NASA-CR-203074

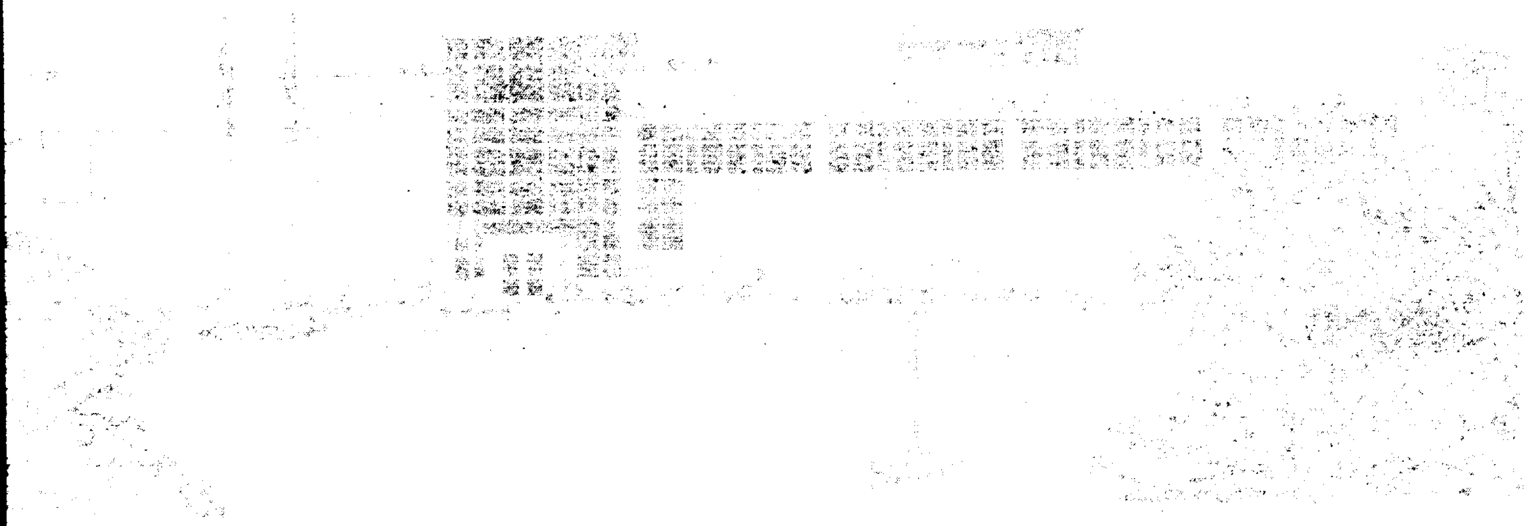
NASA/WVU Software IV & V Facility
Software Research Laboratory
Technical Report Series

NASA-IVV-96-006
WVU-SRL-96-006
WVU-SCS-TR-96-15
CERC-TR-TM-96-007

IN-61-CR
008162

Software Project Management and Measurement on the World-Wide Web (WWW)

by John Callahan and Sudhakar Ramakrishnan



National Aeronautics and Space Administration

West Virginia University

According to the terms of Cooperative Agreement #NCCW-0040, the following approval is granted for distribution of this technical report outside the NASA/WVU Software Research Laboratory

George J. Sabolish [illegible]	[illegible]
George J. Sabolish Date	John R. Callahan Date
Manager, Software Engineering	WVU Principal Investigator

Software Project Management and Measurement on the World-Wide-Web (WWW) *

John Callahan
callahan@cs.wvu.edu

Sudhaka Ramakrishnan
sudha@informix.com

NASA/West Virginia University Software IV&V Facility
100 University Drive
Fairmont, WV 26554
304-367-8235, 304-367-8211(fax)

January 23, 1996

Abstract

We briefly describe a system for forms-based, workflow management that helps members of a software development team overcome geographical barriers to collaboration. Our system, called the Web Integrated Software Environment (WISE), is implemented as a World-Wide-Web service that allows for management and measurement of software development projects based on dynamic analysis of change activity in the workflow. WISE tracks issues in a software development process, provides informal communication between the users with different roles, supports to-do lists, and helps in software process improvement. WISE minimizes the time devoted to metrics collection and analysis by providing implicit delivery of messages between users based on the *content* of project documents. The use of a database in WISE is hidden from the users who view WISE as maintaining a personal "to-do list" of tasks related to the many projects on which they may play different roles.

