The Art and Science of Software Release Planning

Günther Ruhe
Moshood Omolade Saliu

University of Calgary
Laboratory for Software Engineering Decision Support

Abstract

Release planning (RP) addresses decisions related to the selection and assignment of features to a sequence of consecutive product releases such that the most important technical, resource, budget and risk constraints are met. Release planning is an important and integral part of any type of incremental product development. Poor release planning decisions can result in: (i) unsatisfied customers not getting what they expect, (ii) release plans unlikely to be delivered within a given schedule, quality and effort constraints not being met, and (iii) release plans not offering the best business value for the investment.

The objective of this paper is to describe and position the 'art and science' of software release planning. The "art of release planning" refers to relying on human intuition, communication and capabilities to negotiate between conflicting objectives and constraints. The "science of release planning" refers to emphasizing formalization of the problem and applying computational algorithms to generate best solutions.

Both art and science are important for achieving meaningful release planning results. We investigate the release planning process and propose a hybrid planning approach that integrates the strength of computational intelligence with the knowledge and experience of human experts.

Planning Releases for Incremental Software Development

Incremental development allows customers to receive part of a system early. This allows for creating early value and providing early feedback. Each release of the system is a collection of features that forms a complete system that will be of value to the customer. New releases also serve to fix defects detected in former product variants.

Release planning (RP) addresses all decisions related to the selection and assignment of features to a sequence of consecutive product releases. In application of Pfleeger's [13] definition of good decisions in software engineering, we assume that a good release plan should be characterized by the following criteria:

- It provides maximum business value by offering the best possible blend of features in the right sequence of releases.
- It satisfies most important stakeholders involved.
- It is feasible in terms of the existing resource capacities available.
- It reflects existing dependencies between features.

Difficulties of Release Planning

To find good release plans is an inherently difficult problem. The main reasons for that are the computational and cognitive complexities of the problem. The computational complexity has to do with the difficulty of determining a good release plan once you have found an appropriate formal description of the problem. However, finding this description alone is difficult (and relates to cognitive complexity). In order to determine a good release plan, you need a proper understanding of what the objectives and constraints for planning are. You also need to know who the key stakeholders are, how important they are, and what their preferences are in terms of candidate features. The information for all of these factors is uncertain and dynamically changing. This makes release planning a very difficult problem. But does this mean that there is no alternative to ad hoc and intuition based planning?

Experience reports on release planning practices and requirements prioritization are rare. Our comparative analysis of existing methods results in the following observations:

- RP is typically done ad hoc and not based on sound models and methodologies. This is even the case when planning involves several hundred features. Karlsson and Ryan [7] and Lehtola et al. [8] observed during their industrial studies that most organizations select requirements informally.
- Industrial experiments by Regnell et al. [14] and Carlshamre [3] concluded that the task of selecting and scheduling requirements in an optimal way is not only important but inherently complex. We argue that such optimality cannot be simply achieved by using ad hoc planning methods.

- Planning and follow-up re-planning is a time-consuming process. Time is mainly devoted to elicit the relevant information and to negotiate compromises between stakeholders [1], [10].
- The scope of planning is often limited to just the next release [3].

The project management process area as defined in CMMI [9] covers activities related to planning, monitoring, and controlling projects. According to the CMMI, the purpose of project planning is to establish and maintain plans that define project activities. These include developing the project plan, involving stakeholders appropriately, obtaining commitment to the plan, and maintaining the plan. For all these processes, there are existing guidelines and standards (e.g. IEEE/EIA 12207.0) on how planning should be performed "in principle". However, not very much is known about how to actually perform this process in an effective and efficient manner. The standards and guidelines do not give the proper answers on how to operationally assign features to releases to ensure maximal business value.

We will present two approaches on planning releases - one focuses more on human intuition and communication (called 'the art of release planning'), and the other approach relies more on a systematic, repeatable methodology and meaningful data (called 'the science of release planning'). We will now present these two approaches and how they are related to each other.

