CrownVision: Metrics Visualization for Project Management

Sergiu Dascalu
Department of Computer Science and Engineering
University of Nevada, Reno, NV
dascalus@cse.unr.edu

Norm Brown
National Institute for Systems Test and Productivity
Tampa, FL
norm.brown@verizon.net

Sohei Okamoto
Department of Computer Science and Engineering
University of Nevada, Reno, NV
okamoto@cse.unr.edu

Sermsak Buntha
Department of Computer Science and Engineering
University of Nevada, Reno, NV
buntha@cse.unr.edu

Namit Chawla
Department of Computer Science and Engineering
University of Nevada, Reno, NV
chawla@cse.unr.edu

ABSTRACT

Software project management tools have become essential for effectively managing software development efforts. Project managers need straightforward yet effective progress and status reports based on metrics for themselves and stakeholders – reports intuitively produced without resorting to a 200-page manual. This paper introduces CrownVision, a new type of project management software system which facilitates project development insight and supports more responsive and effective project development control. This web-enabled innovative system generates visual representations and utilizes evaluative and predictive techniques to accomplish these objectives. The UML-based software model of CrownVision is presented in the paper, together with several examples of use. Planned enhancements and directions of further work are also described.

KEYWORDS

Software engineering, project assessment, project management, metrics visualization, software metrics, evaluative techniques, predictive techniques.

1. INTRODUCTION

New project management tools are frequently created to improve project management through assessment and metrics [1, 2, 3]. Occasionally, a principal objective of such a tool is to not only enable more effective use of metrics in the management process, but to do this in such a way that leads to accurate project assessment [4, 5]. There are numerous studies of how to handle programmers in a long-term and sophisticated project [6, 7, 8]. Too often these software tools try to tackle too many problems in one large program and some features inevitably fail to reach their full potential [4, 5, 8, 9].

CrownVision is a straightforward web-based (thick client) visualization tool tuned to the needs of software and system development managers; it targets a manager's need to effectively gain critical insight into the status and progress of his or her project's development. *CrownVision* is built upon a straightforward mechanism for creating tailored reports based upon metrics; the metrics are applied via readily changeable templates.

The initial version of *CrownVision*, now in alpha-test, is described in this paper. Started as a collaboration between two universities, this project is intended to serve as a concept demonstration of a new class of tools for managing acquisition and development of Department of Defense large-scale software-intensive systems; to serve as an interactive mechanism for refining and elaborating information and displaying requirements directly from senior defense acquisition managers; and, to serve as a baseline from which further development will evolve. Discussions with present and former defense officials are planned for this purpose. Civilian federal government agencies also acquire large-scale software-intensive systems which have virtually identical management needs for which *CrownVision* will be directly applicable.

The remainder of this paper is organized as follows: Section 2 addresses the difficulty in project management as well as the nature of limitations inherent with previous solutions. Section 3 presents *CrownVision*'s main functional and non-functional requirements. Section 4 describes several key use cases of the tool. System design is detailed in Section 5, while Section 6 presents snapshots of *CrownVision*'s user interface. Results of use are outlined in Section 7, followed in Section 8 by a discussion of anticipated directions for subsequent elaboration and further development efforts. Finally, our conclusions are presented in Section 9.

2. BACKGROUND

Due to ever increasing size and complexity of contemporary software projects, the ability to effectively manage these programs is becoming increasingly difficult [10, 11]. As these projects grow to very large-scale, their complexity grows with them. In an attempt to deal with this complexity, such projects are typically decomposed into a number of software subsystems, items, or components built by different software development teams constructed by a large number of programmers - one such

Navy Department program required a software team of over 300 programmers [12]. Similarly, as large software systems themselves are combined into exponentially larger systems-of-systems development projects, complexity again jumps, and with it the management challenge of gaining effective and meaningful insight and control of cost, schedule, functional progress, integration progress, staffing, and risk, which of necessity must be metrics based.

