Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 16 (2013) 1062 – 1071

Procedia Computer Science

Conference on Systems Engineering Research (CSER'13)
Eds.: C.J.J. Paredis, C. Bishop, D. Bodner, Georgia Institute of Technology, Atlanta, GA, March 19-22, 2013.

Representing Advances in Systems Engineering by Using an Electronic Process Guide

Thammanoon Kawinfruangfukul, Supannika Koolmanojwong*, Nupul Kukreja

[a]*Center for Systems and Software Engineering, University of Southern California, Los Angeles, USA*

Abstract

Systems Engineering is a discipline that never stops growing. Systems Engineers are just like students or life-long learners who have to keep themselves up to date with new best propositions. With recent additions of novel concepts in systems engineering such as Social Networking concepts, Value-Based Requirements Prioritization, and Program Model, it is often challenging to translate these concepts into practices. An Electronic Process Guide (EPG) for Incremental Commitment Spiral Model (ICSM) aims to define clear systems engineering practices by providing necessary roles, tasks, step-by-step practical guidance to perform each task, and input, output, and supporting artifacts. This paper introduces the advances in systems engineering field and reports on how this EPG helps in expediting the learning process and supports the process authors in continuous maintenance and enhancement of the process guide.

Keywords: Systems Engineering; Electronic Process Guide; Value-Based Requirements Prioritization; Social Networking

1. Introduction

Nowadays many new concepts in systems engineering are emerging. To help organizations succeed in the current business environment and provide better services to all stakeholders, systems engineers need to continuously learn new disciplines and practices. However, understanding and putting these disciplines or practices into practice may lead to a high learning curve. It is essential for a process user to have access to a knowledge management repository with practical guidelines to facilitate and to come up the learning curve in understanding new systems engineering concepts. It is also important for the process engineers or process authors to have a tool that can effectively describe the processes to the process users. The Electronic Process Guide (EPG), an online knowledge representative framework is a useful tool for presenting up-to-date practices in a well-structured web representation. Having many

* Corresponding author. Tel.: +1-213-740-5931; fax: +1-213-740-4927.
E-mail address: koolmano@usc.edu.

doi:10.1016/j.procs.2013.01.112

versions of process guide can confuse the process users thereby misunderstanding the process. With dynamic content representation and easy-to-maintain feature, the EPG allows process users to access the same information and prevent these problems. In addition, clearly-defined roles and responsibilities in the EPG enable systems engineers to select and focus on necessary concepts and practical steps of each role to perform.

The paper is organized into six sections. Following this introduction, section 2.1 introduces the EPG and elaborates on how the EPG can help systems engineers in communicating among their team members. In section 2.2, we present the overview of systems and software engineering courses taught at the University of Southern California. Section 3 discusses the advances in systems engineering field that we have incorporated in our software engineering course, such as Program Model, Value-Based Requirements Prioritization, and WinBook. In section 4, we present how the EPG represent the process elements and their relationship. Section 5 reports the feedback of process users and process authors followed by conclusion and future works in section 6.

2. Background

In this section, we will introduce the EPG, a tool used to produce the EPG, and the course where we have the EPG as the primary process guideline.

2.1. Electronic Process Guide

EPG is a structured, workflow-oriented, electronic reference document for a particular process. It provides assistance to process authors in defining the process and process users in performing the process [1]. Basically, EPG provides all the process elements information and their relationships in electronic form. It is able to show process diagram in graphical way and makes extensive use of hyper-links to support the flexible navigation. In addition, it may store examples and templates for the process performers to download. The IBM Rational Method Composer (RMC) is a process management platform that integrates best practices of the Rational Unified Process (RUP) framework and provides process content library, delivery processes, and capability patterns allowing for process engineers to author, configure, view, and publish one's software development process [2]. There are two main purposes of the RMC [3]. Firstly, the RMC acts as a content management system that people can store, maintain, and publish their knowledge base of process contents to development practitioners. Secondly, its purpose is to provide a tool for process engineers to select, tailor, and assemble process contents that fit to their specific development projects. Currently we have developed an electronic process guide for the Incremental Commitment Spiral Model (ICSM) [4]. This ICSM EPG has been used in our Software Engineering class to help students coming up with their learning curve.

