A Generic Model and Tool Support for Assessing and Improving Web Processes

D. Rodriguez, R. Harrison, M. Satpathy
The University of Reading
Reading, RG6 6AY, UK
{d.rodriguez-garcia, rachel.harrison, m.satpathy}@reading.ac.uk

Abstract

Process assessment or process improvement is usually not considered in the context of Web applications, and the differences between Web applications and traditional applications are not usually emphasized. In this paper, we discuss a generic quality framework, based on a generic model, for evaluating Web processes. The aim is to perform assessment and improvement of web processes by using techniques from empirical software engineering. A web development process can be broadly classified into two almost independent sub-processes: the authoring process (AUTH process) and the process of developing the infrastructure (INF process). The AUTH process concerns the creation and management of the contents of a set of nodes and the way they are linked to produce a web application, whereas the INF development process provides technological support and involves creation of databases, integration of the web application to legacy systems etc. In this paper, we instantiate our generic quality model to the AUTH process and present a measurement framework for this process. Such a measurement framework can form the basis for assessment and improvement of the AUTH process. Furthermore, tool support is necessary to manage the volume of information associated with a process and the associated products. We also present such a tool to provide effective guidance to software personnel including developers, managers and quality assurance engineers.

1. Introduction

Web based systems and applications deliver a complex array of content and functionality to a broad population of end users. They require new approaches to design and development but present the same issues and challenges as traditional information systems. Therefore, the same software engineering techniques are still necessary but the process should take these differences into account.

Web-based applications differ from other applications from both the product and process point of view. As products, they differ from traditional systems in the following ways:

- Web based applications are distributed and component based, and are part of the client/server paradigm in the sense that they are composed of a series of components such as servers, databases, middleware, etc.
- High reliability: Web applications in general, and E-commerce applications in particular, must have high reliability in the sense that the server is expected to be available all the time.
- High Scalability: Web applications have the potential of attracting and reaching a very wide audience.
- High Usability: The users of Web applications are usually members of the general public, not technical experts. A Web application must have the potential to attract such users. Hence, usability and visibility of Web products must be high. Also, there are no geographical boundaries and so cultural and language issues need to be kept in mind.
- Security: In many Web applications (e-banking, e-commerce etc.), security is the prime concern.
- Saleability (advertising, web-site popularity etc.). Marketing is a major concern of most web applications. So many marketing ideas need to be incorporated in an application.

Web applications also differ from traditional applications from the *process* point of view: there are more technologies (HTML, XML, network protocols, multimedia, Java and script languages) and thus, many roles (authors, developers, graphic designers, legal issues etc.) that have to be managed. In addition, the shorter time to market, shorter product life cycles and continuous maintenance are much more pronounced in the case of Web applications as compared to traditional ones.

Process assessment and process improvement are the means to develop quality applications within time and budget. Our aim is to perform process assessment and process improvement of Web processes using techniques from Empirical Software Engineering such as quality models. Among the models that are available for process assessment and improvement, the most influential ones include the Capability Maturity Model (CMM) [11], ISO/IEC 12207 [13], ISO 9000 [16], the BOOTSTRAP

Proceedings of the Eighth IEEE Symposium on Software Metrics (METRICS'02)
0-7695-1339-5/02 $17.00 © 2002 **IEEE**

model [20], and the ISO 15504 (SPICE) [14]. These process models are high level and are not specific enough to cater to the needs of each of the individual processes since: (i) the nature of the processes vary widely; (ii) most of the models are more oriented towards enhancing the maturity of an organization and take a monolithic view of the overall development process; and finally (iii) the process models, while emphasizing process activities, often put too little importance on the products which are the results of the process activities. Product quality aspects are usually addressed by models such as ISO 9126 [15], FURPS+ model [10], etc. However, how such product models are related to the above process models, has not been properly addressed. A process model should aim to increase the quality of the products they produce; however, the relationship between the above models and product quality is far from clear.

In this paper, we will define a generic quality model for web processes with the aim of improving web products (in addition to improving the quality of web processes). To do this we adopt a generic process model [29]. The generic model is a template which could be instantiated to be the quality model of any individual process. We will demonstrate how this model could be instantiated to individual web processes and discuss how a measurement framework could be defined and used as the basis of process assessment and improvement. In addition, the volume of information that we can associate with the whole hierarchy of processes and products can be very high. Tool support becomes imperative when dealing with the processes under the generic quality model. Based on previous work [31], we have upgraded a project management tool to accommodate the distinctive features of the generic model for quality assessment and improvement.

Section 2 describes related work. Section 3 briefly introduces a framework for the web development process, discusses the generic process model for Web development and also instantiates the generic quality model to the authoring process. Section 4 discusses a measurement framework for the authoring process. Section 5 describes our tool support for the generic quality model. Finally, section 6 concludes the paper.

2. Related Work

2.1. Web Methodologies and Processes

Several methods are available in the literature for the development of web applications. The most popular ones are the Object-Oriented Hypermedia Design methodology (OOHDM) [30] and the Relationship Management Methodology (RMM) [12]. Both have their origin in hypermedia design based on HDM (Hypermedia Design Methodology) [9]; while OOHDM follows an object-oriented approach, RMM is based on the entity relationships model which can be more suitable for database driven applications. Web Processes are also described by Lowe and Hall [21], who provide a framework for the development of hypermedia applications. Web based projects are composed of activities such as formulation, planning, analysis, modelling, page generation, testing and customer evaluation. Web Engineering [21] is an adaptable, incremental (evolutionary) process populated by a set of framework activities that occur for all web-based business-critical systems. The framework includes domain analysis, product modelling, process modelling, project planning, development and documentation. The Hypermedia Flexible Process Modelling (HFPM) presented by Olsina [26] is another engineering-based approach, which includes analysis-oriented descriptive and prescriptive process modelling strategies. It describes existing processes giving guidelines for the planning and management of a hypermedia project.

2.2 Evaluation of Web Applications and Processes

We have classified the web development process into two almost independent sub-processes: (i) the *authoring process* (AUTH process), which creates the hypermedia and (ii) the process of developing the *infrastructure* (INF Process), which provides the interfacing with databases, security protocols, etc. Since we will use this model in our work, we describe it in more detail in the next section.