While software development metrics have been widely researched in terms of their utility to managing software developments, this work has typically been applicable to relatively small-scale projects. To shed more light on the problems endemic to management of large-scale and very-large-scale software development projects, and their potential solutions, we have established the *CrownManagement* research initiative [13]. This project seeks to uncover essential metrics relationships and develop related tools needed to bring meaningful insight to the software acquisition and development manager. *CrownVision*, the metrics presentation component of *CrownManagement*, is aimed at effectively delivering this insight to managers and stakeholders.

This paper addresses *CrownVision*, the project metrics collection and visualization component of the *CrownManagement* process for managing acquisition and development of large-scale software-intensive projects.

Since the initial version of *CrownVision* presented in this paper is intended as a proof-of-concept baseline demonstration tool to be further evolved, it does not possess the complete spectrum of capabilities planned for future elaboration. The most relevant planned capabilities for *CrownVision* are outlined in the Future Work section of this paper.

3. REQUIREMENTS SPECIFICATION

The development of *CrownVision* was performed following a systematic software process [6], which included the creation of several software models for the system, most of them described using the UML (Unified Modeling Language) notation [14, 15]. Having the process followed and the models (artifacts) created at all stages of software development – from requirements specification, through high-level and low-level design, to implementation, integration and testing – has instilled a discipline of development which proved to be beneficial for increasing the quality of the software prototype created.

The excerpts from *CrownVision*'s software model presented next serve primarily to describe the tool's main capabilities. They also provide an illustration of practical software construction that could be useful for guiding other software development projects, particularly web-based software applications.

Functional and non-functional requirements are defined in this section; however, not all of requirements are presented due to space limitations. Subsection 3.1, Functional Requirements, describes in detail the CrownVision system's behaviors while subsection 3.2., Non-Functional Requirements, gives examples of constraints placed on the system. The style used for representing requirements is the simple and efficient one proposed in [16]. In particular, having the requirements "atomic" is useful for easy tracing, verification and validation [4, 5, 16].

3.1 Functional Requirements

Functional requirements are roughly divided into three sections, corresponding to the system's main components, namely *template*, *project*, and *report* management.

In short, *templates* are defined (and then populated with data) to capture some specific aspect of an ongoing project, for instance lines of code (LOC) created per week by each programmer in a team, the number of bugs find by each tester in each set of test cases, the number of hours worked per month by each team member, the overall accumulated cost of the project, etc. (Note that a template populated with data is also referred to as a *table* – in other words, templates are project-indepedent while tables are project-dependent).

A *project* is created in *CrownVision* anytime metrics collection and analysis are needed in relation with an actual software project (that is, a *CrownVision* project corresponds to a real-world development project).

Finally, a *report* is generated in relation to a project, based on a specific template and a specific *visualization form* (or *view type*). Examples of visualization forms available in *CrownVision* include line graphs, bar graphs, pie charts, and "spider" charts.

R01 The system shall authenticate a user by user name and password with user account information in the system.

Requirements for the Template Component

R02 The system shall allow a user to create templates.

R03 The system shall allow a user to specify the size of the template matrix.

R04 The system shall allow a user to specify the data type of each of the columns in the template matrix.

R05 The system shall allow a user to add a column to the template matrix.

R06 The system shall allow a user to remove a column from the template matrix.

R07 The system shall allow a user to modify a template.

R08 The system shall allow a user to delete a template.

Requirements for the Project Component

R09 The system shall allow a user to create projects.

R10 The system shall allow a user to add templates to a project.

R11 The system shall allow a user to remove templates from a project.

R12 The system shall allow a user to insert data into a template.

R13 The system shall allow a user to modify existing data of a template.

R14 The system shall allow a user to delete data from a template.

R15 The system shall allow a user to modify a project.

R16 The system shall allow a user to delete a project.

Requirements for Report Component

R17 The system shall allow a user to create reports.

R18 The system shall allow a user to select a project, a template, and view mode for generating and displaying a report.

R19 The system shall allow a user to save reports.

R20 The system shall allow a user to delete reports.

