Building Web Solutions with the Rational Unified Process: Unifying the Creative Design Process and the Software Engineering Process

A Rational Software & Context Integration white paper

TABLE OF CONTENTS

Abstract

This paper describes how the Rational Unified Process™ developed by Rational® Software can be used for the development of Web applications. This paper focuses particularly on the front-end of the lifecycle, and how to integrate the creative design process with the software engineering process of the Rational Unified Process. It reflects the experiences and the extensions of the Rational Unified Process made by Context Integration, a leading Web solutions integrator, delivering Web solutions to their Fortune 1000 client using their Web Opportunity Workshop (WOW™) and their Web optimized methodology, inContext™.

Introduction

A new generation of software development has taken shape over the last few years, sometimes referred to with terms such as e-Commerce, e-business, and the like. These Web applications provide customers and businesses with functionality such as on-line transaction processing, supply chain integration, dynamic content, and personalization. These systems have robust, flexible, and scalable architectures, designed to handle the performance and load necessary for the reality of high and unpredictable volumes. Typically these architectures are multi-tiered, providing encapsulation of business logic and integration of multiple disparate, "legacy" systems. The building of these sorts of Web applications is the subject of this paper. In the context of this paper we will refer to this sort of software system as a ***Web solution.***

Developing Web solutions has a lot of similarities to development of other software applications, but also differs significantly in a number of areas. In many ways what is being created is a form of consumer media, like a movie or video. Like any consumer product, the look and feel of the Web solution is absolutely critical and necessitates the involvement of a new set of people, including marketing, creative designers, and business executives. Therefore, the challenge for the Web developer is not just the new technologies, but using a process that will facilitate bringing this new set of stakeholders together. This paper is intended to serve as a roadmap to successfully building Web solutions with compelling Web interfaces by leveraging the Rational Unified Process as the fundamental process.

While this paper focuses on the importance of the creative design process, we also want to point out the importance of other disciplines, such as architecture, configuration management, and testing. For example, too little focus on the architecture, or lack of early testing of the executable architecture, is likely to lead to a brittle architecture, and an architecture that will not accommodate required load. Towards the end of this paper we briefly discuss some of the best practices that are essential for successful execution of Web-based projects.

Integrating Creative Design with Traditional Software Development

Never before has so much attention been paid to the user interface of software applications, as it is today with the Web. And for good reason, never before have the stakes been so high. In previous times and technologies, the audiences were, to varying degrees, captive audiences. Now, instead of being captive, the market economy of the Web has much more influence over the success of a software-based business endeavor. Why? Because, in the new marketplace of Web applications, customers are not purchasing software and installing it on their machines, they are surfing to it and deciding in an instant whether it is compelling or not. Making a Web application visually compelling is one of the keys to a successful Web solution, so it is essential that the creative design process be integrated with the overall software development process.

Use Cases Unify the Creative and Software Engineering Process

Web solutions extend traditional software development by marrying the world of art and creative design with the world of software engineering. These two worlds differ in terms of process, skills, and culture.

These differences can often be a major stumbling block on Web projects as these cultures clash. Use cases[1] provide a way to express, in common terms, a shared understanding of the expected behavior of the Web application.

Use cases provide the lingua franca for projects – a language that everyone connected with the project uses to describe what the Web solution will do – users, managers, art directors, architects, and programmers. It allows everyone to speak in terms of the business solution to be solved, and everyone delivers according to this specification.