Systems Engineers need to learn to adapt and adopt new practices all the times. They need to continuously update their knowledge. Various new concepts are incrementally integrated in their process. Hence, a systems engineering process should support problem understanding and communication among all success critical stakeholders, such as systems engineers and clients at all phases [4]. The ICSM EPG provides systems engineers the effective ways in supporting the communication about the project progress among their team members. The ICSM EPG helps systems engineers in following aspects:

- Team synchronization prevents misunderstanding among team members and reduces rework that might happen during project development. Roles, responsibilities, and work products are identified at every ICSM phase so that systems engineers can use the information to synchronize team and project progress.
- The ICSM EPG provides systems engineering practices, completion criteria for each practice, and templates of work products. System engineers can refer to the defined information to perform their responsibilities. If there are any conflicts happened in the project, systems engineers can use the ICSM EPG as a reference to support decision making and resolve conflicts.
- Well-defined roles, responsibilities, artifacts, tasks, and steps for each task provided in the ICSM EPG help systems engineers understand what to do and plan their schedule accordingly. Moreover, each role in the ICSM EPG is clearly defined and identified boundaries of responsibilities that each role should perform, so it prevents systems engineers from performing duplicated tasks and producing redundant work products. In addition, the ICSM EPG provides work breakdown structure view and primary and secondary performers of each task for systems engineers to communicate and collaborate in developing projects effectively. Moreover, role list in the

ICSM EPG can be considered as a checklist for planning and allocating project resources.

- The ICSM EPG supports in maintaining consistency among work products by providing templates and examples of work products.

2.2. Systems and Software Engineering Course

CSCI577ab [5] are the capstone software engineering project courses at the University of Southern California (USC)'s Master of Computer Science with Specialization in Software Engineering (MSCS-SE). The main objective of the course is to prepare students for software leadership careers through the 2040's. Software Engineering I or CSCI577a in the Fall semester focuses on software-intensive systems engineering, including system operational concept formulation, requirements negotiation and definition, prototyping, COTS and services evaluation and selection, system and software architecture definition, life cycle plans and processes, risk analysis, feasibility analysis, and verification and validation. Software Engineering II or CSCI577b in the Spring semester focuses on software product implementation, integration, test, documentation, transition, and maintenance with an emphasis on quality software production.

In the course, students work as a team to develop e-services projects for small businesses or nonprofit organizations. They perform various systems engineering roles such as Requirements Engineer, System Architect, Operational Concept Engineer, Verification and Validation Personnel, and Life Cycle Planner. In addition, students can apply systems engineering practices and produce artifacts, such as Operational Concept Description and System and Software Architecture Description. Moreover, working in teams allows students to develop their projects in real world environment by having client representatives, 5-6 on-campus students (co-located), and 1-2 remote students who are mainly working as professional systems engineers.

3. Advances in System Engineering Principles

Systems engineering is generally an interdisciplinary approach by nature. Various Systems Engineering concepts are borrowed from other disciplines. This paper, we introduce three novel concepts that are influenced by business, sociology and psychology background.

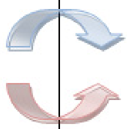
3.1. *Program Model*

The business model canvas [6] is commonly taught in business schools to help entrepreneurs and managers communicate the business model with ease. This model, however, is unsuitable for systems and software engineers owing to the differences in domains and nomenclature. However, the model structure is very helpful in understanding and capturing the high level vision of the 'project' via simple brainstorming. We created a Program Model that is derived from the Business Model Canvas and use it as an educational tool for understanding the high level vision i.e. why undertaking the particular project and what benefits the clients hope to get from it and what all needs to be done to realize those benefits, other than a straightforward system development.

In order to develop systems thinking, it is imperative to understand that just delivering an IT system does not help realize the benefits. There are various other initiatives like change management, advertising, training, data transformation and migration etc., that must be done to realize the value propositions. These set of initiatives along with system development form a 'Program' [7]. Understanding the concept of a Program, how it is different from traditional projects and what it brings to them is the first major step in developing systems thinking as well as for proactive benefits management; knowing that other initiatives are at least equally important developing and delivering an IT system.

The Program Model, as shown in Figure 1, has seven components: assumptions of the business under which the program is value adding, Success-Critical Stakeholders (SCS), Initiatives which are the key activities that must be undertaken in order to deliver the value propositions to the beneficiaries and help the beneficiaries realize value, Value Proposition which are benefits expected from the new system, Cost, and Benefits.