3.2 Non-Functional Requirements

N01 The system shall be developed as an web application.

N02 The system shall use the Apache web server.

N03 The system shall use PHP with GD extension for implementation.

N04 The system shall use MySQL database server.

N05 The system shall be intuitive and user-friendly.

4. USE CASES

4.1 Use Case Diagrams

Since the *CrownVision* tool is focused on user interactions, identifying and analyzing use cases has been a very important part of the modeling process. In essence, the program performs actions in response to user requests. The main user requests, modeled as use cases, are illustrated in the use case diagrams shown in Figures 1, 2 and 3. A detailed description of each use case is then presented in Subsection 4.2.

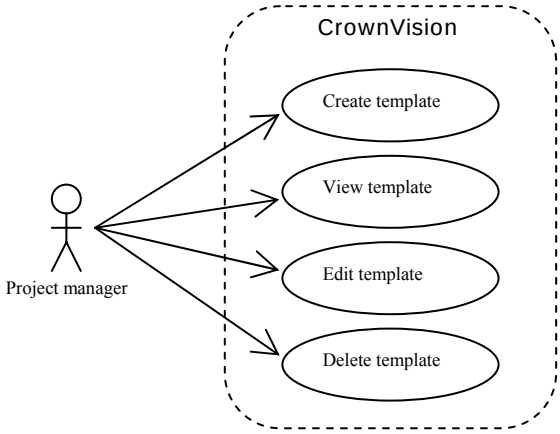

Figure 1. Use case diagram of the template component

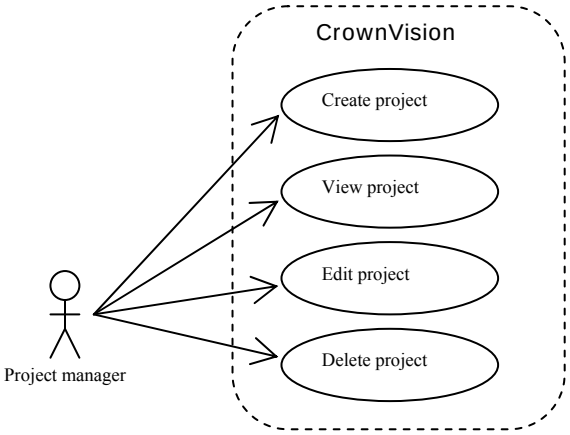

Figure 2. Use case diagram of the project component

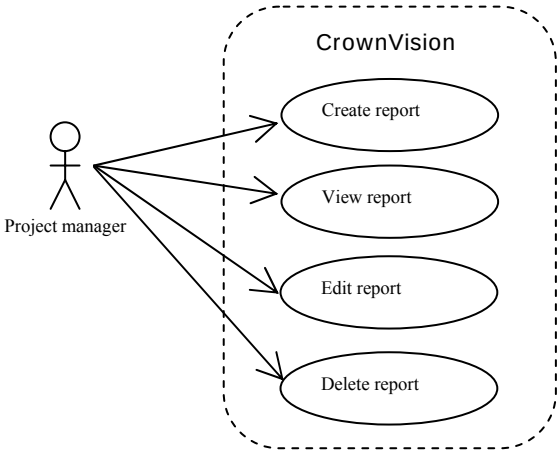

Figure 3. Use case diagram of the report component

4.1 Descriptions of Use Cases

Create a template

This is the interaction between the system and the user when the user requests to create a new template.

View template

This is interaction between the system and the user when the user requests to browse existing template. In addition, as part of this interaction, the user can look at the details of the template (e.g., columns, data type of each column).

Edit template

This is the interaction between the system and the user when the user requests to modify a selected template. This interaction is not permitted if the selected template is already being used by one or more projects.

Delete template

This is interaction between the system and the user when the user requests to delete a selected template from the system's database. Again, this interaction is not possible if the selected template is already being used by one or more projects.

Create project

This is the interaction between the system and the user when the user requests to create a new project.

