received in this step based on which requirement priorities are determined. From the software developer organization's viewpoint some stakeholders are prior to others and this is specified by weighing them. Each stakeholder can have a certain weight which is used to display requirements of that stakeholder from a set. Stakeholders' weight can be determined based on certain rules and principles. If stakeholders are not prior, from developer organization's viewpoint, all of them are considered to have a same weight.

### B. Resource Estimation

For every determined requirement the corresponding resource must be estimated. This is not always necessarily respected since some resources can be calculated using several parameters or other resources. Having perceived estimations of every certain requirement it is necessary to perceive constraints of each requirement (a task which is performed in next step).

### C. Pre-Release Planning

Estimations and constraints are determinant parts of every release planning methodology. In fact, constraints show the number of requirements that can be selected for every release. If a release is already planned, a certain set of constraints are to be used, but if a release is planned gradually and repeatedly (like most software design and planning methods), new constraints should be received for each software.

### D. Analysis of Primary Release Plan and Selecting The Final Release

Pre-release planning can be investigated and analyzed from various perspectives (and respecting various parameters). This is accomplished by the project manager in an Ad Hoc manner but it can also be done by defining some certain parameters. The most important parameters in analyzing primary release plans include type of resources used in every planning, amount of resources used (particularly time and cost) and flexibility of each plan. Furthermore, analysis parameters can be prioritized to be ranked based on their priority.

## V. Release Planning Customization Using a Pattern

It is tried in customization of release planning general process to investigate experiences in the form of effective parameters on each stage and to consider the influence of every parameter independently. Besides, PRP Software is designs and run to enter and record all states and to trace interrelationships of parameters and instances. The software provides the release planner with possibility to add new parameters and instances and allows a simple relationship between them.

Every step of release planning general process involves a set of inputs, outputs and parameters and the suitable method is selected using the parameters and their instances in customization process. Selecting a proper method for every step determines inputs to be used and outputs to be generated. Although most methods have some common and some uncommon inputs in every step but determining precise inputs is highly dependent on selected method. Fig. 2 illustrates relationships between parameters, instances and inputs of the selected method in every step.

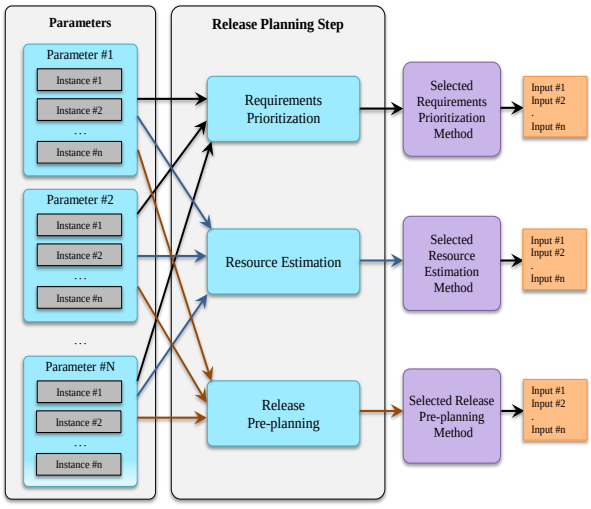


Fig. 2. Relationships between steps of release planning general process, parameters, instances and inputs of the selected method.

Achieving a proper method to perform every certain task, in its simplest form, includes developing a table of mapping between different parameter instances and methods. This table examines all different parameter instances using its own mapping method. Using the table the software team or developer organization can selects parameters instances to perceive a suitable method to implement its considered

release planning general process (based on past experiences). As Fig. 2 displays, steps can have common parameters instances of which affect the whole release planning procedure.

One of the most important advantages of this mapping is rapid and simple customization of release planning process. By using initial data and entering them into the PRP the release planner can perceive certain proposed methods. Moreover, he (she) has benefited from past experiences used in mapping without previous knowledge. The past experience is the relationship between different parameters in customization which has resulted in the mapping table. Using PRP the planner is even able to enter his (her) own experiences of interrelationships of parameters and instances and make the table more perfect and precise.

Huge number of steps is the main disadvantage of the mapping table. This leads to generation of numerous states as a result of adding a certain parameter or instance. Furthermore, many generated states may be practically impossible which should be identified and separated by the planner. Therefore, the customization process has to be altered or optimized so that the planner can achieve the most suitable methods in shortest time.

The concept of "pattern" which is a well-known notion in software development is used in release planning to optimize planning customization and remove the main weakness of this method. In fact, in optimizing the customization method release plan is generated using parameters and instances and various experiences of that step instead of creating different states in every step. In other words, not all states are generated for parameter instances in release planning but only successful experiences in using a certain method are documented based on instances. This removes the need to enter and investigate all states and the planner can directly enter his (her) own experience. Using past experiences in software development is tightly tied with the concept of "pattern". There are different patterns in software development (such as designing pattern and architecture pattern) the main idea of which is to make rapid use of pas experiences in order to accomplish software development tasks.