The most common approach to evaluation is to focus on usability [25]. For example, Garzotto et al [8] decompose *usability* into learnability, comprehensibility, memorability, handling ability and niceness. In turn, they decomposed these into other lower level factors. Olsina et al [27] have defined a more general approach defining a tree of quality attributes based on the ISO 9126 standard.

Fewer attempts have been made to evaluate web processes; most notable is the work of Lowe et al. [22] and Christodoulou et al. [5]. Christodoulou et al. compare various hypermedia methodologies and systems, and suggest which methodology is more suitable for certain type of applications. Lowe et al. define a series of *abstract tasks* which state what process entities are to be assessed, what aspects of the development process are involved and what activities need to be performed. These abstract tasks are defined in a template. Lowe [22] also proposes a reference model for identifying elements (resource, activities and artefacts) in various development process models in order to provide a common terminology. Mendes et al [24] have carried out empirical studies concerning estimates of Web authoring effort.

Bazzana and Fagnoni [3] describe PI3, a process improvement plan for an IT company. They used CMM to improve key process areas (KPA) such as: technical, business, organizational and cultural. They describe how a company at level 1 in CMM can address improvement plans in accordance with business objectives, including: (i) how they defined processes and tools for testing and configuration management, (ii) adoption of defined practices and tools, (iii) tailoring measurement plans according to the company's objectives and (iv) formalization of experience for constituting an initial QMS (Quality Management System).

2.3 Quality Models, Process and Product Assessment

The GQM method [1] proposes a measurement plan for assessing the quality of entities like products, processes or people. It starts with a set of business goals and the goals are progressively refined through questions until we obtain some metrics for measurement. The measured values are then interpreted in order to answer the goals. Fixed quality models such as McCall's quality model FCM (Factor-Criteria-Metric) [23], ISO 9126 [15] or flexible ones such as QMS (Quality Management System) [18] can be used to set goals and refine those goals into question obtaining an appropriate set of metrics. ISO 9126 [15] describes a generic model for specifying and evaluating the quality of software products. The model isolates six factors, called Functionality, Usability, Reliability, Efficiency, Maintainability and Portability; and the quality of a product is defined in terms of the quality of the above factors. Each factor may in turn be defined by a set of subfactors. The FURPS+ model [10] used by HP is similar to ISO 9126.

Focusing on product quality alone may not guarantee that an organization will deliver products of good quality. Products are created by processes. Many models have been developed based on the view that improving the quality of a process will deliver products of good quality. Prominent among them are the CMM [11] and ISO 9001 [16]. Models like BOOTSTRAP [20] and SPICE/ISO 15504 [14] are variants of the CMM. ISO/IEC 12207 [13] does a classification of all processes associated with software development and offers general guidelines which can be used by software practitioners.

Application of Metrics in Industry (**ami**) [19] combines CMM and the GQM method, and the result is that it provides a complete framework for process improvement. The **ami** approach is iterative, goal-oriented, quantitative, involves everyone in the organization and integrates necessary management commitment. It covers the whole of the process improvement cycle. CMM or other standards like Bootstrap, SPICE, ISO 9001 etc. are used to identify weak areas in the development process. This information along with the business and environment specific objectives is used to define some software process goals. The goals are validated and then refined into sub-goals. The sub-goals are further refined into metrics and a measurement plan is made to collect data. The data are then analysed and related to the original goal. Based on the data collected an action plan may be made to improve the development process. New goals are then defined and the cycle is repeated.

PROFES [17] follows an orthogonal approach to process improvement. It uses ISO 9126 to identify sub-factors related to product quality that need to be improved. Following a PPD (Product-Process Dependency) model, it then identifies the process attributes that need to be improved for achieving the desired product quality. Then an action plan is made following an ISO 15504 compliant method and the plan is executed.

Satpathy et al. [29] define a generic process model for process assessment and process improvement. The generic model is a template that can be instantiated to generate the quality model for any individual process like specification, coding, testing etc. and employs a duality relationship between product and process quality attributes. The duality relationship says that any process attribute in addition to having a process aspect may also have a product aspect. As an example, let us consider the *understandability* attribute. The two perspectives are (i) the concepts of the process should itself be understandable to the process executer and further (ii) the process should make its output product sets understandable. Since we are looking at product attributes from the process quality point of view, we may miss out some important product factors, which are not directly addressed by process quality factors. In order to alleviate this problem, the authors use major product quality models (such as ISO 9126, FURS+) as references while defining our generic process quality model. In this paper, we will upgrade the generic process model so that it could be a generic process model for web development processes. The resulting model will form the basis of a measurement framework for the process concerned.

3. Assessing Web Quality

3.1. A Web Development Framework

As mentioned earlier, the web development process can be broadly classified into two almost independent sub-processes or activities: the authoring process (AUTH process) and the process of developing the infrastructure development (INF process). The AUTH process concerns the creation and management of the contents of a set of nodes and the way they are linked to produce a web or hypermedia application. Apart from writing the web pages and linking them together, or possibly with databases, little technical knowledge is necessary for authors. On the other hand, the INF development process provides technological support and involves creation of databases, writing programs (CGI, Applets, servlets etc.), integration of other systems like legacy systems etc. to the web application.