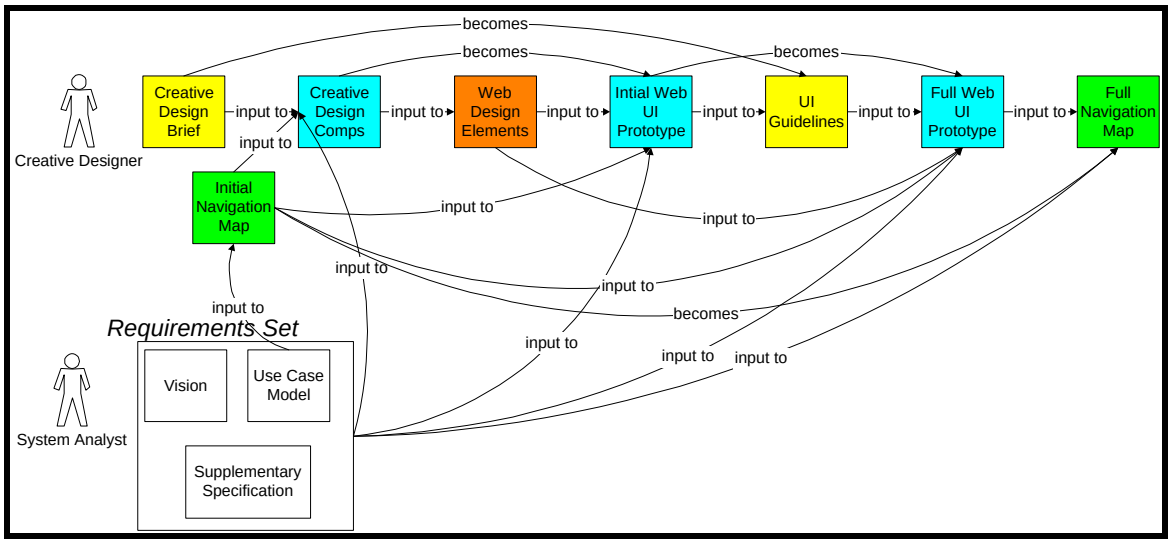

Figure 1: Integrating the Creative Design Process and the Rational Unified Process

Requirements

Building Web solutions involves a variety of stakeholders more so than other software application development. These stakeholders will usually include business executives, marketing, creative design, customer support, and the technology development team, among others. Having a process that facilitates communication among these varied stakeholders is critical to success.

Successful Web solutions start with a compelling ***Vision*** statement. The vision statement must be developed personally by the stakeholders of the solution, ensuring agreement on the project's goals from the outset.

The Vision serves several purposes:

- It achieves agreement on what problems need to be solved.
- It defines the boundaries of the system.
- It describes the most important features of the system.

Once the vision is agreed upon, facilitated workshops are held with the stakeholders to further identify the users of the system and services the system should provide to those users. The users and services are documented using the Unified Modeling Language (UML); users are represented by the UML as ***Actors***,[2] while services are represented using the UML concept of ***Use Cases***. This way of documenting requirements makes it easy for end users to articulate what services are needed, and for the development team to validate that requirements are complete and accurate. Use cases also provide a consistent thread through the development process, as we shall subsequently see.

[1] A use case is a textual description that defines a sequence of actions a system performs that yield an observable result of value to a particular user of the system.

[2] An **actor** defines a coherent set of roles that users of the system can play when interacting with it. A user can either be an individual or an external system.

Whenever possible, be sure to re-use use cases developed from previous Web solution projects. This is important to speed the process of arriving at the appropriate solution in "Web-time". Like design and analysis patterns, there are also use-case patterns. For example, every retail store on the Web will have many of the same use-cases. Do not reinvent the wheel. Leverage use-cases from past projects to deliver in Web-time.

In parallel to the development of the use cases for the system, non-functional requirements are captured in the ***supplementary specification*** and a set of common terminology is documented in the ***glossary***. The supplementary specification captures those requirements related to general functionality (to the whole system), usability, reliability, performance, security, Web hosting and supportability. The glossary ensures that all project members have the same view of the meaning of important concepts.

Creative Design Brief

Developing Web solutions requires an increased focus on the creative design of the user interface. In parallel to identifying actors and use cases, the initial ***user interface guidelines*** are developed; these high level guidelines are often referred to as a ***Creative Design Brief*** in the Web community. The Creative Design Brief defines:

- The mood of the site (e.g. does the site convey authority, playfulness, or service? Is it conservative or provocative?)
- How users will be accessing the site (e.g their connection speed)
- The browsers the users will be using
- Whether the site will use frames
- Any color limitations the site will have
- If applicable, a graphics standards guide (including standards on logos and all corporate colors)
- What sort of "bells and whistles" are wanted (e.g., mouse-overs, animation, news feeds, multimedia, etc.)

Navigation Map

The ***navigation map*** is a view of the Web solution showing how users of the site will navigate it, represented in a hierarchical "tree" diagram. This sort of diagram is often referred to as a "site map", but we have chosen to call it a navigation map, because of the multiple definitions given to the phrase "site map". Each level of the diagram shows the number of clicks it takes to get to that screen/page. Generally, you want to have the most important areas of the Web site only one click away from the first page (commonly known as the "home page").