Assumptions (Under what *Business assumptions* will this *'model'* be true)			
Stakeholders (Who is accountable for the initiatives)	**Initiatives** (What to do to realize benefits)	**Value Propositions** (Benefits)	**Beneficiaries** (Who derives value)
• Who/what resources are required for 'executing' the initiatives • Do you need to 'partner' with another department or organization? • Do you need to hire anyone?	• What are the key activities that must be done to for delivering/realizing the value propositions?	• Why undertake this project/program? • What are the value propositions you seek to satisfy/serve?	• Usually the customers or end users • Can also be project sponsors
Cost (Cost factors) • What are the costs involved in executing this program? Ex.: Personnel Costs, Hardware/Software Costs, Office Rental, Equipment/infrastructure etc.		**Benefits** (Key performance indicators – KPIs) • Against what metrics will you track the benefits delivered? MEDIC (Maintain, Eliminate, Decrease, Increase or Create)	



Legend:

Initiatives that need to be undertaken to help beneficiaries **derive value** from the expected benefits/value propositions

Initiatives that need to be undertaken to help **deliver value** to the beneficiaries (i.e. "how" will the benefits reach the beneficiaries?)

Fig. 1. Program Model framework

Listing Program Model's components in bullet points enables all SCSs to identify and realize the benefits as well as connect all related components comprehensively and effectively. Blue and pink arrows in Program Model help in identifying additional non-technical initiatives, such as business aspects that may be needed to realize the potential benefits for purposed system. These may also help in identifying some additional SCSs who are part of the system and need to be represented to create shared visions of the system. The last two components are Cost which are the costs drivers for executing the program and Benefits which can be tangible and intangible benefits. These components are then elaborated further and used for Return On Investment Analysis.

The Program Model is a high level overview of the benefit chain or the result chain diagram. The development team can use the Program Model to brainstorm or clarify which SCSs are accountable for which initiatives and which initiatives contribute to value proposition. Creating benefit chain also helps identifying and missing SCSs or initiatives/ benefits that may have been left out of the Program Model.

3.2. WinBook

Winbook is a social networking influenced tool for collaborative requirements brainstorming and negotiation [8]. Winbook was developed after studying how people collaborate on Facebook and organize their emails using color-coded labels using Gmail. Non-technical stakeholder friendly methods for system definition and evolution have always been a significant challenge. The new avatar of a WinWin Negotiation Framework [9] sought to alleviate this problem and has provided significant affirmations of the same [8].

As shown in Figure 2, the win conditions are captured on the wall (similar to a user's wall on Facebook) that all SCSs can view, access, and modify the win conditions concurrently. Winbook maintains social networking look-n-feel by allowing posts that can be win conditions, issues, options, and comments with the user's avatar. Winbook requires a special 'role' called the shaper, who is a team member responsible for shaping the discussions and shortcutting team-wide discussions of term definitions or overlapping artifacts. Winbook offers a color coded view of the notion of WinWin Equilibrium [10] – agreed win conditions or options are shown in green, those marked as potentially agreed as yellow and those not agreed are shown in red. This color coded visualization of the wall helps see the agreement status at a glance and helps the stakeholders know what the development team is comfortable to committing and be accountable on. Thus, the wall serves as color-coded negotiable contract where it's clear what are the expectations (win conditions), potential hurdles (issues), tactics for resolution (options), and what all can the team commit to delivering at their current level of understanding of the project.

Fig. 2. Winbook Tool

3.3. Value-Based Requirements Prioritization (VBRP)

Although Winbook helps capture value propositions of the stakeholders, a critical step is identifying the priority of each value proposition and planning the system and software development lifecycle around the most valuable items first. To support the requirement prioritization, decision analysis framework called TOPSIS (Technique of Ordered Preference by Similarity to Ideal Solution) was selected as a VBRP framework of choice [11]. It was decided to also use the same for learning purposes to make explicit value-oriented decision making in systems and software development.

TOPSIS is a decision analysis framework based on the vector space model of computation where a set of alternatives is scored against a set of criteria. As depicted in Figure 3, each criterion is assigned a relative weight and a direction of preference i.e. if more/less of the particular criterion is preferred. For example, if *Cost* is a criterion and if the alternatives must be ranked by 'low cost' the direction of preference would be '-'. If however, higher cost items must have preference then the direction would be stated as a '+'.