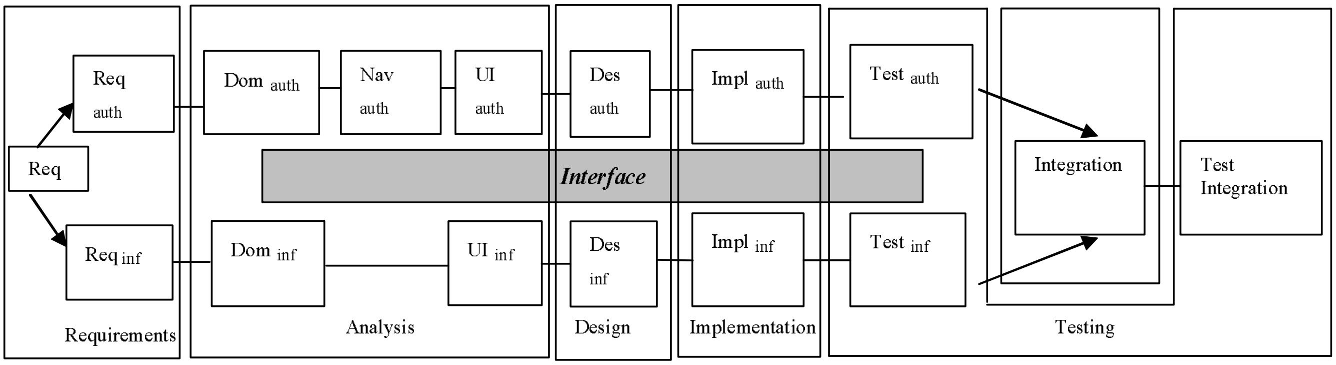


Figure 1 Web Development Workflow

This separation of concerns defines processes that could be developed simultaneously and independently with minimal interactions between them. This division is also reflected in the hypermedia terminology when defining the terms *application* and *system*. Application refers to the result of the development process (the end product), whereas systems refers to the tools and the infrastructure used in creating and supporting the applications [12].

This framework allows us to identify process steps and their artefacts. Figure 1 shows that the requirements capture phase collects all the web-related user requirements, from which the requirements for the AUTH process and the requirements for the INF process can be derived. After this separation, both the processes can proceed in parallel and almost independently

3.2. A Generic Quality Model

The modelling of web processes will identify specific issues (process steps and artefacts) associated with hypermedia and Web applications, which will make it possible to apply empirical software engineering techniques to analyse and evaluate a process. The feedback will allow us to improve the actual process.

As discussed earlier, the generic model can be instantiated to generate the process model for any individual process. The individual process models are then used to provide a measurement framework for the process concerned. The generic model is defined by a set of 8 factors (functionality, usability, efficiency and estimation, visibility and control, reliability, safety, scalability, maintainability) and each factor is further defined by a set of sub-factors. The model assumes that a process is a relation between an input product set and an output product set. For instance, the design process is defined by the following relation: *({requirement document, specification}, {design})*. Thus, the design process takes a requirement document and a specification and produces a design. Note that a process need not have a unique definition. For example, a design process need not refer to the requirement document and in such a case, the process definition would be: *({specification}, {design})*.

In this paper, we instantiate this generic model to the AUTH process, and based on the instantiated model, we will present a measurement framework for this process.

3.3. Customizing the TGPM to the AUTH process

Table 1 briefly outlines each of the subprocesses of the AUTH process. The subprocesses follow the approach of OOHDM [30]. As shown in Figure 1, once the requirement phase collects all the web related user requirements, requirements for the AUTH process (Req $_{auth}$) can be obtained from it. At the end of the AUTH process, we expect to obtain the hypermedia content of the application, i.e., a set of linked contents (HTML or XML pages, multimedia artefacts, etc.).

Req $_{auth}$	Req $_{auth}$ is obtained from Req, the overall web requirement.
Dom $_{auth}$	Identify from Req $_{auth}$ classes and relationships between them and obvious attributes. A preliminary class diagram for authoring is formed.
Nav $_{auth}$	Obtain from the class diagram for authoring, a navigational model [2]. This model describes navigational structure between the web pages.
UI $_{auth}$	Describe *how* the navigational structure is presented to the user.
Des $_{auth}$	The preliminary class diagram is completed.
Impl $_{auth}$	Implement contents (pages, sounds, pictures, etc), and links from the detailed design.
Test $_{auth}$	Test for syntax, lexical problems, image-related problems, document structure, etc. [3] (functionality, performance, compatibility, reliability, security, etc. could be verified after integration)

Table 1 Stages in AUTH Process

Once a Web process is defined according to the type of application, the generic quality framework can be instantiated to assess the process. Following the parallel framework defined in Section 3, we need a different quality models for the AUTH process and INF process. Here, we will use the generic quality framework for generating the quality framework (quality factors, sub-factors etc.) of the AUTH process. The definitions of the various factors and the sub-factors of the AUTH process are given in Appendix A.

Table 2 shows the quality tree of the AUTH process, which essentially enumerates the major metrics associated with the various sub-factors. Note that we have tailored the generic model given in [29]. For instance, we have added customer satisfaction because of its importance in Web applications. Similarly, we have removed the *Safety* sub-factor because it is not relevant to the AUTH process.

The quality model will help us to build goal trees in a systematic manner [7]. Because of the duality of the sub-factors, their definitions cover both the process as well as the product aspects of the goal. Each subfactor identifies related metrics and the actors responsible for the collection/ analysis of those metrics. Our tool stores all this data that can be used in the construction of the goal tree.

FACTOR	SUB-FACTORS	METRICS (Examples)
Functionality	Compliance	Strict use of the HTML/XML standards (Yes/No) No. of deviations from prescribed standard (e.g.)
	Completeness	Trend of missing features in the content No. of Use Cases implemented vs. planed over a no. of projects
	Generality	Additional features in content that the AUTH process supports No. of different types of authoring languages (HTML, XML) the process can use No. of different types of links the process can use
	Consistency	Trend of the no. of structural inconsistencies (e.g. ..)
	Correctness	Trace of the no. of mismatches between the design and the content over a number of projects
	Interoperability	No. of different types of systems to which the contents can be linked
Usability	Understandability	Effort to understand the AUTH process Effort to understand the products (presence of On-line help, manuals etc) No. of search related problems
	Learnability	Effort to do authoring (No. of hours/ page) Level of training required to do authoring
Efficiency and Estimation	Cycle time	Estimated cycle time vs. actual cycle Integration frequency (daily, weekly, etc.)
	Cost/Effort	Planned development time vs. actual time, Html pages/person month
	Complexity estimation	Estimated complexity vs. actual complexity (Compactness, Stratum [4], Graph complexity, etc.) Link complexity Psychological complexity (Subjective assessment)
	Schedule/Priority	No. of times schedules are not fulfilled
	Resource usage	Estimation resources vs. actual resources (people, tools)
Visibility and Control	Automatic checks and feedback	Interactions (e.g. no. of link failures, dangling links, inland pages, etc))
	Traceability	Subjective assessment of difficulty in tracing between analysis, design, and implementation
	Progress monitoring	No. of Milestones No. of times milestones are not met No. and kind of approaches to monitor progress
	Improvement measures	No. and type of improvement actions
Reliability	Failure frequency	Trend of MTBF (mean time between failures) over projects Trend of MTTF (mean time to failure) over projects No. of times failures encountered during authoring (tool crashing etc)
Scalability	Scalability	Plot of time, effort, cost against projects of varying sizes
Maintainability	Reusability	No. of modules reused vs. total no. of modules over a no. of projects
	Testability	Faults discovered during testing/Totals no. faults over a number of projects Trend of Faults/Lines of HTML/XML)
	Portability	No. of browsers and versions considered (e.g. palm appliances)
	Analysability	Effort to analyse causes of an authoring defect (e.g. broken links etc).
	Modifiability	Effort to correct an authoring defect
	Defect trend	Trend of the no. of defects in AUTH process Trend of the no. of incident reports Trend of the no. of change requests
Customer satisfaction	Communicativeness	Trend of customer complaints