## VI. Software Development Patterns

The current use of the term "pattern" in originates from writing of an architect named "Christopher Alexander" [9] who had written many books in urban planning and building architecture. In 1987, two researchers, named "Ward Cunningham" and "Kent Beck"[10], who were working on designing user interface with Smalltalk programming language decided to use some of Alexander's ideas to develop a 5-lattern small language. Later Jim Coplein [11]used published results of the two researchers to prepare a catalogue of idioms in C++ language and published it a one of his books. Between 1990 and 1992 members of "Gang of Four" (Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides) met each other frequently and tried to prepare a catalogue of patterns and introduced it in a materialist workshop in 1991. In August 1993 Kent Beck and Grady Booch met in Colorado, a meeting which formed the basis of Hillside group. Not long afterward the famous book of "Designing Patterns" was published by the GoF [12]. Several definitions of patterns are presented in various texts. Generally, it can be said that every pattern shows how a certain problem is solved through a specific solution. But pattern is more than a solution and indicates that the problem in consideration occurs in a certain context in which there are other interests. In fact, the solution proposed by a pattern contains a type of structure which brings about a balance between specific interests or constraints to present the best solution for a problem.

Every pattern has its own from which shows that pattern's specifications. The Gang of Four [12] presented a form for architecture patterns which is mostly used and is called "GoF form". Of course, there are also other patterns available but mostly consent on some basic specifications and these must be clear for the pattern in each form.

## VII. Definition of Release Planning Patterns

Similar to other software developing patterns, release planning pattern should be well defined to be used in release planning. Patterns employed in release planning are divided into two groups based on their effect: patterns effective on all release planning steps and patterns effective on a certain step. The former group is “release planning patterns” and the latter is “release planning steps’ patterns”. Moreover, as mentioned in release planning customization, some parameters are common among some multiple steps and some others only influence a certain one. Putting all parameters together makes it possible to obtain the set of effective parameters on all release planning steps which are called “effective parameters on release planning”. According to what mentioned above, release planning pattern is displaying a best experience in release planning which:

- Is obtained by valuing every effective parameter on release planning,
- Contains value of one or more parameter,
- Determines running mode of every step in the general process, and
- Is shown by instances of effective parameters on release planning.

By the definition, release planning pattern can result from influence of one or more already, successfully implemented effective parameter which is used in similar problems with similar parameters instances for several times. Moreover, release planning pattern is displaying a best experience in release planning which:

- Is obtained by valuing every parameter effective on the same step of release planning general process,
- Contains value of one or more effective parameters on the same step of release planning general process,
- Determines running mode of the same step of release planning general process,
- Is shown by instances of effective parameters on the same step of release planning general process.

The pattern of release planning steps can be divided into three scopes using this definition: requirement prioritization, resource estimation and Pre-release planning in each of which patterns specific to that certain step is identifies and described. The patterns are developed using experiences in

resource estimation, requirement prioritization and release planning tasks.

## VIII. The Structure of Release Planning Patterns

Release planning patterns are proposed in order to improve planning quality by making use of results of successful experiences in this field. Considering the need to employ a well-defined structure for such patterns to describe them in this framework, a set of characteristics is proposed to be used to show features of software development in the structure of release planning patterns. The framework contains specifications required for distinguishing release planning patterns. This framework includes:

- **Pattern name**

Shows the selected name for release planning pattern. Every pattern has a main name distinguishing it from other patterns.

- **Side name**

Every pattern may have one or more side names besides it main name. The side name may be used for pattern application in certain fields.

- **Pattern problem**

Pattern problem describes specifications of a problem or complexity in which the pattern can be applied. The pattern objective is usually presented in pattern problem in a explicit manner. The objective specifies pattern application. The problem is different in every certain step.

- **Context**

This determines a certain filed in which the considered pattern is employed. It can be release planning or every step of release planning general process.

- **Constraints**

It describes different parameters and instances or various steps of release planning. One or more parameters are considered in this pattern for each of which an instance is mentioned. Constraints are also presented by parameters. Pattern problem is solved respecting these parameter instances.

- **Solution**

This defines the selected method to solve the problem. The method is proposed considering the problem (pattern objective) and parameters (constraints) and considers and meets all instances.

- **Resulting context**

It describes possible states and situations after pattern implementation including results (good or bad goals) of running the pattern, other problems and patterns likely to occur in the new context.

- **Rationale**

It describes reasons of selecting the proposed problem solution. In fact, this part explains how a release planning pattern really works, why it works and why it is good. The rationale must specify how every constraint's parameter instance is met by the proposed solution of the pattern.