Keywords: Workflow management, verification and validation, software engineering, issue tracking, software measurement, software metrics, software process

1 Introduction

Collection of metrics and adherence to a disciplined development process are difficult tasks in any software project. Yet, project developers and managers need to understand their processes in order to coordinate assigned tasks effectively. To understand their own work processes and assess the status of an ongoing project, they must be able to measure various aspects of the project tasks, people, and products. An assessment of a project's status is critical to the reassignment of resources, adjustment of schedules, and measurement of product quality.

Change activity is a powerful indicator of a project's status. Automated systems that can handle change requests, track issues, and process electronic forms provide an excellent platform for tracking the status of the project. We have developed a World-Wide-Web-based approach

*This work is supported by NASA Grant NAG 5-2129 and NASA Cooperative Research Agreement NCCW-0040. More information pertaining to WISE can be found at `http://research.ivv.nasa.gov/projects/WISE/wise.html` called the Web Integrated Software Environment (WISE) that supports measurement of change activity as an *implicit* part of the software process. WISE provides a forms-based, workflow management system that helps members of a software development team overcome geographical barriers to collaboration. WISE allows for the improvement of the software process in a realistic environment based on dynamic analysis of changes to information and communication in the workflow. WISE tracks issues in software development process, provides informal communication between the users with different roles, supports to-do lists, and helps in software process improvement. WISE minimizes the time devoted to metrics collection, analysis, and reporting tasks not related directly to project activities. Automated tools like WISE focus on understanding and management of a software process rather than the bureaucracy of an organization.

Since the summer of 1995, WISE has been used on several projects within the National Aeronautics and Space Administration (NASA) and the private sector to coordinate and monitor software verification and validation (V&V) activities. This paper briefly discusses issues related to the use of WISE in the automation of software project management and measurement. From our practical experiences with WISE. we believe that such automated tools can transform chaotic software development projects into more controlled and manageable processes.

2 Overview

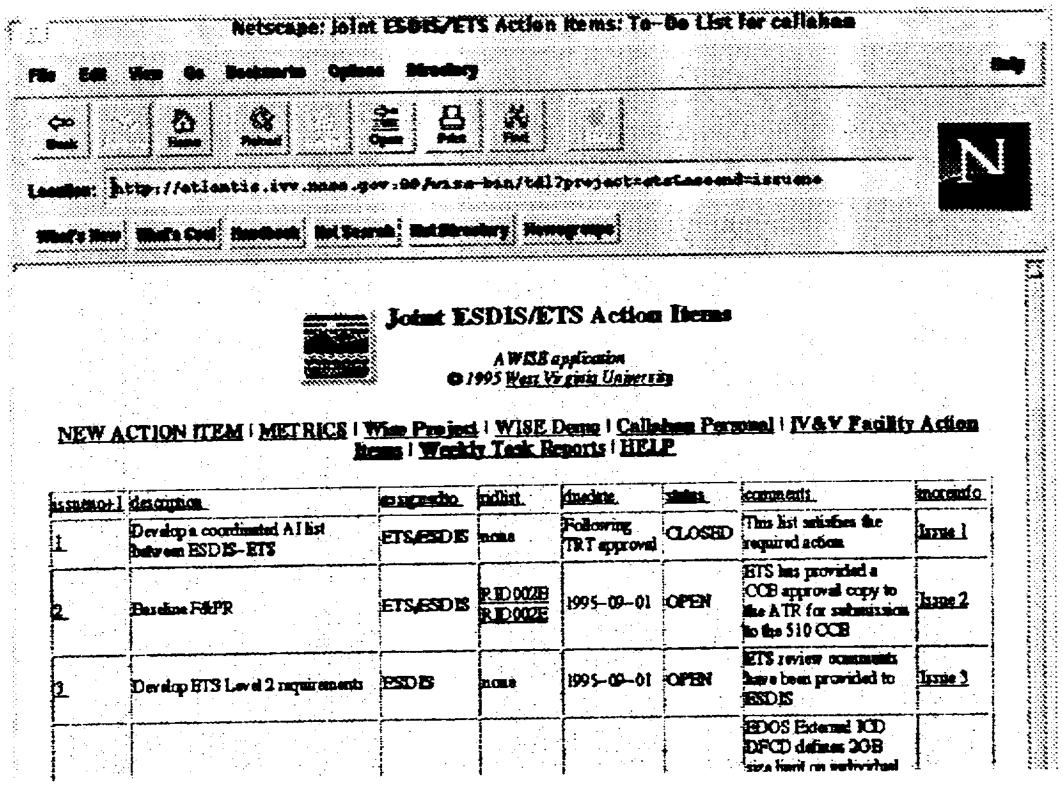
WISE can be installed and made available at a specific URL (Universe Resource Locator) on the Internet or within a corporate network. Using an appropriate browser[1], each user connects into the WISE home page to view their personal "to-do lists" (TDLs) available on several projects. Each person in an organization may be assigned to one or more projects and therefore may have one or more TDLs under WISE. Figure 1 shows a portion of a person's "to-do list" for a sample project.