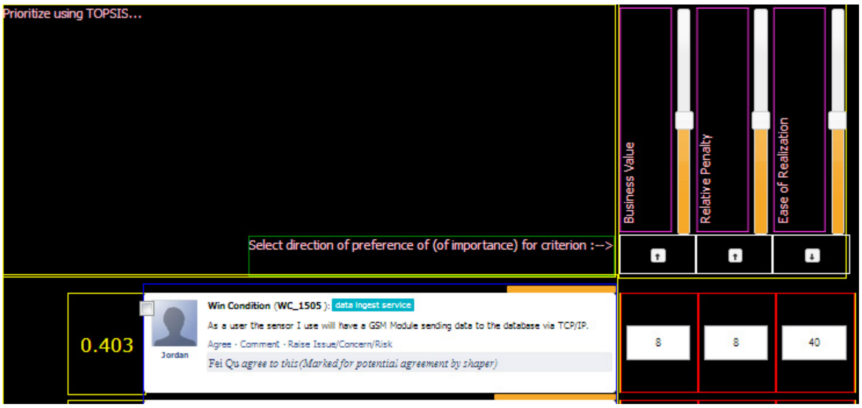


Fig. 3. Winbook tool with TOPSIS framework

Each alternative is assumed to be a vector in *criteria space*. The theoretical ideal alternative is the one with the best score for each criterion. The theoretical non-ideal alternative is the one with the worst score for each criterion. TOPSIS rank orders the alternatives by minimizing their vector distance to the ideal while maximizing it from the non-ideal alternative.

TOPSIS has now been integrated into Winbook itself, so the development teams can prioritize their requirements (i.e. win conditions) using decision analysis model. From the top right corner of Figure 3, they are also able to perform a sensitivity analysis by changing the weights of the various criteria and seeing how the resultant priorities change. They are thus better able to channelize the efforts of their negotiations and planning to work with the most valuable items first. Since Decision theory is one of VBSE's 4+1 theories [10], it is very value adding for the students and systems engineers to learn how to use a decision analysis framework in practice and to provide some rigor support to the decision making activities.

4. Representing the advances by using Rational Method Composer

Many systems engineering processes are complex such as Program Model, Winbook, VBRP mentioned above. It is hard for both process engineers and process users to capture all the process guidelines and make them easier to follow. In an attempt to model the ICSM and its advanced concepts, we have considered various EPG generator tools such as Spearmint [12] and IBM Rational Method Composer [13]. Upon the detailed comparison between various electronic process guide generator tool [14], with more capabilities, easy to use, and a robustness framework, IBM Rational Method Composer was selected to handle the diverse and high degrees of flexibility and tailorability of the ICSM.

Rational Method Composer or RMC represents process elements in terms of roles, tasks, work products, and guidance. A task is composed of steps, while a work product can be categorized into artifacts, deliverables, and outcomes. Additionally, there are many kinds of guidance such as templates, examples, guidelines, concepts, and tools. A process engineer can provide detailed descriptions of each element in the forms of pure texts, hyperlinks, html-embedded texts, or pictorial representations. Each process element notation is represented by comprehensive and easy to understand graphical representations. The relationship between process elements can be easily defined and configured. For example, we can assign tasks to a responsible role(s), artifacts as an input or an output of a task, guidance as supplementary information to a practice, and a template or an example as optional information to an artifact.