View project

This is the interaction between the system and the user when the user requests to browse an existing project. In addition, the user can look at various details of the projects with this interaction (e.g., associated templates, reports generated so far, visualization forms available).

Edit project

This is the interaction between the system and the user when the user requests to modify a selected project.

Delete project

This is the interaction between the system and the user when the user requests to delete a selected project from the system's database. To allow this, administrator rights are required and confirmation is requested.

Create report

This is the interaction between the system and the user when the user requests to create a new report.

View report

This is the interaction between the system and the user when the user requests to see an existing report.

Edit report

This is the interaction between the system and the user when the user requests to modify a selected report.

Delete report

This is interaction between the system and the user when the user requests to delete an existing report from the system's database. To allow this, administrator rights are required and confirmation is requested.

5. DESIGN OVERVIEW

This section provides an overall perspective on how the system was created and implemented. *CrownVision* was designed as a web-based application that generates dynamic contents. PHP [17, 18] and MySQL [19] are the major tools used for creating the system's prototype. From a design point of view, *CrownVision* can also be divided into three main components: Project, Template, and Report.

Figure 4 shows the site map of *CrownVision*. The home page has links to each component page, and dynamic contents are generated and inserted into the main section of the web page. For each component, the links to edit operations are provided as browse operations in order to simplify the interface and optimize the user's actions.

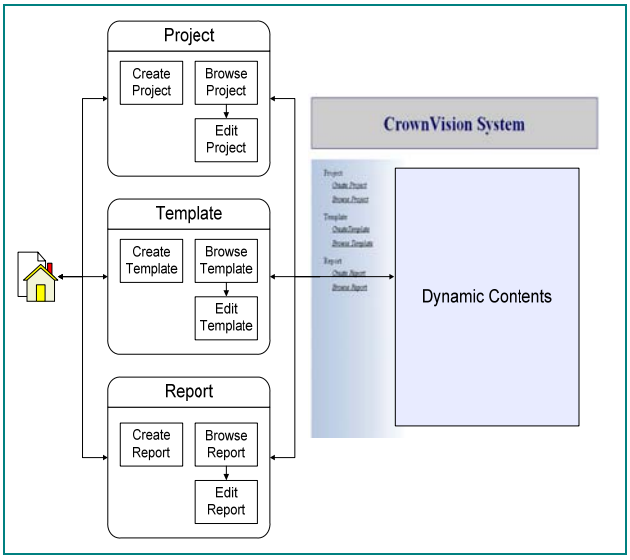

Figure 4. CrownVision site map

Figure 5 shows the organization of functions defined in the *CrownVision* software application. The high level architecture of *CrownVision,* consisting of four major components, is presented in this figure. In short, the *Utility* component contains general utility functions such as Authenticate and ShowNavigation, while all other functions belong to their related components. These functions generally use the PHP function package for implementation. To manipulate the database, the PHP package also includes functions to send commands to the MySQL server.

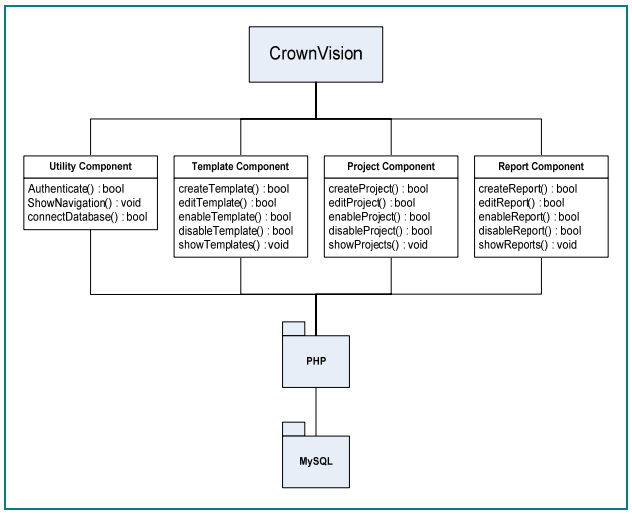

Figure 5. CrownVision high-level function organization

6. USER-INTERFACE DETAILS

In this section, several actual screenshots are taken from *CrownVision's* alpha prototype. They provide details on how the user interacts with the system for project management data collection and visualization purposes.