Producing a Web site navigation map early in the project provides a valuable communication vehicle between stakeholders and the development team. Users are able to better envision how the Web application will be navigated, and creative designers are better able to understand the necessary navigational scheme. The navigation map naturally evolves from the use-case model, since the use-case model describes what services the system provides the end users. For this reason, the navigation map should be created after the initial use-case model is created. (In some cases, stakeholders will have sketched a site navigation map prior to creating the use-case model. In these cases, this early navigation map serves as a useful input to the use-case modeling workshop to ensure that the site offers all the behavior envisioned by its stakeholders.)

The navigation map is a variation of the 'use case storyboard' technique described in the Rational Unified Process activity 'Model the User Interface'. The development of the navigation map starts by identifying the major windows or Web pages for each of the use cases. At this stage we do not know exactly what each screen/page will look like or even exactly what specific screens/pages we will have, so we focus on identifying 'logical pages'. These logical pages are early candidates for the user interface that may be split or merged as the user interface evolves. Logical pages are represented in analysis with the UML construct ***boundary class***. Later in design and implementation, we will represent HTML pages and other visual elements with the UML construct Component.

Once logical pages are identified, the navigation map looks at how a user will navigate from one logical page to another, as well as the major features provided by the logical screen. See figures 2, 3, and 4 for an example.