Each item on a person's TDL is called an *issue*. Each issue contains a set of fields and associated values. These fields are a superset of those shown for each issue on the TDL. Individual issues may be viewed by selecting them as hyperlinks on a TDL. For example. Figure 2 shows an individual issue (i.e., issue number 2) from the TDL shown in Figure 1. WISE TDLs and issue forms can be customized for specific organizational needs. For example, a WISE application can be configured to maintain a list of tasks, an order-entry database, problem reports, or phone contact lists. Legal types of fields in an issue can include:

- free-form text
- finite selections (single and multiple)
- hypertext (i.e., arbitrary HTML[2])
- numbers (integer, real)
- dates and times
- radio buttons
- check boxes

[1]WISE requires support for HTML 2.0 tables. The Beta and production releases of WISE also require support for Java. Many WWW browsers support tables including the most recent versions of NetScape^{tm} and Mosaic.

[2]The use of arbitrary HTML allows WISE to be easily integrated with other WWW-based tools, documents, and resources on a software project. For example, Figure 1 shows issue connections to Review Item Discrepancy (RID) documents on another WWW server. We have also used this facility to connect WISE almost effortlessly to the NCSA prototype HyperNews tool.


Netscape: Joint ESDIS/ETS Action Items: To-Do List for callahan
File Edit View Go Bookmarks Options Directory
Help
Back
Home
Reload
Open
Print
Find
Location:
What's New
What's Cool
Handbook
Net Search
Net Directory
Newsgroups

Joint ESDIS/ETS Action Items

A WISE application
© 1995 West Virginia University

NEW ACTION ITEM | METRICS | Wise Project | WISE Demo | Callahan Personal | IV&V Facility Action Items | Weekly Task Reports | HELP

issuenoⅠ	description	assignedto	ridlist	duedate	status	comments	moreinfo
1	Develop a coordinated AI list between ESDIS-ETS	ETS/ESDIS	none	Following TRT approval	CLOSED	This list satisfies the required action	Issue 1
2	Baseline F&PR	ETS/ESDIS	RID002B RID002E	1995-09-01	OPEN	ETS has provided a CCB approval copy to the ATR for submission to the 510 CCB	Issue 2
3	Develop ETS Level 2 requirements	ESDIS	none	1995-09-01	OPEN	ETS review comments have been provided to ESDIS	Issue 3
						EDOS External ICD DFCD defines 2GB size limit on individual	

Figure 1: A user's "to-do list" for a sample project

An issue appears on an individual's TDL based on their role(s) on projects. Roles are created and configured by project administrators to control information access. A user may play more than one role on a project, but the visibility of fields in an issue and the types of changes allowed on an issue form are dependent on the user's role(s). Roles act as filters on all project issues. This approach provides users with a unique view of their tasks on a project.