After the user provides his or her username and password, the *CrownVision* system displays its main interface browser, which contains on its left-hand side a simple yet effective navigation bar, shown in Figure 6.

Figure 6. Navigation bar in CrownVision's main interface

Assuming several templates are already available, one of the first operations a user can perform is to create a new project (due to space limitations, the creation and management of templates are not shown in the paper). As a simple example, Figure 7 shows how a user can create a new project ("test project") with just two tables ("cost table" and "LOC table"), based respectively on two existing templates ("cost template" and "LOC template"). As a matter of difference in concepts and terminology, while templates can be shared across projects, tables belong strictly to individual projects.

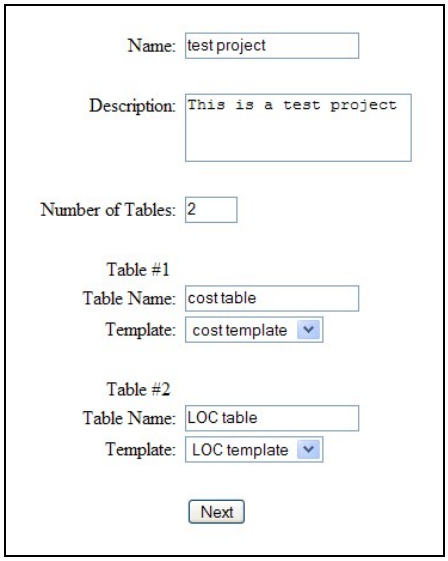

Figure 7. Creating a project

Considering a different project next, the screenshot in Figure 8 shows the tool's interface after the user has clicked the "Edit Metric" in the main navigation bar (Figure 6). Through the interface shown in Figure 8, the user can input data into a project table. As long as the project is active (that is, not archived) and the table exists, the user can add, modify or delete rows to the table at any point in time. The "update" button at the bottom of the screen is provided to record the data into the database.

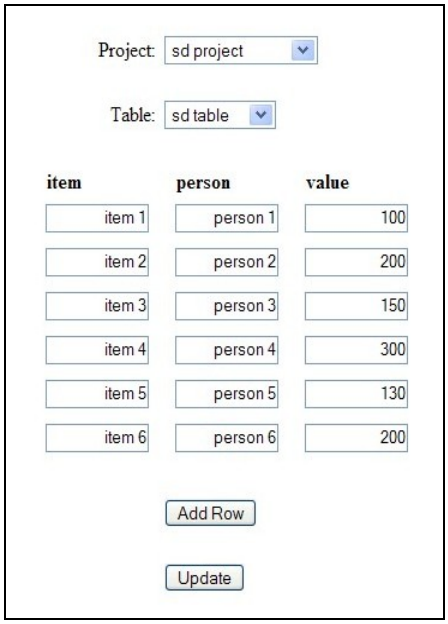

Figure 8. Editing a metrics table

The screenshots presented next give illustrate some of the visualization capabilities of the *CrownVision* software system.

For example, the interface presented in Figure 9 is displayed when a user clicks "Browse report". As a result, the system shows the existing reports. The user may modify or delete a report by clicking "edit" or "delete", respectively. The "view" button can be used to look at the detailed content of a report. Note that the visualization type of the report is indicated in the "type" rubric of the interface (e.g, line graph, or graph chart).