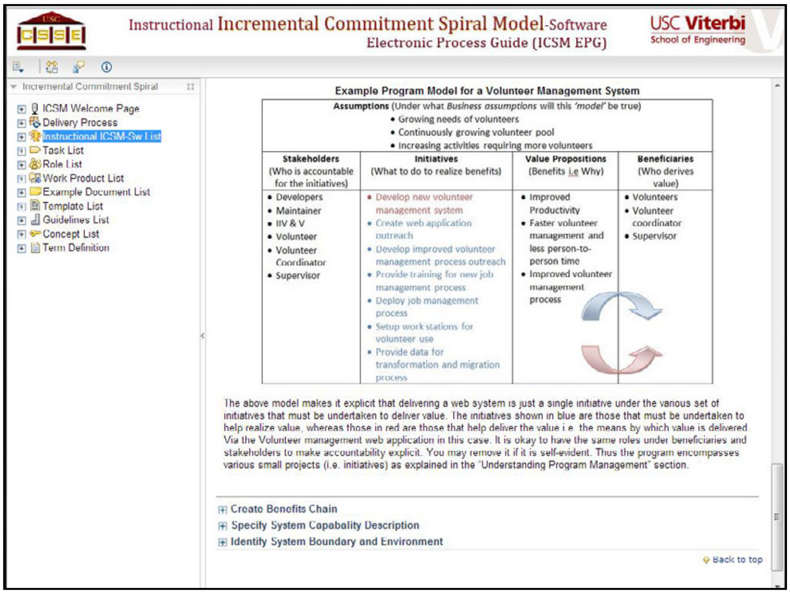


Fig. 4 A screenshot from the ICSM EPG

Figure 4 shows an example of how the ICSM EPG represents a task called "Create a high level Program Model". The ICSM EPG explains general information of the task such as primary performer, secondary performers, output and steps of this task. Program Model, which is a new concept that needs to be explained to the process authors, is added to the ICSM EPG as a step under a task called Create a shared vision. The process user will see textual and graphical representation explaining how to perform this step in detail.

After all the process elements and their relationships have been defined, a process engineer can configure the process framework by defining the time lines, such as phases, iterations, and milestones, and constructing the work breakdown structure, team allocation, and work product usage by assigning activities and the associated tasks in the appropriate time lines. If there is more than one delivery process, since the method content and the process content are separated, the process engineer can pick and choose or reuse by assigning process elements to appropriate processes.

5. Discussion

5.1. Feedback from Systems and Software Engineering Class

5.1.1. The ICSM EPG as a process communicator

By surveying 54 graduate students who are using the ICSM EPG in their Software Engineering course during fall 2012 semester at the University of Southern California on how the ICSM EPG supports them in various activities. On the process communication perspective, Figure 5 shows that 91% of the respondents rate the ICSM EPG as an effective tool. However, one participant or 2 percent of the respondents finds the EPG is too complex for a beginner and prefers a physical print out process guidelines and face-to-face communication, rather than an online tool. In general, they give the ICSM EPG an average score of 8.37 out of 10 (1= not effective, 10= effective) as an effective tool to communicate among their team members about the project process, steps, input, output artifacts, roles, responsibilities, milestones, success criteria and etc.

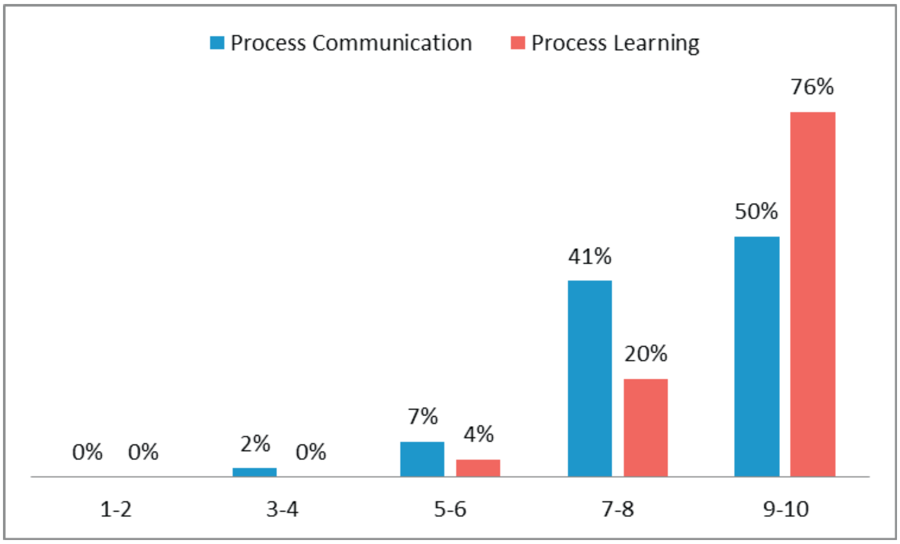


Fig. 5. Effectiveness of how the ICSM EPG supports a system engineer in process communication and learning

Table 1 summarizes the supporting rationale why the system engineers think that the ICSM EPG can support them in process communication. We find that the most important beneficial factor is clearly-defined roles, tasks, artifacts, and milestones which can help systems engineers know what to do, distribute appropriate work allocation, and communicate among their team members. The second most important factor helped in communicating about the project progress. For example, primary and secondary performers of a certain task can refer to the same process element when they want to synchronize and stabilize their works. The third most important factor is an ability to facilitate in understanding roles and responsibilities. The ICSM EPG provides pictorial and tabular information, such as a work breakdown structure, to help understanding roles, responsibilities, and relationships among roles. Understanding relationships among roles can help understand not only their roles, but also other team members' roles.