Changes to an issue can be submitted by a user directly through the WWW browser because issues are implemented using the CGI Forms Interface. Once submitted, the issue changes are recorded in a log and the issue may disappear from the user's TDL and appear on the TDLs of other users. The specific workflow of issues within an organization is dependent on the customized definitions of issues, fields, and roles on a project in the WISE site configuration file. WISE also provides for delivery and submission of forms via electronic mail to accommodate user's between Internet firewalls, but such users must read the mail via an HTML capable browser. Thanks to the ubiquitousness of WWW browsers, users can access WISE through the World-Wide-Web from a variety of hardware platforms and operating systems.

WISE also provides on-demand access to project metrics. WISE keeps track of changes to issues and other project events. WISE collects metrics based on these events and presents graphical views of the project statistics. In one project, for example, issues can track problems discovered during development. Issues can cycle from an *open* status to a *solved* or *closed* status. Issues marked as solved are those problems that have been diagnosed but not fixed. Once a problem is fixed, the issue can be marked *closed.* Figure 3 depicts metrics for a user's project plotted over time. It shows that the number of open issues initially exceeds the number of closed issues and that very few issues are recorded as "solved" before being closed. Furthermore, this organization noted that the trend in open vs. closed issues helped to estimate the time of their first software release as the projected time when the number of open issues fell below the number of closed issues.

WISE is non-intrusive because it provides a "to-do" list of each issues to each developer in the team. Each issue in the "to-do" list can be acted upon by changes to the values of fields in an issue. The types of changes allowed are dependent on a user's assigned role(s). The composition of the forms and views defines the totality of the software process. Thus, the process is not fixed or globally defined by the manager, but it is highly dynamic and may change based on the different roles of development personnel throughout the lifecycle of a development effort.

3 Process Improvement

In order to achieve the goals of any software project, one must be able to assess the quality of a development process itself. We must be able to improve the software process continually and determine if it is progressing at an acceptable rate. To produce quality products and improve the capability of the organization to produce better products, the software process must improve as well.

Software maturity frameworks are usually characterized by different levels in which an organization can be categorized depending upon the results of the assessment of the organization's software process [7]. Many case studies have been conducted and have shown that there is a need to turn the corner from chaotic, unpredictable cost to a more manageable and controlled software process whereby schedule slippage and cost overruns are avoided.