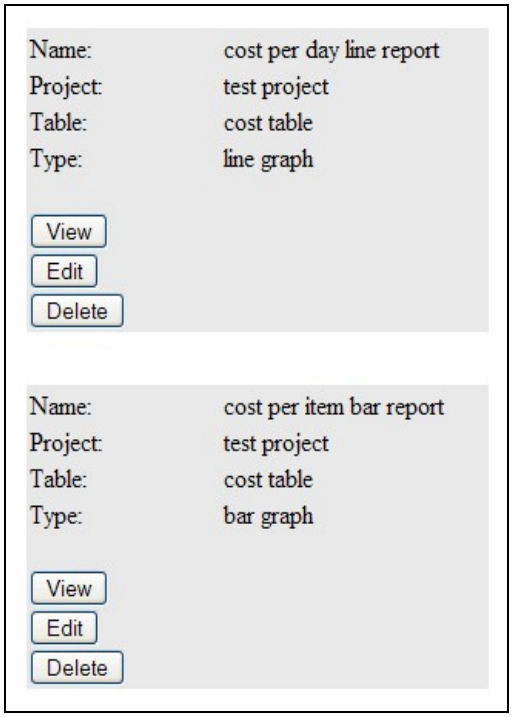

Figure 9. Browsing reports

Figure 10 gives an example of the detailed content of a report. The details of the report are presented in the top part of the screen, while specific report data is displayed at the bottom of the screen using a graphical representation. Types of available visualizations in CrownVision include line graphs (illustrated in Figure 10), pie charts (shown in Figure 11), bar graphs (an example is given in Figure 12), and "spider charts" (depicted in Figure 13). All these visualization forms are powerful, quick and comprehensive means of conveying information to project managers. In particular, major changes, trends, gradients, extremes, weights and proportions could be identified practically at a glance by the eyes of a trained manager.