- **Known use**

Previous and known uses of the pattern are presented in this section.

- **Related sub-patterns**

A release planning pattern can include patterns of every stage. Therefore, this characteristic presents the names of patterns related to every certain stage.

The structure can be used for each pattern of various stages of release planning.

## IX. Development Method of Release Planning Patterns

Development of various release planning patterns is one of the most important factors considered in this area. Similar to software development patterns obtained based on previous experiences the most basic method to develop a release planning pattern is to make use of best experiences in release planning area. Since the issue is being proposed for the first time, no documentation is available on release planning experiences especially on release planning parameters.

In the first methods, as observed in customization of stages of release planning general process, only some limited investigations are done to categorize effective parameters on requirements prioritization and other stages lack such explorations. Hence, development of requirements prioritization patterns tries to make best use of these investigations. In the second method, different parameters and states they generate are implemented and experienced in several projects and different software development companies. Thus, case studies are selected so that they cover different states and instances of parameters appropriately. Patterns developed in this way are generated based on executive realities and as an example of method implementation. Moreover, it is possible to develop more patterns.

## X. Conclusion

Pattern-based release planning employs the idea of using release planning general process and its customization. The use of release planning general process enables making use of patterns in all projects and removes limitations from their applicability. The parametric nature of this process helps determining patterns based on specific rules and principles (parameters themselves). Release planning is facilitated using planning patterns. Moreover, release planning patterns transfer knowledge and experiences and provide higher customization capacity and this is of great significance for those companies developing different software projects.

Developing and generating release planning patterns can be considered as one of the most important tasks for future. Besides, developing parameters, instances and determination indicators of the exact method of resource estimation and release planning may be an important future task which helps software developing teams significantly.

## References

[1] A. J. Bangnal, V. J. Rayward Smith, and I. M. Whittley, "The next release problem," *Information and Software Technology*, vol. 43, pp. 883-890, December 2001.

[2] A. Seyed Danesh, R. Ahmad, M. R. Saybani, and A. Tahir, "Companies approaches in software release planning – based on multiple case studies," vol. 7, 2012.

[3] P. Carlshamre, "Release planning in market-driven software product development: provoking an understanding," *Requirements Engineering*, vol. 7, pp. 139-151, September 2002.

[4] A. Seyed Danesh, "A survey of release planning approaches in incremental software development," in *Computational Intelligence and Information Technology*, V. V. Das and N. Thankachan, Eds., Pune, India: Springer Berlin Heidelberg, 2011, pp. 687-692.
[5] J. Li and G. Ruhe, "Web-based decision support for software release planning," in *Proc. Workshop on Applications, Products and Services of Web-Based Support Systems*, Halifax, Canada, 2003, pp. 13-20.
[6] G. Ruhe and O. Saliu, "The art and science of software release planning," *IEEE Software*, vol. 22, pp. 47-53, November-December 2005.
[7] K. Bittner and I. Spence, *Managing Iterative Software Development Projects*, Addison-Wesley, 2007.
[8] M. Svahnberg, T. Gorschek, R. Feldt, R. Torkar, B. S. Saleem, and M. U. Shafique, "A systematic review on strategic release planning models," *Information and Software Technology*, vol. 52, pp. 237–248, March 2010.
[9] C. Alexander, S. Ishikawa, and C. A. M. S. Sara Ishikawa, *A Pattern Language: Towns, Buildings, Construction*, Oxford University Press, 1977.
[10] K. Beck *et al*., "Using pattern languages for object-oriented programs," Tektronix, Inc.1987.
[11] J. O. Coplien, *Advanced C++ Programming Styles and Idioms*, Addison-Wesley Pub. Co., 1992.
[12] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995.

**Amir Seyed Danesh** is a PhD student at University of Malaya, Kuala Lumpur, Malaysia since 2008 in the Department of Software Engineering. He received the M.Sc degree from Shahid Beheshti University, Tehran, Iran in software engineering in 2006. He has published papers in several journals and conferences. He worked as software engineer for over 5 years in Iran .His interested in requirements engineering, software release planning, software quality and software improvements.

**Rodina Ahmad** is an associate professor in software engineering and information systems at the Faculty of Computer Science and Information Technology, University of Malaya. She teaches information systems and software engineering modules at both the undergraduate and master levels. She holds a degree in computer science and mathematics from Hartford, Conn. Her master degree was from rensselaer Polytechnic Institute, USA. Her PhD degree in information systems is from National University of Malaysia.

**Seyed Mahdi Zargarnataj** graduated from Shahid Beheshti University in 2006 with a master's degree in software engineering. He works in the ISA Research Center at the Shahid Beheshti University. Also he developed software architecture in some software development projects. Currently he works as software team manager in the railway interlocking project.