The Art of Release Planning

The art of release planning focuses on human intuition, communication and human capabilities to decide which features should be selected and released in which release. One justification for this approach can be the belief that release planning is an ill-defined problem. For an ill-defined problem it is not clear what the problem is, and therefore, what the solution is. Another reason for handling release planning as an art can be an organization's lack of emphasis on processes in general. This can be intentional or just the indication of a lack of RP process maturity.

Small releases are one of the key principles of extreme programming [11]. This approach just leverages the well known advantages of incremental development to avoid the "big bang" syndrome and to receive early customer feedback. RP in extreme programming focuses on planning for the next iteration, and also considers this activity more of an art than a science. The planning procedure relies on physical meetings between the most important stakeholders (if available) to discuss and negotiate informally which features should be developed next. This includes estimation of required effort. The number of features to be decided upon is typically small. According to [11], extreme programming does not provide guidance on how to decide upon features and priorities when multiple stakeholders are involved. It also does not suggest techniques to balance conflicting demands between multiple stakeholders.

Even in more plan-driven environments, release planning is often done quite informally, relying on elementary tools such as spreadsheets. The release decisions are made by trying to contact the main stakeholders and manually balancing their interests and preferences with the resources available. However, exponential growth in the number of possible release plans surpasses the power of manual plan generation, especially as the number of features and stakeholders grow larger.

The Science of Release Planning

The science of release planning is primarily based on the belief that the problem can be (at least approximately) formalized, and that there are meaningful results from solving the formalized problems. Some researchers have modeled the problem as a specialized optimization problem. In formulating an optimization model for RP, Bagnall et al. [2] assign weights to customers based on their importance to the software company. The objective is to find a subset of customers whose features are to be satisfied within the available cost. Jung [6] follows similar footing with a goal of selecting features that give maximum value for minimum cost and within allowable cost limits of a software system. These optimization approaches cope better with larger problem sizes, but customers are not given the opportunity to participate in RP decisions. They both also plan just a single release ahead.

Attempts at formalizing certain aspects of the RP problem while allowing most part to be carried out ad hoc include the incremental funding method (IFM) by Denne and Cleland-Huang [5] and maintenance planning by Penny [12]. IFM differs from other RP approaches because of the focus on revenue projections, but other RP constraints are not considered. Penny's approach focuses on release monitoring by ensuring balance between required effort and available effort. The science part of these models does not formalize the major impacting factors in RP.

The formulation we are presenting in this paper extends the former formulation given in [15] in terms of offering higher flexibility in the number of releases to be addressed. However, for simplicity reasons we only present and illustrate the case of K = 2 releases.

Decision Variables
Suppose we have a collection of features F = {f(1),f(2), … ,f(n)}. This set of features may be related to new functionalities, change requests by customers, or defect corrections. For the sake of simplicity, we generally refer to all three categories as a set of features. The final goal is to assign the features to a finite number K of release options. This is described by decision variables {x(1),x(2), …,x(n)} with x(i) = k if requirement i is assigned to release option $k \in \{1,2,\ldots K\}$ (and x(i) = 0 otherwise).

Dependencies between Features
As pointed out by Carlshamre [4], there are a variety of possible dependencies between features, and the relative portion of features involved in such dependencies is high. In our formulation, two types of dependency constraints are considered: Features can be in a coupling relation called C, or in a precedence relation called P. Coupled features are supposed to be released jointly as they are dependent on each other. Similarly, if it does not make sense to offer a feature in a release earlier than another feature is released, we have a precedence relationship.

Resource Constraints
Resources are an essential aspect for release planning. For feature realization, project managers typically have to consider various resource constraints. Most frequently, these constraints are related to different types of resources - budget or effort consumption. All of these constraints include bounds on the maximum capacities available for each resource type.

We assume T resource types and capacities Cap(k,t) related to all releases k = 1…K and all resource types t = 1…T. Every feature f(i) requires an amount r(i,t) of resources of type t. Thus, each release plan x assigning feature f(i) to release k expressed as x(i) = k must satisfy the following constraints:

$$\sum_{x(i)=k} r(i,t) \le Cap(k,t) \qquad (1)$$

for all releases k and all resource types t

Stakeholders
Stakeholders are extremely important in product planning. In reality, there are differences in the importance of stakeholders. We assume here a set of stakeholder S = {S(1),…,S(q)}. Each stakeholder p can be assigned a relative importance $\lambda(p) \in \{1,..,9\}$. We use an ordinal nine-point scale here to allow sufficient differentiation in the degree of importance. Therein,

- $\lambda(p) \in \{1,3,5,7,9\}$ means very-low/low/medium/high/extremely-high importance, respectively.