Table 1. Top-ranked factors supported in using the ICSM EPG for a team communication about the project progress

Rank	*Factors supported in using the ICSM EPG for a team communication*	*Frequency*
1	Clearly-defined roles, tasks, artifacts, and milestones	30
2	A reference for the project progressing	18
3	Facilitating in understanding roles and responsibilities	6
4	Team synchronization	5
5	Sharing same information among team members	4
6	Assigning roles and responsibilities	3

5.1.2. The ICSM EPG as a learning supporter

In addition, they rate the ICSM EPG with average score 9 out of 10 (1= does not help at all, 10=very helpful) in supporting them in learning new concepts in systems engineering. As shown in Figure 5, 96% of the respondents agree that the ICSM EPG tremendously help them coming up with their learning curve. As summarized in Table 2, clearly-defined roles, tasks, and steps in each ICSM phase are the most important factor to understand and learn roles and responsibilities. The cross referenced links among different views, such as work breakdown structure and work flow, make it easy to jump back and forth between different views led to their apprehension and realization. Moreover, the ICSM EPG is based on a tree-like structure. It is easy to navigate. By providing schematic or relationship diagrams of what role entail, students in the study group can visualize the relationship of roles, tasks, and work products. For instance, Figure 6 shows responsible tasks and work products of a role. An Operational Concept Engineer can get the overall responsibilities and learn in detail by clicking provided icons. Having defined boundaries among roles reduces confusion while students or systems engineers are learning roles and responsibilities. Furthermore, the ICSM EPG provides templates of work products and examples to speed up their learning and perform roles and responsibilities effectively.

Table 2. Top-ranked factors supported in using the ICSM EPG in learning about roles and responsibilities

Rank	*Factors supported in using the ICSM EPG in learning about roles and responsibilities*	*Frequency*
1	Clearly-defined roles, tasks, and steps in each ICSM phase	45
2	The cross referenced links among different views	8
3	Tree-like structure	6
3	Providing schematic or relationship diagrams of what role entail	6
5	Defined boundaries among roles	5
5	Providing templates and examples	5

Fig. 6. Responsible tasks and artifacts for an operational concept engineer

Additionally, based on the survey results, system engineers are comfortable in learning new practices by using the EPG. Table 3 identifies features and usages sorted from highest to lowest importance in helping understanding the process and in developing the artifacts.

Table 3. Features and usages in the ICSM

Rank	*Features*	*Usages*	*Frequency*
1	Clear description and boundary of each role	Roles are representatives of the ICSM principles, so studying roles and relations helps understand the ICSM process and usage. Having clearly-defined roles identifies primary and secondary performers who produce the artifacts.	27
2	Providing templates, guidelines, examples, and work products	Templates, examples, and guidelines maintain consistency of the artifact and reduce defects. Work products are used to identify which artifacts should be produced in each phase.	26
2	Cross referencing among different views	Learning in different views allows students to visualize and understand roles and the ICSM process better.	26
2	Role/Task/Guidelines/Work Product/Example Document List	Accessing via lists helps students save time in finding what they would like to know and be required to perform.	26
5	Defining tasks and breaking down into steps with all associated role	Learning tasks and their steps facilitates students in learning roles and responsibilities and the ICSM process in practice.	19
6	Various delivery processes	Delivery process identifies tasks, mile stones, and work flow in each ICSM phase.	17
7	Flow charts and diagrams of the ICSM process	Graphic views present concurrent view of each role and help students have a clear vision of the ICSM process.	9
8	Tree-like structure	Tree-like structure enables team members to navigate the ICSM EPG and process easily.	8
9	Defined terms and its explanation	Defining terms help understand acronyms and understand contents in the ICSM EPG better.	6
10	Suggested the default deliverables and completion criteria in each phase	It identifies what artifacts should be produced and how many responsible performers should do.	5
11	Baselines	A baseline provides overall phases, mile stones, and artifacts.	3
12	Common mistakes	Common mistakes help prevent errors when performing some tasks and producing artifacts.	2