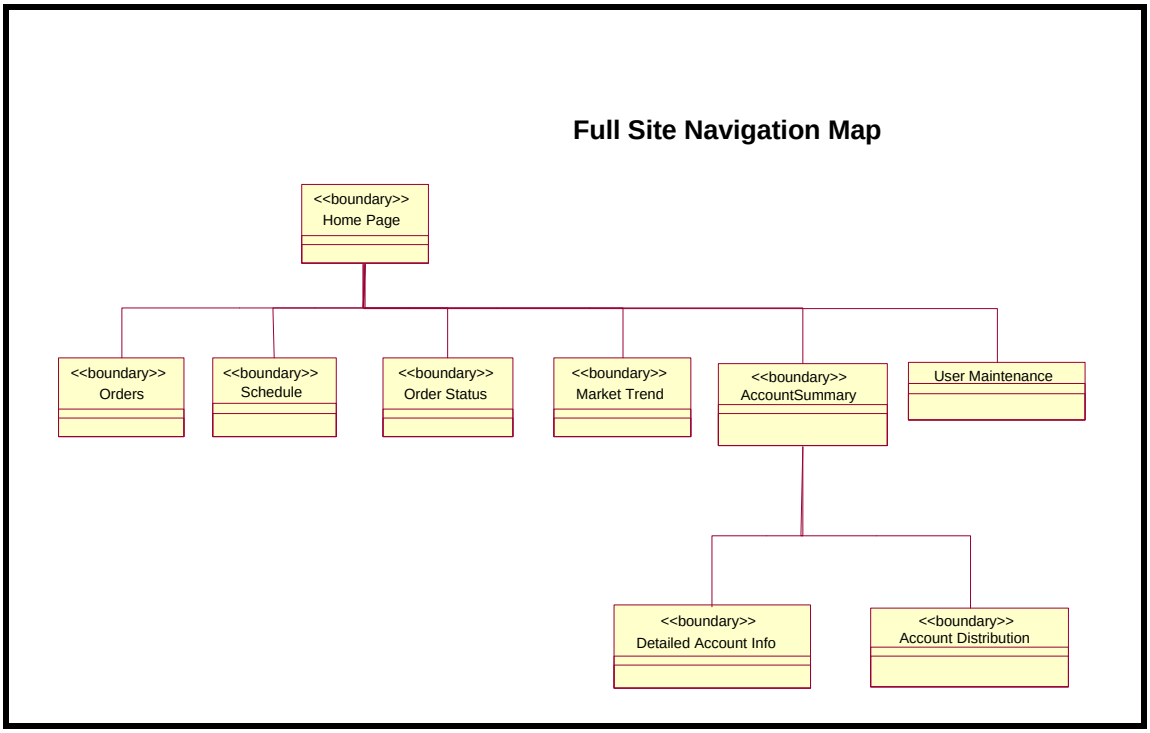

Figure 2: Full Web site navigation map example

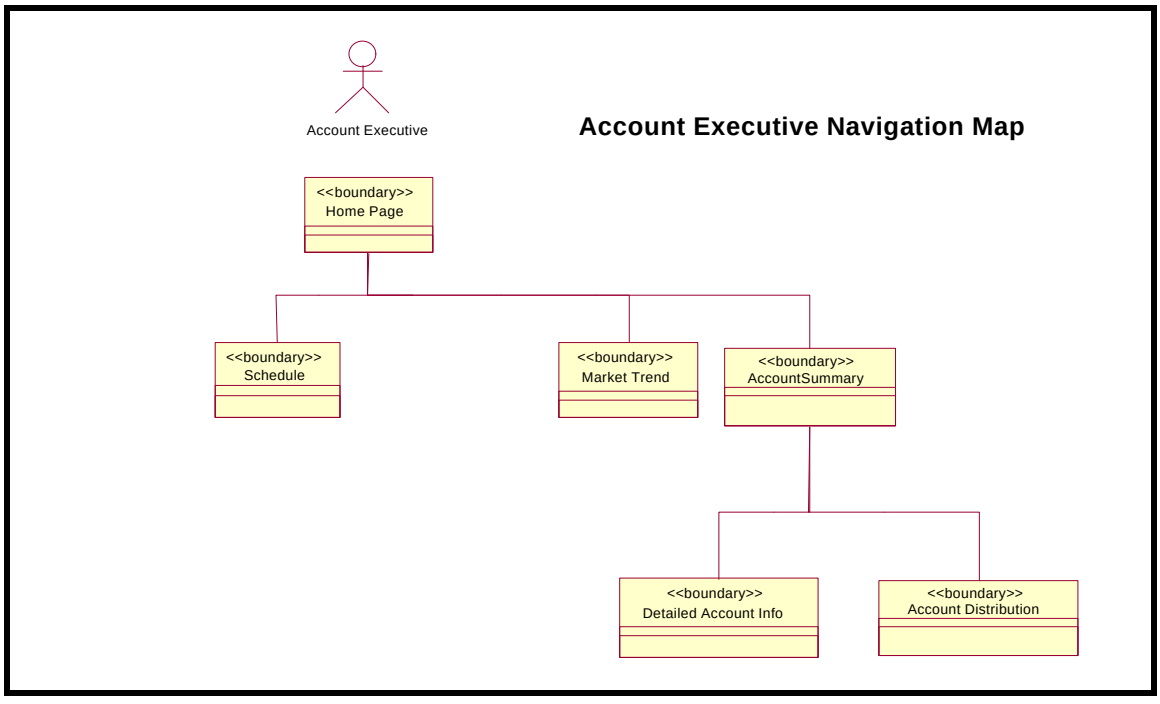

Figure 3: Actor specific navigation map examples

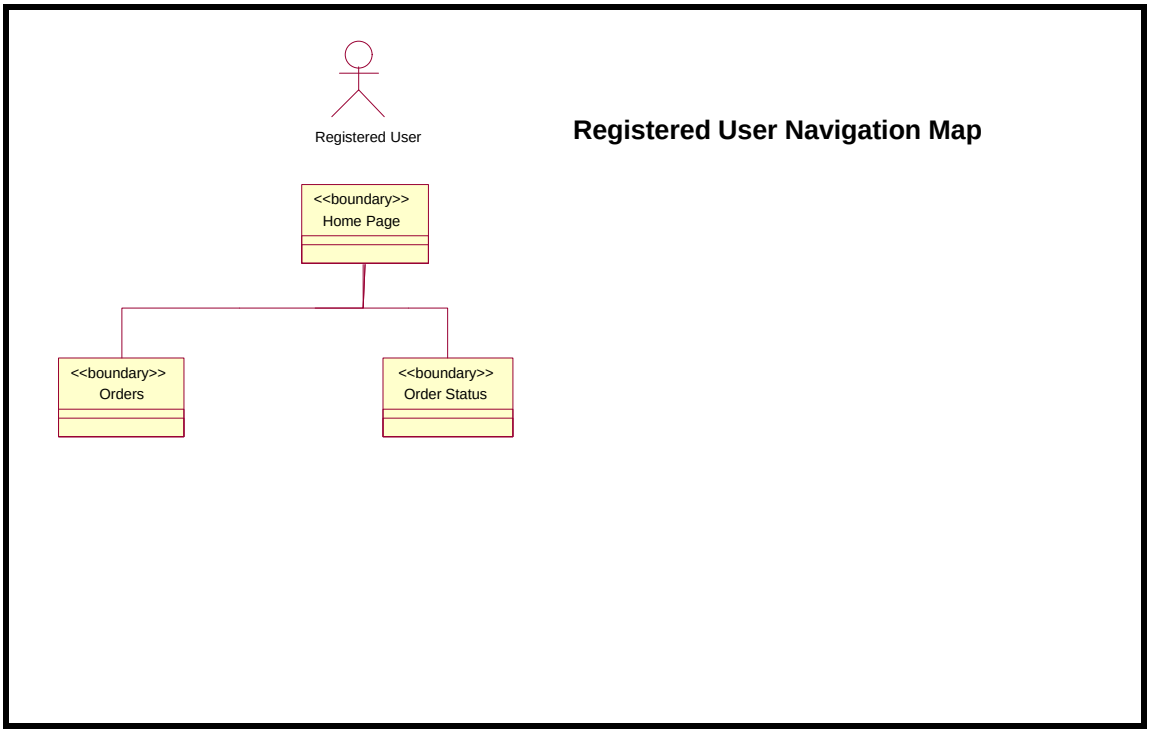

Figure 4: Actor specific navigation map examples

This provides us with a high level view of the navigation map. For a larger system, you typically do one navigation map per actor (see Figure 3 and 4). To drill further into details, the same type of diagram is done for each use case, identifying additional pages (boundary classes) to which the user will navigate while performing the use cases. As the pages (boundary classes) are identified, we also describe the information that they handle.

Creative Design Comps

Creative Design Comps present to the stakeholders a number of visual options for their Web solution in order to "prime the pump"of the creative process of arriving at a truly compelling visual design for the Web solution.

"Comps" (as they are typically abbreviated) consist of mock-ups of what the site might look like. These mock-ups are typically "flat" pictures framed with browser window graphics to give a look of a browser window. The idea behind doing comps is to postpone the investment of the significantly more costly HTML prototypes until there is consensus on the specific graphical direction for the site. The specific non-functional nature of the comps also helps to overcome any misunderstanding that the Web prototype is already complete.

The Rational Unified Process provides an activity, *Prototype the User Interface*, which provides a general approach for gathering feedback on the user interface. For Web development, we extend this slightly by using the concept of creative design comps. To create the comps, we take one of the most important use cases, and develop many alternative designs (e.g., at least 10) for its look and feel. From this set, we choose the three most promising options to present to the stakeholders. It typically takes three iterations with the stakeholders before there is consensus on the final Web design. This is a creative and iterative process.

Once there is agreement and sign-off, the creative design comps evolve into a functional user interface prototype. The non-functional comp is simply the first generation of the prototype, created to solicit feedback for the stakeholders regarding the many visual features of the site.

Web Design Elements

Web Design Elements are the discrete graphical images that are assembled to build the Web pages for a site. Consistency of the user interface across a Web site is essential to usability; the Web site should provide a consistent user experience. To ensure this, the project must consistently use a set of standard graphical components across the site. These components should be developed early on in the project, along with guidelines for their usage, to ensure that all team members understand when and how to use these components.

Examples of Web design elements include graphical elements such as navigational devices and page backgrounds. Reusing high quality standard graphical elements across the entire site ensures consistency, reduces time to market and reduces development cost, as well as increases quality by deploying a smaller set of higher quality components.

The Rational Unified Process identifies components from use cases in "Workflow Detail: Analyze Behavior". These components are then refined, implemented and unit tested in "Workflow Detail: Design Components".

The Web Design Elements are created in parallel with the Initial Web UI Prototype. The Creative Design Comps serve as the input for creating the initial elements used for the UI prototype, and as decisions are finalized on the UI prototype, the Web Design Elements should be completed and signed-off.