The interpretation of the even numbered values (i.e. $\lambda(p)$ = 2, 4, 6, 8) is assumed to be between the preceding and succeeding odd number values.

Prioritization of Features
Prioritization of features can be done based on different criteria. In our formulation, priority can be expressed in relation to:

- Value (how valuable is this feature?)
- Urgency (how urgent is the feature?)

Both evaluations can be done on a nine point scale again. For the value proposition, each stakeholder is asked to assign an ordinal value, value(p, i) $\in$ {1, 2,…,9} to each feature based on the assumed (relative) impact of the feature to the overall value of the product. Lowest and highest value is expressed by value(p, i) = 1 and value(p, i) = 9, respectively.

Urgency can be motivated by time-to-market concerns of certain product requirements. For simplicity reasons, we assume here the planning of two releases ahead of time. In that case, each stakeholder has nine votes available for each feature, and the votes are to be distributed among the three possible options (release 1, release 2, or postpone the feature). The higher the number of votes assigned to option k, the more satisfied the stakeholder would be if the feature is put in the respective option. Consequently, an urgency vote urgency(p,i) = (9,0,0) expresses highest possible urgency assigned by stakeholder S(p) to feature f(i) to enable it appear in the first release, urgency (0,9,0) would be a strong indication that the feature is desired in the second release, and urgency (0,0,9) would be a very strong suggestion to postpone this feature. An urgency vote of (3,3,3) would express no urgency preference. Any combination of the three cases expresses different degrees of these three basis options.

Objective Function
You cannot generate good plans without a proper understanding what the objective of planning is. Typically, the objective is a mixture of different aspects such as value, urgency, risk, satisfaction/dissatisfaction or return on investment.

The actual form of the explicit function tries to bring the different aspects together in a balanced way. In our model, we assume the following:

- An additive function with the total objective function value determined as the sum of the weighted average satisfaction WAS(i,k) of stakeholder priorities for all features f(i) when assigned to release k.

- Each value WAS(i,k) is determined as the weighted average of the products of the two dimensions of prioritization described above.
- The degree of impact of the stakeholder S(p) is determined by the relative importance $\lambda(p)$.
- For each release option k, normalized parameters $\xi_{(k)}$ describe the relative importance of each option.

According to these assumptions, the objective is the maximization of a function F(x) among all release plans x subject to the satisfaction of resource constraints (described in Equation 1) and dependency constraints stated above. F(x) is given as:

$$F(x) = \sum_{k=1\ldots K} \sum_{i:\, x(i)=k} WAS(i,k) \qquad (2)$$

where

$$WAS(i,k)=\xi(k)[\Sigma_{p=1\ldots q}\, \lambda(p)\cdot value(p,i)\cdot urgency(p,i,k)] \qquad (3)$$

Solution Approach based on Integer Linear Programming

The above formalized description allows the application of efficient optimization algorithms offering optimal or nearly optimal solutions. This is a fundamental difference to simple spreadsheet calculations aimed to support a series of elementary arithmetical operations. As we have seen, the problem is by far too complex to achieve (and guarantee) release plans of a certain level of quality by using just spreadsheets.

The given formulation of the release planning problem constitutes a specialized integer linear programming problem. All the stated objectives and constraints are linear functions, and the decision variables are integers. This problem belongs to the class of computationally most difficult problems. General purpose algorithms are available to solve small to medium sized problems [16].

However, the structure of the problem is specialized and allows the application of more efficient solution techniques. The solution algorithms we have adopted are based on solving a sequence of linear programming problems without integer conditions. These relaxed problems are well understood and can be efficiently solved. This approach is combined with heuristics to speed up the process and to generate not just one, but a set of sufficiently good solutions. A solution generated is considered to be sufficiently good (also called 'qualified') in terms of the objective function value if it achieves at least 95% of the maximum objective function value. At this point the argument is that it is more practical to offer a small set of qualified solutions more or less of the same quality, but being different in their structure.