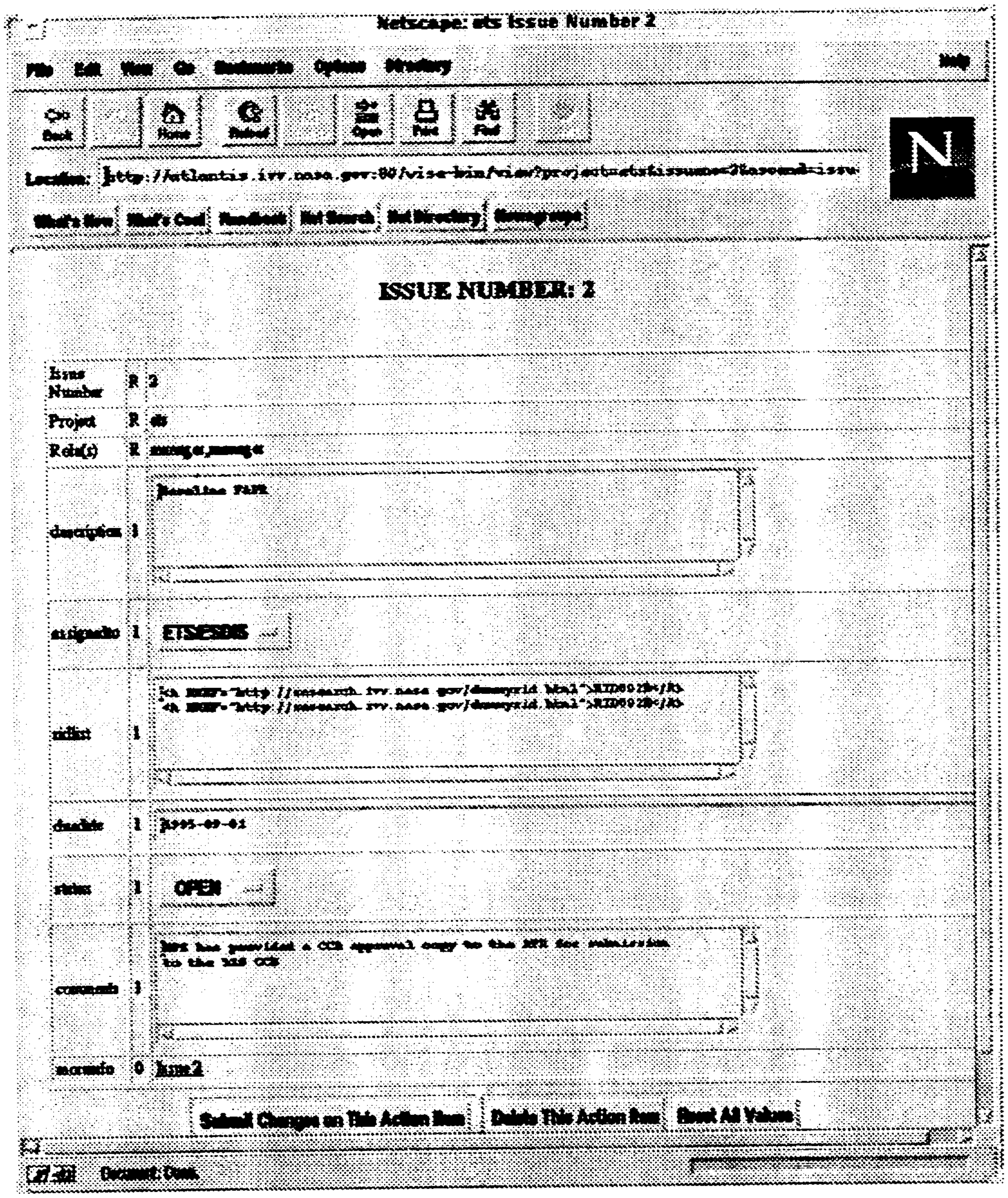
Automated support of measurement plays an important role in software process improvement. Project managers rely on a range of methods for measurement including status reporting and change management. Handling change requests, problem reports, activity log entries, and other issues in a software development effort becomes quite complicated even in small groups. We believe that automation plays a key role in increasing productivity, controlling quality, and introducing predictability into the software process. But the effective use of software technology is limited by ill-defined processes and poor process management. These problems must be addressed for us to be able to apply new tools and technology. Much research in this area has shown that incremental automation in software management and measurement is necessary to achieve successful software process improvement.

The focus of more recent approaches to process improvement has been through analysis and synthesis of *organizational experience*. Information collected about a workflow in a repository of such experience can be analyzed for specific organizations to help them improve weak areas and capitalize on strengths. The Experience Factory mechanism [2] is an approach to improving software development processes. The Experience Factory packages the experiences of an organization and measures various software process and products. WISE can be used to collect valuable information related to issue resolution in a database implicitly. This information sheds light on the issue solving capacity of a development group and its products. For example, modules with the most severe issues and the average time for solving issues are two statistics that can be obtained by analyzing software process change data.

3.1 Measurement

Software metrics can be defined as the continuous application of measurement-based techniques to the software development process and its products. Metrics supply meaningful and timely management information used to improve a process and its products [12]. Software metrics are standardized ways of measuring the attribute of software processes, products and services.

For example, the Goals-Questions-Metrics (GQM) approach [1] has been widely used to



Location: http://atlantis.ivv.nasa.gov:80/wise-bin/view?project=ets&issuenum=2&command=issu

ISSUE NUMBER: 2

Issue Number	R	2
Project	R	ets
Role(s)	R	manager,manager
description	1	Baseline FAPR
assignedto	1	ETSESDIS
ridlist	1	<A HREF="http://research.ivv.nasa.gov/dummyrid.html">RID0928</A> <A HREF="http://research.ivv.nasa.gov/dummyrid.html">RID0928</A>
duedate	1	1995-09-01
status	1	OPEN
comments	1	[illegible] has provided a CCB approval copy to the [illegible] for submission to the [illegible] CCB
moreinfo	0	Issue2

Figure 2: A WISE issue for the sample project (issue number 2)