Initial Web UI Prototype

The Creative Design Comps evolve into the User Interface (UI) Prototype. The look for the UI prototype is based on the signed-off creative design comp and is constructed in the Rational Unified Process *Activity: Prototype the User Interface*, using the Web design elements identified above. The Initial Web UI Prototype typically supports only portions of the system, based upon the most important and representative use cases.

The development of the Initial Web UI Prototype allows the users and designers to better communicate on both the look-and-feel of the site, as well as the functionality to be delivered. It is essential to get user feedback as early as possible before major investment is done in functionality or UI.

UI Guidelines

When the Initial Web UI Prototype is complete, it is time to develop detailed guidelines for designing the user interface. This style guide will, among other things, specify how and when Web Design Elements shall be used, color schemes, fonts, cascading style sheets, and details on how navigational elements should function and be positioned. The UI Guidelines are defined in the *Activity: Develop User-Interface Guidelines*.

Full Web UI Prototype

The ***Full Web UI Prototype*** expands upon the Initial UI Prototype to cover all use cases. The prototype should now demonstrate full navigation between screens and all visual elements to the site. Real data or dummy data is used depending on the development of the backend functionality of the system.

Although it is expected that the pages developed as part of the Full Web UI Prototype will be refined during each of the construction iterations of the project, the goal of the development of the Full Web UI Prototype is to come to agreement with the stakeholders on the scope and specific nature of each of the user interfaces.

The Full Web UI Prototype is based upon the Rational Unified Process *Guideline: Use-Case Storyboard*.

Full Navigation Map

After completion of the Full Web UI Prototype, the ***Full Navigation Map*** should be created. This map should be based upon the initial site map, as well as the fully defined use-cases for the Web solution. This navigation map should include all known pages/screens identified in the Web UI Prototype.

But it is Not Just Pretty Pictures

One of the primary foundations of the "new economy" is software, specifically, the software of the Internet and the World Wide Web. Many companies, caught up in the gold-rush mentality of the Internet, fall into the trap of thinking that delivering in "Web-time" means developing without sound software engineering practices. Companies building innovative Web solutions need to focus on visually compelling user interfaces, as well as robust architectures. Users of the Web are impatient with poorly designed sites; it only takes a moment to turn a potential customer away forever. While there are many new opportunities presented by the advent of the Internet and the World Wide Web, these solutions are still essentially software systems.

The software engineering lessons learned over the past few decades are clear: a focus on quality, driving toward solutions iteratively while reducing risk, and a focus on flexible architectures are essential to being successful in a rapidly changing world. Failure to approach Web development with the discipline of a sound software engineering approach is a formula for failure. Recent failures of some highly publicized Web sites bear this out.

The following represent the six best practices of software development embodied in the Rational Unified Process, along with how they apply to Web solutions development:

- **Develop Iteratively:**
  Iterative development is based on an approach that relies on continuous discovery, invention, and implementation. It forces the identification of projects risks early in the development life-cycle, making it possible to manage and attack them in a concerted, timely and efficient manner.

  The controlled iterative development practice maps directly to the need of Web applications for shorter and rapid release cycles.

- **Manage Requirements:**
  Requirements management is a systematic approach to eliciting, organizing, communicating, and managing the changing requirements of a software-intensive system or application.

  Given that the Web application requirements can change as frequently as market tastes and trends, tracking their evolution and tracing them through the development life-cycle becomes ever more imperative.

- **Utilize Component Architectures:**
  Architecture describes the structure of the application in terms of components, the way they are integrated, and the fundamental mechanisms and patterns by which they interact.

  Given that Web applications are essentially open and extensible, and can change on an on-going basis, utilization of component architectures becomes vital. Component architectures can scale, and adapt to the ongoing changes, and maximize the reuse of line-of-business or third party component software.

  In some cases, the architectures of Web solutions come right out of the box. Whenever possible and appropriate, buy the architecture. Functionality such as personalization, content management, load balancing, and security are all available in product bundlings. Using products which include these components is a important way to enable your company to deliver in Web-time.

- **Model Visually:**
  Models help us to understand and shape both the problem and the solution. A model is a simplification of reality used to comprehend complex systems.

  Web architectures, being n-tiered and distributed, can be both complex to design, develop and deploy. Managing this level of complexity requires a well thought-out and articulated architecture and design. Visual modeling notations such as UML provide the mechanisms for expressing system architecture and design.