The Art and Science of Release Planning

Art and science based approaches to release planning are not contradictory but complementary. Both approaches have their own justifications. While the art approach finds it challenging to cope with the complexity of the RP problem as the number of factors grows larger, the science approach copes better with complexity, but lacks the ability to completely encode the equivalent of the analytical mind with which the human decision-maker evaluates problem situations.

What we are proposing is to create a synergy between the art and science of release planning by integrating computational intelligence applied to the formalizable portion of the problem, with human intelligence addressing the implicit and tacit aspects of release planning.

The overall approach shown in Figure 1 is a high level framework stressing the continuous process needed to perform planning and re-planning. Instead of determining just one plan for a vaguely defined problem, a sequence of problems is solved generating a set of 'good enough' alternative solutions. These solutions are presented to a human decision-maker (say the project manager) who evaluates the alternatives based on experience and familiarity with the problem context. We further refine this hybrid approach by giving a description of the main RP tasks to be performed during the three phases of the planning process: Modeling, Exploration and Consolidation.

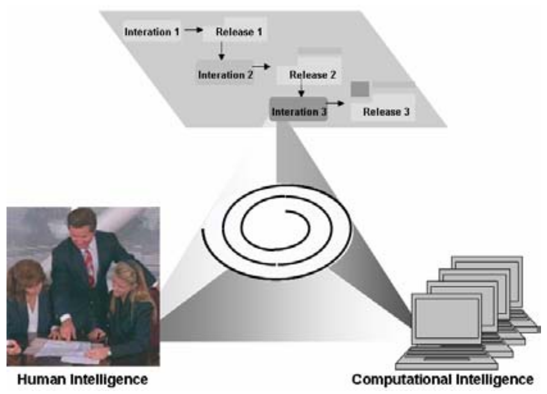

Figure 1: The hybrid release planning framework

Phase 1 – Modeling

The top of the triangle addresses the conceptualization of the problem. It focuses on the formal description of the dynamic real world to make it suitable for computational intelligence based solution techniques. The main tasks of this phase include:

Planning objectives and constraints: This task involves the formalization of the objective function that models the underlying problem and all the constraints present. The process of realizing this task is what we discussed during the instantiation of the science-based modelling of RP above. All impacting factors and prioritization schemes are contained in the objective function and constraints.

Stakeholder voting: Stakeholders are the people who directly or indirectly influence or are influenced by the software release plans. They may include developers, managers, customers, and users. Giving stakeholders the opportunity to vote on features according to their preferences is important because they set the course of the project and decide the evaluation criteria for its success. The result of this activity is a set of priorities assigned to features according to stakeholders preferences.

Resource Estimation: Estimates of the likely amounts of resources of different types needed to implement each feature must be determined during the modelling phase, since solution to the objective function during exploration depends on it. Resource estimation in the form of effort, cost, and schedule has always being a major challenge in software engineering. One of the reasons for this is the fact that estimates made at an early stage in the development life-cycle are generally fraught with uncertainties, because little information is known about the project that early. However, hybrid approaches that integrate judgment of human experts with formal techniques have proven most promising in this situation.

Phase 2 – Exploration

The actual solution plan generation based on the formal model is done here. Because of the sophistication of the model, there is a need for efficient special-purpose solution techniques. In our case, specialized integer programming algorithms were developed to explore the solution space and generate solution alternatives.

Phase 3 – Consolidation

The solution alternatives generated by the computational algorithms are presented to the decision maker for evaluation. The decision maker should be able to investigate the current solution alternatives and analyze the solutions based on experience and problem context. This contributes to the understanding of the problem and may result in modification of parts of the underlying model or in some local decisions (e.g. pre-assigning some features to specific releases). Typically, these decisions reduce the size and complexity of the problem for the next iteration. Several iterations are possible until a desirable solution alternative is derived.

Scenario-playing and Re-Planning: Having invested into the formalization of the problem, one is now in the position to perform what-if scenarios:

- What happens if some of the stakeholder weights are changed?
- What happens if some of the capacities are increased (or are decreased)?
- What happens if the objective function and/or some of the constraints are modified?