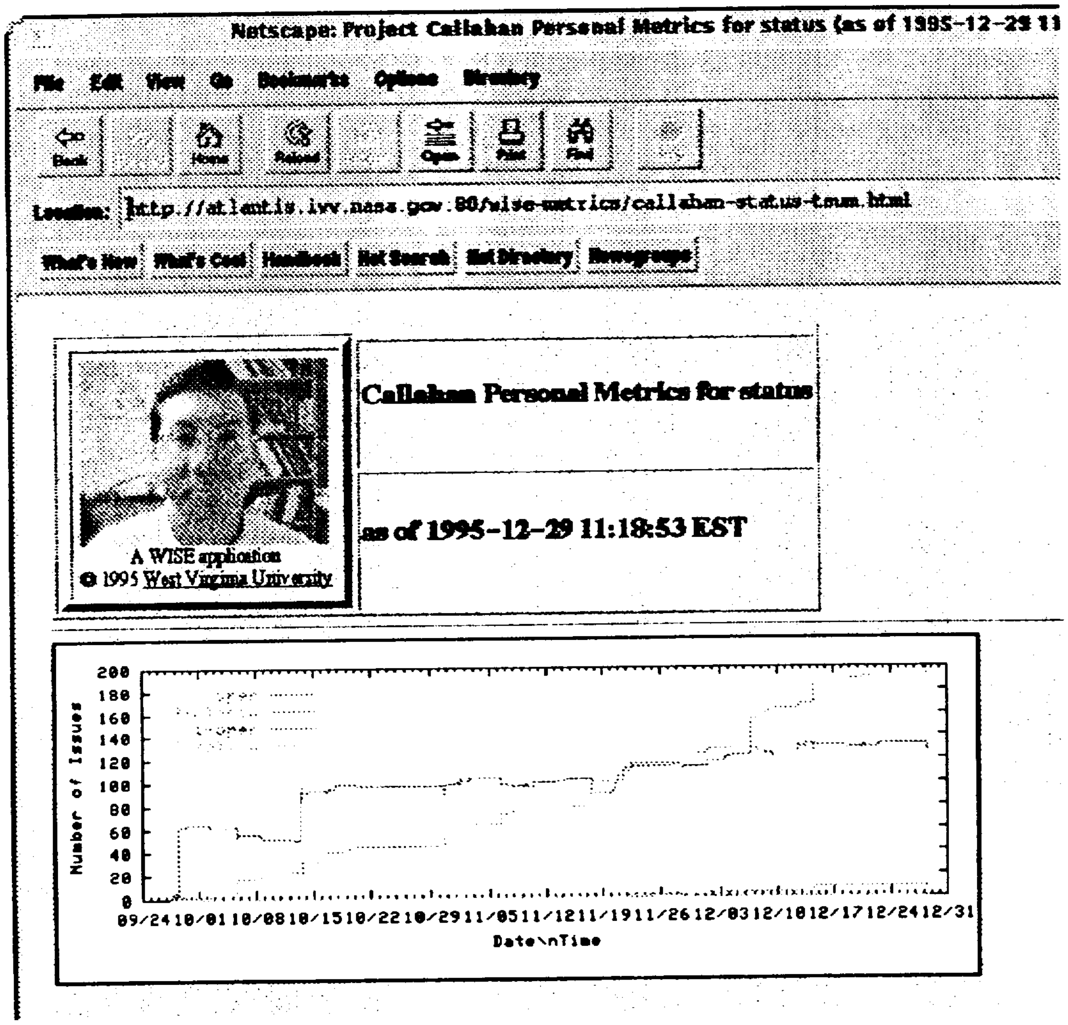

Netscape: Project Callahan Personal Metrics for status (as of 1995-12-29 11
File Edit View Go Bookmarks Options Directory
Back
Home
Reload
Open
Print
Find
Location: http://atlantis.ivv.nasa.gov:80/wise-metrics/callahan-status-tmm.html
What's New What's Cool Handbook Net Search Net Directory Newsgroups
A WISE application
© 1995 West Virginia University
Callahan Personal Metrics for status
as of 1995-12-29 11:18:53 EST
Number of Issues
200
180
160
140
120
100
80
60
40
20
0
09/24 10/01 10/08 10/15 10/22 10/29 11/05 11/12 11/19 11/26 12/03 12/10 12/17 12/24 12/31
Date\nTime