5.2. Process Management and Maintenance

With a continuous improvement in system engineering concepts and methodology, to introduce and familiarize advance practices to system engineers, we need a tool that help process engineers to maintain and update its content easily. RMC fully supports process tailoring and configurations, so the EPG can surely be extended to support process evolution. To add new information, the process engineers simply add process content. The RMC allows them to de-associate the outdated content and create a new association without restructuring the delivery process from scratch. In facilitating process reuse, RMC clearly separates the method contents and process contents, so the process engineer can tailor the method content without affecting the process content, and vice versa. For example, when we added the Program Model and its related concepts, to our existing knowledge repository, it took us as little as 2 person-hours to create necessary process elements and another 1 person-hours to re-build the delivery process.

6. Conclusion and Future Work

In this paper, we introduced new system engineering concepts such as Program Model, WinBook, and Value-Based Requirements Prioritization Framework. The systems engineers may understand the concepts and benefits of these breakthrough theories but without a good process guide, it would be very difficult for them to identify how to apply these concepts and what should they do next.

To help systems engineers understand the ever-changing and continuous improvements in systems engineering, we need a tool that has easy-to-understand representation and easy-to-update structure. IBM Rational Method Composer is a tool that satisfies all these requirements. We have used RMC to develop the ICSM EPG for our graduate-level software engineering class. Students, who act as systems engineers in their real-client project development, found that the ICSM EPG is an effective tool that helps them learn new concepts about their roles, responsibilities, and development processes. On the other hand, as process authors who need to introduce new process areas or practices and maintain the process guide, RMC provides quick and easy way to ensure that the process guide is always up to date.

Furthermore, in order to optimize the capabilities of RMC, the ICSM EPG can be extended to track project progresses or link with development modules by integrating it with other IBM Rational tools such as Rational Portfolio and Rational Software Architect. With an integration capability of RMC, RMC can be used with tools for analysis, development, testing, and management of projects and portfolios. For example, RMC can be deployed as project plan templates for IBM Rational Portfolio Manager, so it allows the manager to plan the project and allocate resources effectively.

References

1. M.I. Kellner, et. al, Process Guides: Effective Guidance for Process Participants. Proc. of the 5th Int. Conf. on the Software Process: Computer Supported Organizational Work, Lisle, Illinois, (1998) 14-17.
2. IBM Rational Method Composer, http://www-01.ibm.com/software/awdtools/rmc/.
3. P. Haumer, IBM Rational Method Composer: Part 1: Key concepts, December 2005, http://www.ibm.com/developerworks/rational/library/dec05/haumer/.
4. S. Koolmanojwong and B. Boehm, The Incremental Commitment Model Process Patterns for Rapid-Fielding Projects, ICSP 2010, Paderborn, Germany.
5. CSCI577ab Software Engineering Class, http://greenbay.usc.edu/csci577/fall2012/.
6. A. Osterwalder and Y. Pigneur, Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers, Wiley, 2010.
7. J. Thorp, The information paradox : realizing the business benefits of information technology, McGraw-Hill Ryerson, 2003.
8. N. Kukreja and B. Boehm, Process Implications of Social Networking-Based Requirements Negotiation Tools, In International Conference on System and Software Process (ICSSP) 2012, In Press.
9. D. Yang, et. al, WikiWinWin: A Wiki Based System for Collaborative Requirements Negotiation, Proc. of the 41st Annual Hawaii International Conference on System Sciences, January 2008.
10. B. Boehm and A. Jain, The Value-Based Theory of Systems Engineering: Identifying and Explaining Dependencies, INCOSE 2007, San Diego.
11. N. Kukreja, et. al, Selecting an Appropriate Framework for Value-Based Requirements Prioritization: A Case Study, In 20th IEEE International Requirements Engineering Conference, illinois, Chicago, USA.
12. U. Becker and M. Verlage, The V-Model Guide: Experience with a Web-based Approach for Process Support, Proceedings of Software Technology and Engineering Practice, 1999.
13. Introducing IBM Rational Method Composer, http://www.ibm.com/developerworks/rational/library/nov05/kroll/.
14. M. Phongpaibul, S. Koolmanojwong, A. Lam, and B. Boehm, "Comparative Experiences with Electronic Process Guide Generator Tools," ICSP 2007, Mineapolis, USA