Illustration of the Proposed Approach

Sample Problem Description

In what follows we will illustrate some key ideas of the proposed approach. For that purpose, we present a sample project with key project data summarized in Table 1.

- The trial project encompasses 15 features to be planned for the next two releases - release 1 and 2.
- The relative importance of release 1 and release 2 are given as $\xi(1) = 0.7$ and $\xi(2) = 0.3$, respectively. That means there is a focus on the importance of release 1 against release 2.
- For simplicity, only two stakeholders S(1) and S(2) are involved in the prioritization of features. Their degree of importance is $\lambda(1) = 4$ and $\lambda(2) = 6$, respectively.
- There are four resource types involved in the realization of the features. Each of them has total capacity bounds for the two release periods. We observe that the total demand of all features exceeds the total capacity. This means that not all features can be implemented in the next two releases.
- There are three coupling and five precedence constraints:

  $\{(7,8), (9,12), (13,14)\} \in C$ and

  $\{(2,1), (5,6), (3,11), (8,9), (13,15)\} \in P$.
- For example, $(7,8) \in C$ means that feature f(7) and f(8) have to be in the same release.
- $(2,1) \in P$ means that f(2) must be implemented before f(1), although both could be implemented in the same release.

Stakeholder Urgency Voting

To illustrate urgency voting, let us consider feature f(1). Stakeholder S(1) voted urgency(1,1) = (5,4,0) which implies that the feature is desired in release 1 but could as well come in release 2. On the same feature, S(2) has voted urgency(2,1) = (0,3,6) which implies the desire to postpone the feature with some acceptance also for assignment to release 2. However, S(2) is strongly against having f(1) in release 1.

Computation of the Objective Function Value
Given the above data, how do we compute the objective function value for each release? The objective function value depends on the release each feature is scheduled for.

Suppose feature f(1) is scheduled in release 1, then we have the following situation:

S(1): value(1,1) = 6, urgency(1,1,1) = 5, $\lambda(1) = 4$
S(2): value(2,1) = 2, urgency(2,1,1) = 0, $\lambda(2) = 6$
Using $\xi(1) = 0.7$, WAS(1,1) is computed as:
WAS(1,1)= 0.7[(4 x 6 x 5) + (6 x 2 x 0)] = 84.0

If f(1) is scheduled in release 2, then we have:
S(1): value(1,1) = 6, urgency(1,1,2) = 4, $\lambda(1) = 4$
S(2): value(2,1) = 2, urgency(2,1,2) = 3, $\lambda(2) = 6$

Using $\xi(1) = 0.3$, WAS(1,2) is computed as:
WAS(1,2)= 0.3[(4 x 6 x 4) + (6 x 2 x 3)] = 39.6

Just looking locally at these two options, it would be more favourable to assign f(1) to release 1. As the example in Table 1 shows, there are 15 features to be assigned. Thus, for each release plan, the F(x) sums the WAS according to the assignment of all features. The goal as formulated in (2) is to maximize the result.