Figure 3: Time plot of open, solved, and closed issues on a project

select metrics based on their relation to overall organizational goals. Instead of collecting metrics at random, measurement is based on the goals of a project. To determine if goals are being achieved, a set of questions is constructed related to each goal. To answer each question, metrics are needed. The metrics are organized into this hierarchy to provide continuous assessment of a project's status. The technique is a powerful way to track progress towards project objectives. For over a decade, the Software Engineering Laboratory (SEL) at NASA's Goddard Space Flight Center has significant experience in measuring software processes for over a decade. Their studies have presented the result of collecting valid software engineering data [3]. Data collected at the SEL was based on the changes made to the software during development. It was later followed by evaluating software development processes through analysis of change data.

Metrics can be used to understand a software process, evaluate a software product and goals, control resources and products, and predict attributes of a software process in the future. To improve the software process, WISE collects useful data in terms of changes and responses made to issues generated during the normal workflow of a software project. An issue created in the workflow may be an idea, a question, an error, or any other identifiable "chunk" of communication between project participants. An issue is encapsulated in a form and stored in a backend database. Participants can view issues dynamically and generate metrics based on issue properties. Database queries can extract important information relevant to project metrics.

Some software metrics represent measures of the properties of a software process. Using the GQM approach, for instance, a metrics program can address many corporate needs including measurement of process maturity, a tools-evaluation database, and trend analysis [11].It is highly recommended to embed metrics tools in the existing development environment. Developers should understand the need for metrics, otherwise they will neither provide accurate data or use the results of metric analysis. Secondly, metrics should be kept close to the developers. This way the developers would be able to access measurements, evaluate them, and take action as part of standard operating procedure and without hindering schedule or budget. WISE metrics are generated implicitly and can be viewed at any stage of the project by all developers. A separate metrics group is not present that collects metrics and analyzes them. The time devoted to metrics collection and analysis is minimized by WISE. The front end of WISE is kept very simple so that developers need not become experts in measurement theory.

WISE generates analysis based on many aspects of a workflow model that are relative to the changes in issue forms. This avoids burdening users with the task of metric collection. It is important to remember that metrics can only show problems and give ideas as to what can be done. It is the actions taken as a result of analyzing the data that yield improvements to the process. This is the reason why it is critical for metrics users to understand that measurement is not the goal. The goal is improvement, through measurement, analysis, and feedback [6]. In [6], a practical view of software measurement that formed the basis for a company-wide software metrics initiative within Motorola has been described. The software process program must be defined in a precise powerful and rigorous formalism. Such an environment becomes a vehicle for the organization of tools for facilitating development and maintenance of the specified process [10].

WISE is programmable and can be customized to the specific process of an organization, but such processes are specified in terms of *behavioral descriptions* of people, roles, and form-based information. Such an approach yields an incremental, bottom-up definition of an organization's processes. There are many advantages to using behavioral descriptions to specify software process models [13]. The behavioral approach describes software development as a collection of activities or processes which may take place concurrently. This approach leads to better automation, message passing for communication, and provides greater visibility for software processes within an organization. For instance, one should try to describe the software process in terms of the events that occur during the development effort rather than changes to the product. Metrics should be based on analysis of these events. WISE addresses this by logging events in order to track issues throughout their lifecycle. Other software process tools, like Marvel [8], define processes from the top-down: they help in defining the detailed software configuration process by informing users which components are potentially affected before performing subsequent editing, compiling, and other coding activities. Tools such as Marvel assist in development and maintenance efforts through controlled automation, but do not address the source or motivation of changes throughout the lifecycle.

In our deployment of WISE, we have been careful to integrate it carefully into existing process environments. Many software tools play an important role in the software development process but use of a tool within a process changes the process itself. For example, some tools are used by users taking on a specific role. Other tools are used by users in multiple roles, during many activities and for processing documents of multiple users. Inserting an tool could have a significant impact on the development process. In order to control the insertion of the tool a method called "Tool Insertion Method" [5] has been proposed. The key elements of TIM are tracking the progress of a tool used to improve the process. Indeed, we have used WISE in the development of WISE itself to study the effects of such changes.