Table 2 Quality factors/sub-factors/associated metrics of the AUTH Process

4. A Measurement Framework based on the Generic Model

A measurement framework will allow us to identify areas for measurement and improvement. The generation of our measurement framework for a web process is composed of four major steps: (i) define the process architecture (i.e. the process steps, the input products, and the artefacts) (ii) upgrade the generic model of [29] (add new subfactors which are of importance to web applications) to make it a generic model for web processes (iii) instantiate the upgraded generic model to the process concerned and (iv) identify the metrics associated with the subfactors of the instantiated model.

For the AUTH process, it is necessary to generate the process model which will elucidate the internal structure of this process. A process is any software activity that takes product(s) as input and produces product(s) as output. Formally, it could be defined as a relation from a set of products to another set of products; the set of relations from *m* input products to n output products could be denoted by [29]:

$$IP_1 \times IP_2 \times \ldots \times IP_m \leftrightarrow OP_1 \times OP_2 \times \ldots \times OP_n$$

where IP_i and OP_j are the types of the *i*-th input product and the *j*-th output product respectively.

We will now obtain the measurement framework for the AUTH process. As discussed in Section 3, Table 1 summarizes the internal structure of the AUTH process Thus, Table 1 takes care of step (i) of the measurement framework. Appendix A gives the definitions of the subfactors of the instantiated process model for the AUTH process. Such definitions take care of steps (ii) and (iii) of the measurement framework. Table 2 identifies the metrics associated with the subfactors in the instantiated model, and hence step (iv) of the measurement framework is complete and our measurement framework for the AUTH process is now ready. Any measurement method can use this framework. In particular, we will use the GQM approach here.

The GQM method helps us to identify, focus, document and analyse a small but relevant number of measurements that need to be collected on a regular basis. The method is composed of 3 steps. First, the conceptual level identifies a quality and/or productivity goal, i.e., the purpose of measurement in relation to an entity (product, process or resources) from a specific point of view (manager, developer, maintainer etc.). A goal can be an assessment goal or it can be an improvement goal. The second level, called the operational level, breaks down the goal by means of questions that characterise the entity. Finally, the quantitative level specifies metrics that need to be collected in order to answer those questions. All three steps are summarized in a tree structure, called the goal-tree [1].

Our measurement framework offers a systematic way of creating a goal tree [1]. Let us assume that there is a goal concerning the AUTH process for which we will obtain a goal-tree. We first identify what factor(s) of our instantiated model concerns the goal. Then from Table 2, we identify the relevant subfactors. The definitions of the subfactors (as in Appendix A) will generate the questions of the goal-tree. Finally, from Table 2, we can obtain the appropriate metrics. Note that, because of the duality of the sub-factors, the definitions cover both the process as well as the product aspects of the goal. Further, from Table 1, we can easily identify the participants who will collect and/or analyse the metrics of the goal tree.

Let us take the example of the following two assessment goals: (i) Assess the estimation of the AUTH process from the viewpoint of the manager, and (ii) Assess the maintainability of the AUTH process from the viewpoint of the manager/ maintainer. Tables 3 and 4 summarize the corresponding goal-trees.

Goal	*Object of Study*: **AUTH process** *Purpose*: **To assess** *Focus*: **Efficiency and Estimation** *Point of view*: **Manager**
Q1	What is the underlying authoring language?
M1.1	*Note the languages used (HTML, JavaScript, XML, Xlink etc.).*
Q2	Does the estimated complexity match the actual complexity? What is the technique used for complexity estimation?
M2.1	*Note the difference between estimated/ actual complexity*
M2.2	*Note the technique (e.g. Stratum [4], Compactness [4] Depth [4] Imbalance [4])*
Q3	Does the estimated authoring cycle time match the actual cycle time? What is the technique to estimate the cycle time?
M3.1	*Note the difference (actual cycle time/ estimated cycle time)*
M3.2	*Note the technique*
M3.3	*Integration frequency (how often the modules of the application are integrated)*
M3.2	Average cycle time
Q4	Does the estimated effort match the actual effort? What is the technique to estimate the effort?
M4.1	*Note the difference (actual effort/ estimated effort)*
M4.2	*Note the technique*
Q5	Does the estimated schedule match the actual schedules? What is the technique to estimate the schedule?
M5.1	*No. of times schedules are not fulfilled*
M5.2	*Note the technique for estimating the schedules*
Q6	Do the estimated resources match the actual resources? What is the technique to estimate the resources?
M6.1	*Note the differences (estimated no. of people/ actual no. of people)*
M6.2	*Note the technique for estimating the resources*