Table 1: Features, resource consumption and stakeholders' evaluation of features

		RESOURCES				**STAKEHOLDER S(1)**		**STAKEHOLDER S(2)**	
		Analyst & Designers (person-hours)	**Developers (person-hours)**	**Quality Assurance (person-hours)**	**Budget ($ '000)**	**Value**	**Urgency**	**Value**	**Urgency**
ID	**Feature Name**	**r(i,2)**	**r(i,2)**	**r(i,3)**	**r(i,4)**	**value(1,i)**	**urgency(1,i)**	**value(2,i)**	**urgency(2,i)**
1	Cost Reduction of Transceiver	150	120	20	1000	6	(5, 4, 0)	2	(0, 3, 6)
2	Expand Memory on BTS Controller	75	10	8	200	7	(5, 0, 4)	5	(9, 0, 0)
3	FCC Out-of-Band Emissions	400	100	20	200	9	(9, 0, 0)	3	(2, 7, 0)
4	Software Quality Initiative	450	100	40	0	5	(2, 7, 0)	7	(7, 2, 0)
5	USEast Inc. Feature 1	100	500	40	0	3	(7, 2, 0)	2	(9, 0, 0)
6	USEast Inc. Feature 2	200	400	25	25	9	(7, 2, 0)	3	(5, 4, 0)
7	China Feature 1	50	250	20	500	5	(9, 0, 0)	3	(2, 7, 0)
8	China Feature 2	60	120	19	200	7	(8, 1, 0)	1	(0, 0, 9)
9	12 carrier BTS for China	280	150	40	1500	6	(9, 0, 0)	5	(0, 8, 1)
10	Pole Mount Packaging	200	300	40	500	2	(5, 4, 0)	1	(0, 0, 9)
11	Next Generation BTS	250	375	50	150	1	(8, 1, 0)	5	(0, 7, 2)
12	India BTS Variant	100	300	25	50	3	(9, 0, 0)	7	(0, 6, 3)
13	Common Feature 01	100	250	20	50	7	(9, 0, 0)	9	(9, 0, 0)
14	Common Feature 02	0	100	15	0	8	(9, 0, 0)	3	(6, 3, 0)
15	Common Feature 03	200	150	10	0	1	(0, 0, 9)	5	(3, 6, 0)
	Total resource consumption:	**2615**	**3225**	**392**	**4375**				
	Available capacity - Release 1	**1300**	**1450**	**158**	**2200**				
	Available capacity - Release 2	**1046**	**1300**	**65**	**1750**				

Let us assume that we have determined the two qualified release plans shown in Table 2. Both plans are within the 95% quality range with respect to the objective function. As an added value, we observe that the two plans are structurally different which gives some additional flexibility for the decision-maker.
Plans like those generated below derive from the power of scientific modeling of the problem. Incorporating the art into it would encompass the presentation of such qualified plans to the decision-maker to evaluate and make decisions based on knowledge and experience.

Summary

A marriage of art and science, such as the one we have proposed, promises to carry the state-of-the-practice to a higher level where due consideration would be given to the following in RP:

- Most organizations would realize the need for a more sophisticated RP methodology rather than the predominantly ad-hoc methods.
- Introducing more formalism that allows planning of projects containing several hundred features with ease.
- Formulating release planning objectives with several impacting criteria rather than concentrating on the values of features only.
- Ability to allow human decision-makers to easily evaluate formally generated release plans, versus dealing with lots of information and little visibility into the structure of the problem.
- Pro-active evaluation of possible planning strategies to better understand the impact of varying problem parameters.

Table 2: Two qualified release plan alternatives

f(i)	Feature Name	Release Plan x1		Release Plan x2	
		x1(i)	WAS(i,k)	x2(i)	WAS(i,k)
01	Cost Reduction of Transceiver	1	84.0	1	84.0
02	Expand Memory on BTS Controller	1	287.0	1	287.0
03	FCC Out-of-Band Emissions	1	252.0	3	0
04	Software Quality Initiative	3	0	1	233.8
05	USEast Inc. Feature 1	1	134.4	3	0
06	USEast Inc. Feature 2	2	516.6	3	0
07	China Feature 1	2	277.2	1	88.2
08	China Feature 2	2	43.2	1	19.6
09	12 carrier BTS for China	3	0	2	72.0
10	Pole Mount Packaging	3	0	3	0
11	Next Generation BTS	3	0	3	0
12	India BTS Variant	3	0	2	75.6
13	Common Feature 01	1	37.8	1	516.6
14	Common Feature 02	1	8.4	1	277.2
15	Common Feature 03	2	54.0	2	54.0
Objective function value F(x)		**1694.6**		**1708.0**	

Conclusions

The decision support approach presented in this paer is implemented as part of the system solution ReleasePlanner® (www.releaseplanner.com). Our on-going efforts are geared towards empirical studies to collect and analyze quantitative and qualitative measures in order to assess the added value of the proposed technique in industrial settings. We have initiated this research through two pilot industrial case studies reported in [1] and [10]. A detailed discussion of such empirical study results should be substantial enough to constitute the topic of a separate paper on its own.