The issue-based approach used in WISE is dominant not only in problem tracking, but is critical in capturing design rationale, informal information, requirements, and change requests. Design dependencies can be represented in a issue-based style [9] and tracked throughout the lifecycle of a project. In all cases, participants in an issue-based discussion contribute their expertise and viewpoints to discover and resolve issues. Each issue is followed by one or more positions that respond to an issue. The issue-based model is now almost 20 years old. There is a clear need for automated tools for software management and measurement created by a focus on understanding, managing, and improving the software process.

4 Conclusions

WISE is an automated system that helps in software project management and measurement. We have described how such tools can help in improving software development processes through tracking and measurement of change activities on project issues. From our practical experience with WISE on several projects, we believe that such automated tools hold much promise for providing realistic assessments of software development projects particularly in large-scale projects with verification and validation contracts. While our use of WISE has been successful, many issues remain problematic including:

Metric frameworks More work is needed to construct metric hierarchies for organizations based on specific strengths, weaknesses, and corporate policies. For example, the skill levels of programmers in different groups within the same company vary widely. Such differences have a profound effect on the ability of measurement tools to predict performance.

Security and privacy These are a major concern in WISE. Several studies [4] have found high participation in automated measurement project by users who feel that they control access to workflow information. Users that feel they are being watched and judged by management will not use or they will circumvent such tools. From our experience, we also feel that measurement should focus on the individual user and let the user control the permissions and visibility of their workflow transactions. WISE allows users to define history visibility so that even managers cannot access change data without explicit permission from the user who owns the data.

Process validation The bottom-up approach to process definition through forms and roles sometimes creates "black holes" in the process where issues can remain unresolved. We have considered the use of finite-state machine model checking tools to find incomplete and inconsistent paths in the composite process as a part of the WISE system.

Although the definition of an organizational process through roles and the workflow of electronic forms may be indirect and imprecise, WISE provides a flexible platform for accommodating the processes changes. For instance. on several projects it is only after an initial prototype that we discovered that new roles were needed, additional issue fields had to be added, and new field had to be added to forms in a project. This incremental approach, however, is much more productive that trying to define a complete and consistent process from the top-down. Indeed, such changes reveal the progress (or lack of) in efforts to improve development processes. We continue to examine this "meta-view" of process improvement as well as collecting and analyzing results of current projects using WISE.

References

[1] V. Basili. Software modeling and measurement: The goal/question/metric paradigm. Technical Report CS-TR-2956, University of Maryland Computer Science Department. College Park, Maryland, September 1992.

[2] V. Basili. The experience factory and its relationship to other improvement paradigms. In *Proceedings of the 4th European Software Engineering Conference, Garmish-Partenkirchen, Germany*, September 1993.

[3] V. Basili and D. Weiss. A methodology for collecting valid software engineering data. *IEEE/ACM Transactions on Software Engineering*, SE-10(6):728-738. November 1984.

[4] M. et al. Bradac. Prototyping a process monitoring experiment. In *Proceedings of the 15th International Conference on Software Engineering*, pages 155-165, May 1993.

[5] T. Bruckhaus. Tim: A tool insertion method. In *Proceedings of the 1994 CAS Conference*, October 1994.

[6] M. Daskalantonakis. A practical view of software measurement and implementation experiences with in motorola. *IEEE Transactions on Software Engineering*. 18(11):998-1010, November 1992.

[7] W. Humphries. *Managing the Software Process*. Addison-Wesley, sei series in software engineering edition, 1989.

[8] G. Kaiser and P. Feiler. Intelligent assistance for software development and maintenance. *IEEE Software*, pages 40-49, May 1988.

[9] M. Lubras. Representing design dependancies in an issue-based style. *IEEE Software*, pages 81-89, July 1991.

[10] L. Osterweil. Software processes are software too. *Communications of the ACM*, May 1987.

[11] S.L. Pfleeger. Lessons learned in building a corporate metrics program. *IEEE Software*, pages 67-74, May 1993.

[12] L. Westfall. Software metrics that meet your information needs. In *Proceedings of the 4th International Conference on Software Quality*, October 1994.

[13] L. Williams. Software process modeling. In *Proceedings of the 10th International Conference on Software Engineering*, pages 174-186, April 1988.