Table 3 GQM to assess the navigational design

Goal	*Object of Study*: **AUTH process** *Purpose*: **To assess** *Focus*: **Maintenance** *Point of view*: **Manager/Maintainer**
Q1	Is the current analysability performance acceptable? Is the tool-support adequate?
M1.1	*Effort to analyse the cause of an authoring error (link, misspelling etc.)*
M1.2	*Efficiency of the tool support (subjective assessment)*
Q2	Is the current modifiability performance acceptable?
M2.1	*Average time for addressing a change request*
M2.2	*Average time for addressing an incident request*
Q3	Are the contents free of defects? Is the process defect-free itself?
M3.1	*Trend of defects in the contents after release*
M3.2	*Trend of process defects and their cause (e.g. a tool crashing)*
M3.3	*No. of change requests after release for each project*
Q4	Does authoring addresses portability issue adequately?
M4.1	*No. of browsers that the process addresses*
M4.2	*No. of versions that the process addresses*
Q5	Is authoring testing adequate?
5.1	*No. of defects discovered during testing/ Total number of estimated faults*
Q6	Is the AUTH process re-usable?
M6.1	*Are the same AUTH process steps used for all projects (Yes/No)*

Table 4 GQM for maintainability

Usually, assessment is done to characterize the current status of a process. If the assessment reveals that a certain aspect of the process is not satisfactory, then this needs to be improved. Following the **ami** terminology [19], this is done by defining a *change* goal. A goal-tree can be constructed in relation to this change goal by following the GQM method. From this goal-tree, improvement actions can be inferred, which may correct the unsatisfactory aspect(s) of the process. Consider an example. Suppose the assessment finds that the actual cycle time of the AUTH process is too high as compared to the estimated cycle time. Then the improvement goal can be: *improve the estimated cycle time of the AUTH process from the viewpoint of the manager*. Then we need to find out what improvement actions need to be taken so that cycle time could be improved.

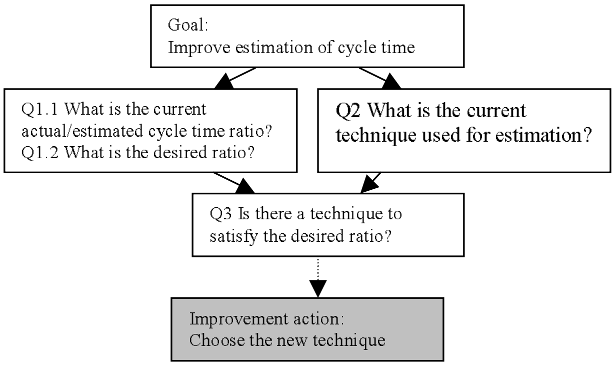

Goal:
Improve estimation of cycle time
Q1.1 What is the current actual/estimated cycle time ratio?
Q1.2 What is the desired ratio?
Q2 What is the current technique used for estimation?
Q3 Is there a technique to satisfy the desired ratio?
Improvement action:
Choose the new technique

Figure 2 GQM Tree for an Improvement Goal

The goal-tree for this improvement goal is shown in Figure 2 [6]. From this goal-tree, we obtain the improvement action which is then integrated to the AUTH process, and then another cycle of assessment follows. This new assessment verifies whether the improved AUTH process produced the desired result.

5. Tool Support for the generic quality model

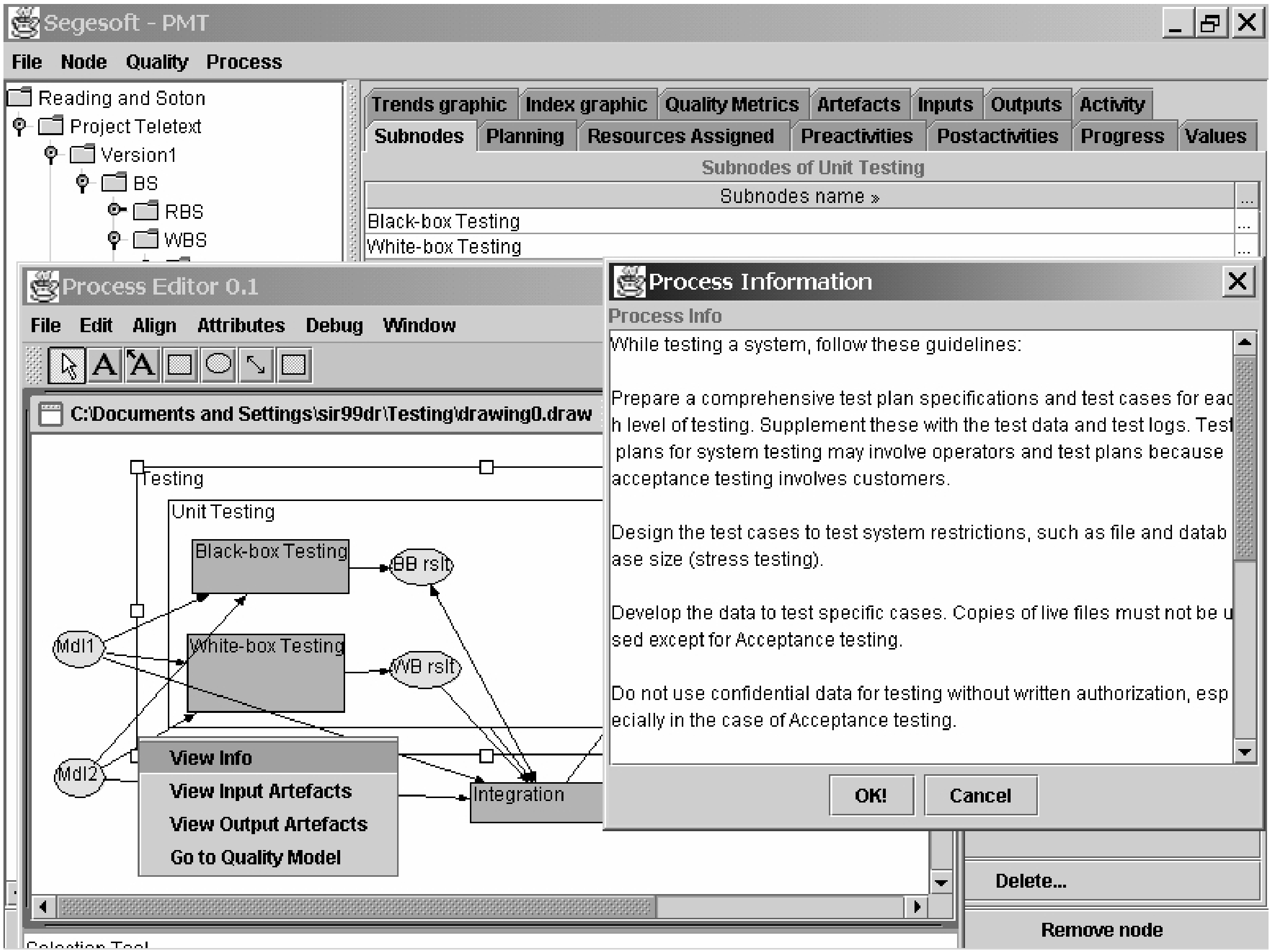
SEGESOFT is a project management tool [31], which provides an environment for training project managers. The system collects and records both actual and simulated project data and implements different techniques such as machine learning, project tracking, dynamic modelling, etc. The basic assumption of this work is that management decisions should be supported by integration of different sources of information. We have upgraded this tool so that it can incorporate features of the generic model. We will refer this upgraded tool as Upgraded SEGESOFT (or USEGESOFT) [28].