Acknowledgment

The authors would like to thank the Alberta Informatics Circle of Research Excellence (iCORE) for its financial support of this research. Many thanks are due to the anonymous reviewers for their detailed and valuable comments that helped in improving the draft.

References

[1] Amandeep, A., Ruhe, G. and Stanford, M., "Intelligent Support for Software Release Planning", Proceedings of PROFES'2004, Lecture Notes on Computer Science, vol. 3009, pp 248-262, 2004.

[2] Bagnall, A. J., Rayward-Smith, V. J., and Whittley, I. M., "The Next Release Problem," Information and Software Technology, 43 (14), pp. 883-890, 2001.

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

[4] Carlshamre, P., Sandahk, K., Lindvall, M., Regnell, B., and Nattoch Dag, J., "An Industrial Survey of Requirements Interdependencies in Software Release Planning," In Proceedings of the 5th IEEE International Symposium on Requirements Engineering, pp. 84-91, 2001.

[5] Denne, M. and Cleland-Huang, J., "The Incremental Funding Method: Data Driven Software Development," 21(3): pp. 39-47, 2004.

[6] Jung, H.-W., "Optimizing Value and Cost in Requirements Analysis," IEEE Software, 15(4): pp. 74-78, 1998.

[7] Karlsson, J. and Ryan, K., "Prioritizing Requirements using a Cost-Value Approach," IEEE Software, 14(5): pp. 67-74, 1997.

[8] Lehtola, L., Kauppinen, M., and Kujala, S., "Requirements Prioritization Challenges in Practice", Proceedings PROFES'2004, Lecture Notes on Computer Science, Vol. 3009, pp 497-508.

[9] M.GCMMI Product Team, "Capability Maturity Model Integration (CMMI®) Version 1.1 Staged Representation," Carnegie Mellon University, Pittsburgh, PA, Mar. 2002.

[10] Momoh, J. A., "Applying Intelligent Decision Support to Determine Operational Feasibility of Strategic Software Release Planning", Masters Thesis, University of Calgary, Canada, 2004.

[11] Nejmeh, B. A. and Thomas, I., "Business-Driven Product Planning Using Feature Vectors and Increments," IEEE Software, 9(6): pp. 34 – 42, 2002.

[12] Penny D. A., "An Estimation-Based Management Framework for Enhancive Maintenance in Commercial Software Products," Proceedings of International Conference on Software Maintenance (ICSM): pp. 122-130, 2002.

[13] Pfleeger, S., "Making Good Decisions: Software Development and Maintenance Projects", Tutorial at 8th IEEE Symposium on Software Metrics, Ottawa, Canada, June 4-7, 2002.

[14] Regnell, B., Beremark, P., and Eklundh, O., "A Market-driven Requirements Engineering Process – Results from an Industrial Process Improvement Programme," Requirements Engineering, 3(20: pp. 121-129, 1998.

[15] Ruhe, G. and Ngo-The, A., "Hybrid Intelligence in Software Release Planning," International Journal of Hybrid Intelligent Systems, 1(2004): pp. 99-110, 2004.

[16] Wolsey, LA and Nemhauser, GL (1998) Integer and Combinatorial Optimization. New York, John Wiley

Authors

Dr. Ruhe holds an Industrial Research Chair in Software Engineering at University of Calgary and is an iCORE Professor since July 2001. His main results and publications are in software engineering decision support, software re-lease planning, requirements and COTS selection, measurement, simulation and empirical research. From 1996 until 2001 he was deputy director of the Fraunhofer Institute for Experimental Software Engineering. He is the author of two books, several book chapters and more than 120 publications. Dr. Ruhe is a member of the ACM, the IEEE Computer Society and the German Computer Society GI.

Omolade Saliu is a PhD candidate and an iCORE scholar in Computer Science Department at the University of Calgary, Canada. His research interests include software metrics and measurement, software engineering decision support, software process-related issues and soft computing. Omolade holds a Bachelor of Technology (B.Tech.) degree in Mathematics/Computer Science from the Federal University of Technology, Minna, Nigeria (1998). He obtained his Master of Science degree in Computer Science from King Fahd University of Petroleum & Minerals, Saudi Arabia (2003). He is a member of the IEEE Computer Society.