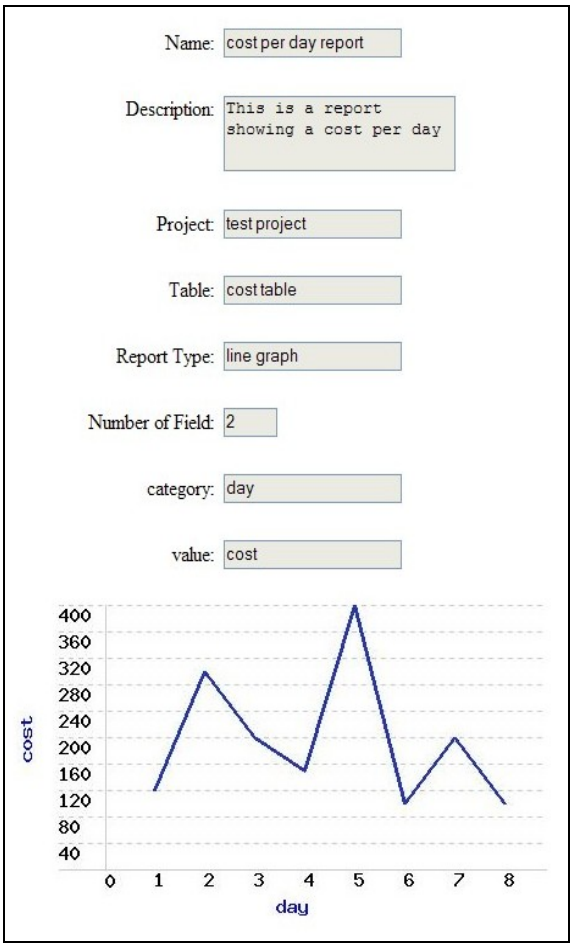

Figure 10. Viewing a report: line graph visualization

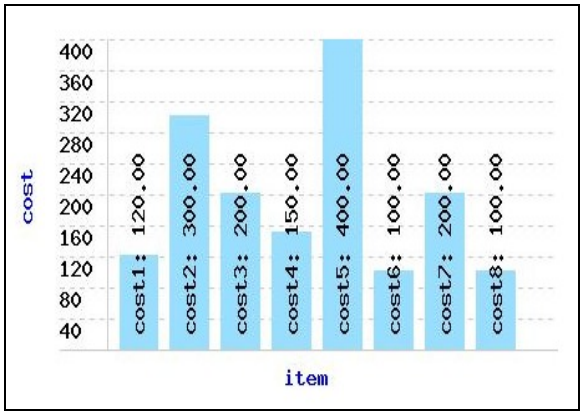

Figure 11. Bar graph visualization

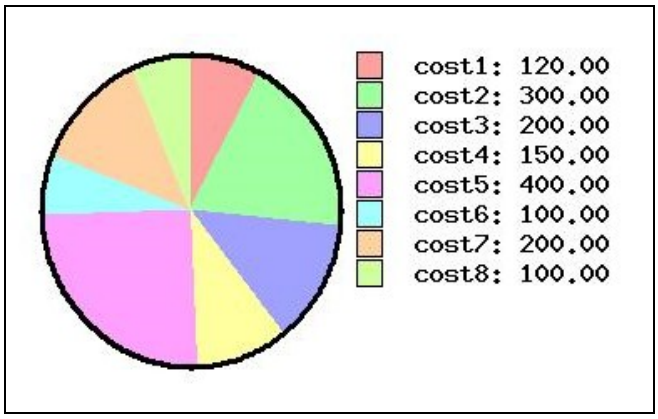

Figure 12. Pie-chart visualization

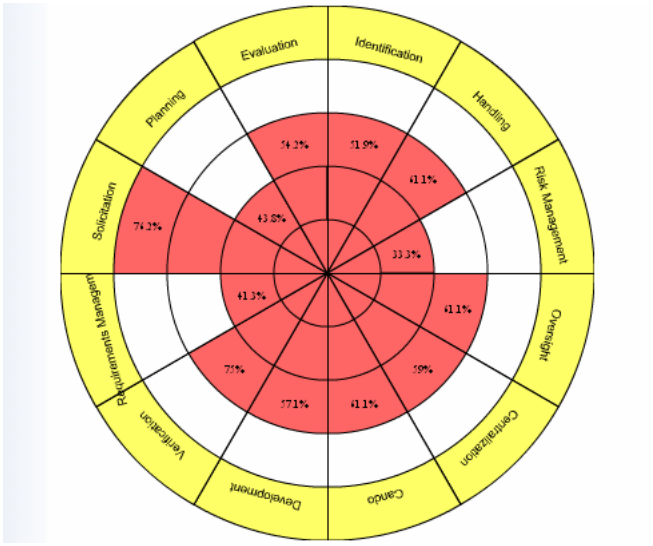

Figure 13. Spider chart visualization

7. RESULTS

Our experimentation with the system has demonstrated that *CrownVision* works well in any web browser, and is platform independent. It enables users to conveniently create, modify, or delete metrics display templates and project configuration profiles. It also allows the users to create reports easily and quickly. Additionally, the visualization configuration of a report is saved into an image file (jpeg) for export to other presentation tools, such as Microsoft PowerPoint. This is a simple, yet powerful concept as the upper-level management and influential decision-makers are particularly interested in visualizations that give a quick, clear and comprehensive view of the project status.

8. FUTURE WORK

At this point in time, *CrownVision* is in its initial version of a spiral development schema. It will undergo additional evolutionary development to further evolve requirements and implement corresponding behavioral modifications. Extensions currently being considered include:

- Mathematical functions — such as summation, algebraic operations, etc. — which could be helpful for project managers to analyze series of data in metrics.
- Wireless devices: currently, *CrownVision* is designed for desktop use. To accomplish our goal, the system should be accessible from anywhere and at anytime. Accordingly, *CrownVision* should enable users to access the system on wireless handheld devices, e.g. personal digital assistants (PDAs).
- Intelligent functions: *CrownVision* will become more useful as it adds intelligent functions for analysis of metric data, for example to intelligently evaluate development and acquisition metrics in terms of thresholds, trends, and other criteria; to provide estimation of progress, cost, and risk; and to suggest potential management actions. Techniques including artificial neural networks and expert systems will be investigated to provide such intelligent functions.
- Lastly, *CrownVision* will implement evaluative and predictive understandings gained from this research.