As we have discussed previously, it is necessary to store a lot of information in relation to a process and the associated products. Following the conventions of the generic model, we may need to keep the following categories of information in relation to a process.

1. A process is either an atomic process or it is defined in terms of subprocesses. Satpathy et al in [29] define a process in terms of subprocesses. A subprocess may be broken down further. At the lowest level, a subprocess is an atomic process, and it is defined in terms of a set of guidelines. So, documentation is necessary to specify the process structure.
2. A process is defined as a relation between a set of input products and a set of output products. Some documentation is necessary to say what such products are.
3. A process can have a hierarchical structure. Each process or subprocess takes a set of input products and a set of output products. If we follow the ISO 9126 product quality model, then each such product can have quality factors and subfactors. Further each subfactor may be associated with metrics. So we need to store all such information which may be associated with the products.
4. Any process can have an instantiated model. Then, in relation to the instantiated model, each process will have quality factors and subfactors. Further corresponding to each subfactor there may be metrics. So, each process should store all such information.

The USEGESOFT tool provides a database where all information associated with processes, products and their relationships can be stored. It also provides a graphical editor for drawing process diagrams in order to build and maintain the internal structure of processes. In Figure 3 we can see the testing subprocess in the graphical editor which supports 'drag and drop' mechanism. USEGESOFT also provides documentation mechanism to specify, store and display all the four types of information which can be associated with a process or subprocess. This information can remain hidden, or it can be displayed from the main window or through a pop up menu. For example, when a process component is selected in the Process Editor, a context menu associated with it allows us to store or visualize all information associated with that component. The snapshot of Figure 3 also shows the high-level information of the testing process through a pop-up menu. It is also possible to follow the pop-up menu in a nested fashion to extract further information. Alternatively, information can also be obtained by following the trail of entities from the main window. Figure 4 shows how definitions of various process quality attributes according to the generic model, along with their dual definitions, can be obtained.

Depending on a requirement, we may need to obtain relevant information from the database. Further, we may need to use this information to present a graphical display or make inferences by using some statistical modelling approach. Currently, as quality information is stored using the open standard XML (Extensible Markup Language) [32] we are able to extract this information manually or transform it using XSL (Extensible Stylesheet Language) [33]. In this way it is possible to manipulate or display the data using a spreadsheet or other data processing applications.



Subnodes of Unit Testing

Subnodes name »
Black-box Testing
White-box Testing

While testing a system, follow these guidelines:

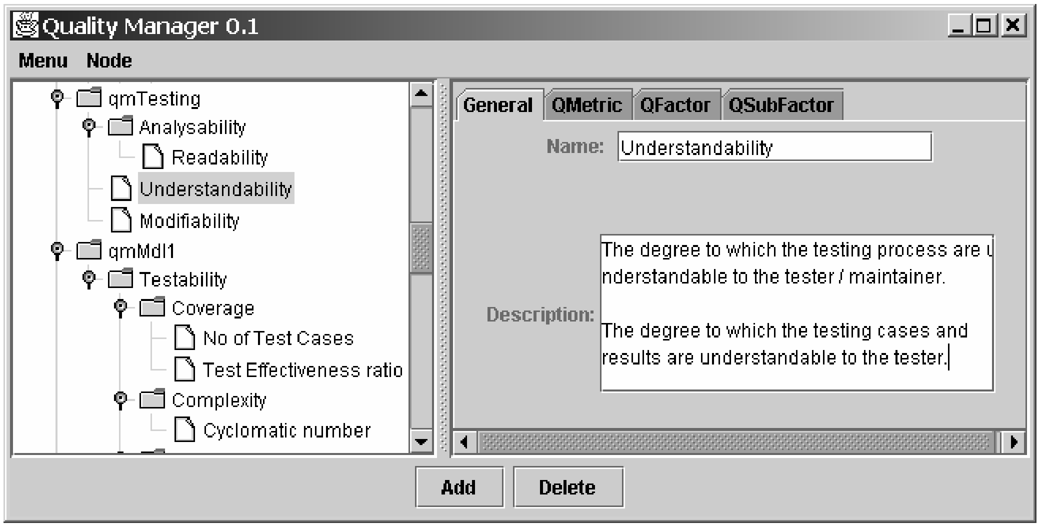
Prepare a comprehensive test plan specifications and test cases for eac
h level of testing. Supplement these with the test data and test logs. Test
plans for system testing may involve operators and test plans because
acceptance testing involves customers.

Design the test cases to test system restrictions, such as file and datab
ase size (stress testing).

Develop the data to test specific cases. Copies of live files must not be u
sed except for Acceptance testing.

Do not use confidential data for testing without written authorization, esp
ecially in the case of Acceptance testing.

Figure 3 Accessing the process information from the popup menu in the Process Editor



Name: Understandability

Description: The degree to which the testing process are u
nderstandable to the tester / maintainer.

The degree to which the testing cases and
results are understandable to the tester.