- **Verify Quality:**
  Quality focuses on both process and product. Adhering to an effective process yields both 'quality' interim artifacts and resultant products.

  Web applications are intended for public exposure, and the cost of failure can be painfully high. In cases of poor performance or reliability, customer retention is at risk. Given the release frequency of the Web applications, testing early, often and in automated fashion is crucial.

- **Control Changes:**
  Web applications contain many objects and components that can be created and modified by many people, frequently in parallel. Controlling changes in this continuous development and deployment environment is essential. Simultaneous management of multiple releases and configurations requires a rigorous configuration and change management process throughout the development life-cycle.

Facilitated Workshops

Building Web solutions involves bringing together many different types of stakeholders and disciplines, including marketing, technology, and various intra- and inter-organizational units. For example, a business to business extranet solution typically involves various stakeholders within an organization as well as various stakeholders at those organization who will be customers of the extranet solution. A consumer Internet site will involve stakeholders within an organization including customer service and marketing as well as actual consumers. In order to address this need, the use of facilitated workshop sessions is essential.

Within the Rational Unified Process, we find guidelines for how to conduct Requirements Workshops, Use-Case Workshops and Use-case Analysis Workshops. Context Integration has put these workshops and the creative design process discussed in this paper together into one workshop, the Web Opportunity Workshop (WOW), which is a set of facilitated workshops specialized for Web development. The purpose of the WOW is to give companies a roadmap to their Web solution. This allows companies to develop the initial requirements in a week (in the form of the vision, use-case model, supplementary specification, and glossary), the creative design direction (in the form of the creative brief, navigation map, and creative design comps), as well as the some initial analysis models (object model and database model). Companies consistently find that this approach helps them to quickly realize their Web opportunity, while building their solution using a well conceived and industry proven approach (i.e., the Rational Unified Process).

Conclusion

The Rational Unified Process is a good foundation for Web development as proven through Context Integration's successful delivery of projects for Fortune 1000 companies. By leveraging the existing process framework of the Rational Unified Process, integrating it with the creative design process, and taking a facilitated workshop approach to getting these projects started off on the right foot, Web solutions can be delivered in a successful and predictable manner.

About Context Integration

Context is a leading Web solution integrator: 100% focused on the rapid delivery of innovative, mission-critical, business applications on the Web, guaranteed. Working with best-in-class partners, we offer Web strategy, architecture and planning, creative design, and solutions construction to Fortune 1000 clients.

Context specializes in E-commerce, enterprise relationship management, and Web business intelligence solutions. Founded in 1992, we have extensive experience integrating new business applications on the Web with existing client-server and legacy production systems. Our Web-optimized development methodology and fixed-price/fixed-time project model guarantee results and customer satisfaction.

About Rational Software Corporation

Rational Software Corporation, the leader in unifying software teams, helps organizations develop and deploy e-business, Web, enterprise-wide, technical and embedded software through a combination of tools, services and software engineering best practices.

Rational's solution unifies the key members of a software team – including analysts, developers, and testers – and provides unique offerings optimized for each of these roles, thereby improving team and individual productivity. Rational simplifies the process of acquiring, deploying and supporting a comprehensive development platform, reducing total cost of ownership.

About the Authors

Stan Ward, Chief Methodologist, Context Integration

- Stan Ward currently leads the development of Context Integration's Web optimized methodology, inContext. Stan has over 10 years of experience delivering solutions for Fortune 1000 clients using many software development and facilitated workshop methodologies. (www.context.com)

Per Kroll, Marketing Manager, Rational Software

- Per Kroll has 7 years experience in implementing the Rational Unified Process, and is currently responsible for product direction and marketing of the Rational Unified Process (www.rational.com)

Corporate Headquarters
One Van de Graaff
Suite 104
Burlington, MA 01803
Tel: 781-229-6500
Fax: 781-229-0808
Web: www.context.com

Rational
unifying **software** teams

Corporate Headquarters
18880 Homestead Road
Cupertino, CA 95014
Toll-free: 800-728-1212
Tel: 408-863-9900
Fax: 408-863-4120
E-mail: info@rational.com
Web: www.rational.com

For International Offices: www.rational.com/corpinfo/worldwide/location.jtmpl

TP-175 7/99. Subject to change without notice