9. CONCLUSION

CrownVision is a web-based tool that provides project management insight and enables a project manager to concurrently monitor and analyze simultaneous projects. It allows the manager to visualize project metrics in a variety of formats – pie charts, bar graphs, line graphs, etc. Because it is web-enabled, it is accessible from anywhere and at anytime by the project manager or the stakeholders. Enhancements for *CrownVision* are planned which will provide additional functionality such as mathematical computation, handheld accessibility, and estimation and predictive capabilities to aid in early bottom-line understanding of program status, and to facilitate effective project control.

ACKNOWLEDGEMENTS

The *CrownManagement* project and its metrics collection and presentation component *CrownVision* are a collaborative effort between the Computer Science Departments at the University of Nevada, Reno and the University of South Florida, under the auspices of the National Institute for Systems Test and Productivity. Funding for this project was provided by the Naval Space and Warfare Systems Command.

References

[1] Liberatore, M.J. and Pollack-Johnson, B., "Factors Influencing the Usage and Selection of Project Management Software," *IEEE Transactions on Engineering Management*, vol. 50, no. 2, May 2003, pp.164-174.

[2] Right Track Associates, Inc., *Managing Problems as Projects: Problem Management in IT.* www.ittoolkit.com/articles/tech/mngprobasproj.htm Accessed November 20, 2005.

[3] Romano, N.C., Jr., Fang, C., and Nunamaker, J.F., Jr., "Collaborative Project Management Software," in *Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS-2002)*, January 2002, pp. 233-242.

[4] Pressman, R., *Software Engineering: A Practictioner's Approach,* 6th Edition, McGraw-Hill, 2004.

[5] Endres A. and Rombach, D. *A Handbook of Software and Systems Engineering: Empirical Observations, Laws, and Theories*. Pearson Addison-Wesley, 2003.

[6] Sommerville, I. *Software Engineering,* Addison-Wesley, 7th Ed., 2004.

[7] Goodman, P. *Software Metrics: Best Practices for Successful IT Management*. Rothstein Associates Inc., 2004.

[8] McGovern, F., "Managing Software Projects with Business-based Requirements," *IT Professional*, vol. 4, no. 5, Sept.-Oct. 2002, pp. 18 – 23.

[9] Putnam L.H. and Myers, W. *Five Core Metrics: The Intelligence Behind Successful Software Management* Dorset House Publishing, 2003.

[10] Moyinihan, T., "Howe Experienced Project Managers Assess Risk," *IEEE Software*, vol. 13, no. 4, May-June 1997, pp. 35-41.

[11] Nidiffer, K.E. and Dolan, D., "Evolving Distributed Project Management," *IEEE Software*, vol. 22, no. 5, Sept.-Oct. 2005, pp. 63-72.

[12] Norm Brown, private e-mail communication, Spring 2005.

[13] Dascalu, S.M., Brown, N., Eiler, D., Leong, H., Penrod, N., Westphal, B., and Varol, L., "Software Modeling of S-Metrics: Synergetic Interactive Metrics Acquisition and Visualization Tool," in *Procs. of the 2005 Intl. Conference on Software Engineering Research and Practice (SERP-2005)*, Las Vegas, NV, vol. II, pp. 870-876.

[14] OMG UML Resource Page, www.omg.org/uml Accessed December 1, 2005.

[15] Booch, G., Rumbaugh, J. and Jacobson, I., *The Unified Modeling Language: User Guide*, Addison-Wesley, 1998.

[16] Arlow, J. and Neustadt, I. *UML and the Unified Process: Practical Object-Oriented Analysis and Design*. Addison-Wesley, 2001.

[17] Atkinson, L. and Suraski, Z. *Core PHP programming*. Prentice Hall, 2004.

[18] PHP website *PHP: Hypertext Preprocessor* http://www.php.net/, accessed December 1, 2005.

[19] MySQL website *MySQL: The World's Most Popular Open Source Database* http://www.mysql.com/, accessed October 31, 2005.