Figure 4 Process quality attributes according to generic model

6. Conclusions and Future Work

In this paper, we have discussed a framework for web application development and the generic model of [29] for assessing and/or improving any process. We have tailored the generic model to assess and/or improve any of the processes (or subprocesses) of the overall Web development processes. In particular, we have instantiated the generic model to obtain the instantiated model for the AUTH process. Based on the instantiated model, we then created a measurement framework for the AUTH process. We have also shown how the GQM method in conjunction with the measurement framework could be used to obtain a measurement or an improvement plan for the AUTH process. Finally, we presented USEGESOFT [28] which provides tool support for storing various quality information associated with the instantiated process models.

Future work will consist of validating the model as well as the measurement plan through industrial case studies. Currently, we are applying our methodology to a museum web application. As part of our on-going work, we are also automating the information retrieval and display in USEGESOFT by using a script language.

Acknowledgments

The authors wish to acknowledge The University of Reading, the EPRSC (EMPAF project ER/L87347) and CICYT, Spain (ARGO Project TIC2001-1143-C3). The authors also wish to thank J. Dolado, the Principal Investigator of the ARGO Project, for useful suggestions.

References

[1] V. R. Basili, G. Caldiera, and H. D. Rombach, "The Goal Question Metric Paradigm", in *Encyclopedia of Software Engineering*: John Wiley & Sons, Inc., 1994, pp. 528-532.

[2] H. Baumeister, N. Koch, and L. Mandel, "Towards a UML Extension for Hypermedia Design", in *UML'99 - The Unified Modeling Language. Beyond the Standard. Second International Conference*, vol. 1723, *LNCS*, R. France and B. Rumpe, Eds. Fort Collins, CO, USA: Springer-Verlag, 1999, pp. 614-629.

[3] G. Bazzana and E. Fagnoni, "Process Improvement in Internet Service Providing", in *Better Software Practice for Business Benefit. Priciples and Experience*, C. T. Richard Messnar, Ed.: IEEE CS, 1999, pp. 267-279.

[4] R. A. Botafogo, E. Rivlin, and B. Shneiderman, "Structural Analysis of Hypertexts: Identifying Hierarchies and Useful Metrics", *ACM Transactions on Information Systems*, vol. 10, pp. 142-180, 1992.

[5] S. P. Christodoulou, G. D. Styliaras, and T. S. Papatheodrou, "Evaluation of Hypermedia Application Development and Management Systems", Proc. of the 9th ACM Conf. on Hypertext and Hypermedia, Pittsburgh, PA USA, 1998.

[6] C. Debou, "Goal Based Software Process Improvement Planning", in *Better Software Practice for Business Benefit. Priciples and Experience*, R. Messnar and C. Tull, Eds.: IEEE CS, 1999, pp. 107-150.

[7] A. Fuggetta, L. Lavazza, S. Morasca, S. Cinti, G. Oldano, and E. Orazi, "Applying GQM in an Industrial Software Factory", *Acm Transactions on Software Engineering and Methodology*, vol. 7, pp. 411-448, 1998.

[8] F. Garzotto, M. Matera, and P. Paolini, "Abstract tasks: a

tool for the inspection of Web sites and off-line hypermedia", Proc. of the 10th ACM Conference on Hypertext and hypermedia, Darmstadt Germany, 1999.

[9] F. Garzotto, P. Paolini, and D. Schwabe, "HDM: A Model-based Approach to Hypertext Application Design", *ACM Trans. on Information Systems*, vol. 11, pp. 1-26, 1993.

[10] R. B. Grady, *Practical Software Metrics for Project Management and Process Improvement*. Prentice Hall, 1992.

[11] W. S. Humphrey, "Introduction to Software Process Improvement", Software Engineering Institute CMU/ SEI-92- TR- 7, June 1992 (Revised June 1993).

[12] T. Isakowitz, E. A. Stohr, and P. Balasubramanian, "RMM: A Methodology for Structured Hypermedia Design", *Communications of the ACM*, vol. 38, pp. 34-44, 1995.

[13] ISO, "Information technology-Software life cycle processes", ISO/IEC 12207, 1995.

[14] ISO, "Information technology-Software process assessment", ISO/IEC TR 15504, 1998.

[15] ISO, "Information technologys-Software product evaluation", ISO/IEC 9126, 1991.

[16] ISO, "Quality management and quality assurance standards", ISO 9000-1, 1994.

[17] J. Jarvinen, S. Komi-Sirvio, and G. Ruhe, "The PROFES Improvement Methodology - Enabling Technologies and Methodology Design", Product focused software process improvement: Second International Conference, PROFES 2000, Oulu, Finland, 2000.

[18] B. A. Kitchenham, J. D. Walker, and I. Domville, "Test specification and quality management - design of a QMS sub-systemfor quality requirements specification", Project Deliverable A27, Alvey Project SE/031, November 1986 1986.

[19] A. Kuntzmann-Combelles, K. Pulford, and S. Shirlaw, *A Quantitative Approach to Software Management: the AMI Handbook*. Addison-Wesley, 1996.

[20] P. Kuvaja, *Software process assessment and improvement : the BOOTSTRAP approach*. Blackwell Business, 1994.

[21] D. Lowe and W. Hall, *Hypermedia and the Web: An Engineering Approach*. John Wiley & Sons, 1999.

[22] D. B. Lowe, A. J. Bucknell, and R. G. Webby, "Improving Hypermedia Development: A Reference Model-Based Process Assessment Method", Proceedings of the 10th ACM Conference on Hypertext and Hypermedia, Darmstadt, Germany, 1999.

[23] J. A. McCall, P. K. Richards, and G. F. Walters, "Factors in Software Quality", US Rome Air Development Center Reports, NTIS AD/A-049 014, 015, 055 RADC TR-77-369, 1977.

[24] E. Mendes, N. Mosley, and S. Counsell, "Web Metrics-Estimating Design and Authoring Effort", *IEEE Multimedia*, vol. 8, pp. 50-57, 2001.

[25] J. Nielsen, *Designing Web Usability*. New Riders, 2000.

[26] L. Olsina, "Building a Web-based Information System applying the Hypermedia Flexible Process Modeling Strategy", 1st International Workshop on Hypermedia Development, Brisbane, Australia, 1998.

[27] L. Olsina, D. Godoy, G. J. Lafuente, and G. Rossi, "Specifying Quality Characteristics and Attributes for Websites", 1st Workshop on Web Engineering, Los Angeles, USA, 1999.

[28] D. Rodriguez, R. Harrison, M. Satpathy, and J. Dolado, "Tool Support for the Typed Generic Process Model", 11th International Conference on Software Measurement, Montreal, Canada, 2001.

[29] M. Satpathy, R. Harrison, C. Snook, and M. Butler, "A Generic Model for Assessing Process Quality", 10th International Workshop on Software Measurement (IWSM'00), Berlin, Germany, 2000.

[30] D. Schwabe, G. Rossi, and S. D. J. Barbosa, "Systematic Hypermedia Application Design with OOHDM", 7th ACM Conference on Hypertext and Hypermedia, Bethesda, MD USA, 1996.

[31] *The SEGESOFT Group*, "Integration of Information in a Training Environment for Software Project Management", Software Quality Management 2001 SQM'01, Loughborough, UK, 2001.

[32] W3C, "Extensible Markup Language (XML) 1.0. W3C Recommendation", W. Consortium, Ed., 1998.

[33] W3C, "XSL Transformations (XSLT). Version 1.0 W3C Recommendation", W. Consortium, Ed., 1999.

Appendix A – Definitions of the Sub-factors of an Authoring Quality Model

(a) Sub-factors of **Functionality**

Compliance:

(i) The degree to which the AUTH process conforms to prescribed (IEEE, ISO or company-specific) standards.

(ii) The degree to which the AUTH process conforms to its own model (i.e. it consists of a set of sub-processes and the sub-processes in turn consisting of sub-processes or sets of steps).

Consistency:

(i) The degree to which the AUTH process uncovers contradictions in the requirement document input (i.e. Req_{auth}).

(ii) The degree to which the AUTH process does not introduce contradictions in the hypermedia content.

Completeness: the degree to which the AUTH process transforms all of the features of the input requirements into the hypermedia content.

Generality (robustness): The ability of the AUTH process to address by over-specifying/ over-implementing conditions, which are not covered by the input product specification but are relevant to its context.

Correctness: The degree to which the process makes the functionalities of the hypermedia content match accurately to the features requested.

Inter-operability: The degree to which the AUTH process contributes to the ability of the product to interact with specified systems.

(b) Subfactors of **Usability**:

Understandability:
(i) The effort with which a typical author understands the logical concepts of the AUTH process.
(ii) The degree to which the AUTH process contributes to the understandability of the hypermedia content.

Learnability:
(i) The effort required for a typical author to learn to use the process.
(ii) The degree to which the AUTH process makes the hypermedia content easy to use (through informative help messages, good user interface etc.).

Operability:
(i) The effort required for a typical author to perform the process steps of the AUTH process.

(c) Sub-factors of **Efficiency & Estimation**:

Cost/ Effort estimation: The degree to which the cost/effort of AUTH process remain within a specified range and the ability of the process to support their estimations.

Cycle Time estimation: The degree to which the AUTH process meets its expected cycle time and the ability of the process to support its estimation.

Complexity estimation: The ability of the process to support the prior estimation of various forms of complexity (Structural Complexity [4], Navigational Complexities, Cognitive Complexity etc), and the degree to which the estimates are accurate.

Schedule/ Priority estimation: The priority of various stages in the process and the ability of the process to support their estimations and scheduling.

Resource Estimation: The degree to which a process keeps its resource usage in a specified range, and its ability to support their estimations.

Process Maturity: The CMM maturity level and/or any ISO certification of the organisation.

(d) Sub-factors of **Visibility & Control**:

Progress Monitoring: The ability of the AUTH process to facilitate monitoring of its progress at any point of time during the process execution to show that progress so far has been correct and effective (e.g. Work product analysis, PERT charts etc.).

Automatic Feedback: The ability of the AUTH process to provide feedback data and to support corrective actions if necessary.

Improvement Measures: The ability of the AUTH process to support the analysis of feedback data in combination with the data of previous runs and improve itself, or result in the improvement of a sibling process, continuously.

(e) Sub-factors of **Reliability**:

Failure Frequency:
(i) The number of (and the interval between) failures encountered during the AUTH process (e.g. tools used for authoring may crash).
(ii) The degree to which the AUTH process makes the hypermedia contents failure-free.

(f) **Scalability**:

The degree to which the AUTH process maintains its efficiency level in terms of time, cost and resource usage in handling problems of large sizes.

(g) Sub-factors of **Maintainability**:

Analysability: The effort with which a typical author can analyse the cause of a fault, process failure or unexpected feedback data. A fault could be one of the following categories.
(i) the fault is discovered during the process execution and the fault may be with the process itself, or it may be from the requirements.
(ii) the fault is discovered at a later point in time, and the cause of the fault is linked to the AUTH process.

Modifiability: The effort with which a maintainer addresses failures, detection of faults and unexpected feedback data during the AUTH process; or faults discovered later but linked to the process.

Stability:
(i)The degree to which addressing a process fault adversely affects the process itself (say, process efficiency or properties like process consistency).
(ii)The frequency of changes done to the process (less the number of changes, more is the stability).

Testability:
(i) The degree to which the AUTH process could be validated itself (success history of its hypermedia contents).
(ii) The degree to which the process contributes to the testability of its artefacts or the final product (for generating test cases).

Defect Trend:
(i) The trend of defects that are observed in the AUTH process itself (defects linked to the process - during or after process execution).
(ii) The degree to which the AUTH process detects defects or deficiencies in the Req_{auth} so that defects in the hypermedia content are minimal.

Reusability:
(i) The degree to which the process contributes to the reusability of the hypermedia contents.
(ii) The degree to which components of the process could be reused in a different context (e.g.: testing process of HTML pages can be applied to WAP pages).

Portability: The degree to which the AUTH process contributes to the portability of the hypermedia contents; i.e., the process should facilitate the migration of the product to a different environment (by taking into account different versions, releases).