

## Bridging Transformation Enterprise - Wide



CITIE  
Cambridge Technology Enterprises

## The Rapid Application Development Process

This paper introduces the Rapid Application Development process for software development and summarizes its advantages and limitations. A description of the goals of the process is given before presenting the various components that make up a RAD project. The team structure is also shown and the roles of each member are described. The RAD phases are defined by specifying their deliverables and the deliverables, in-turn, are detailed with recommendations on their use and content. Examples are given throughout the reading and some diagrams are presented as clarification tools.

**Issam J Zeinoun**

Cambridge Technology Enterprises, Inc.

## The Rapid Application Development Process

### Abstract

This paper introduces the Rapid Application Development process for software development and summarizes its advantages and limitations. A description of the goals of the process is given before presenting the various components that make up a RAD project. The team structure is also shown and the roles of each member are described. The RAD phases are defined by specifying their deliverables and the deliverables, in-turn, are detailed with recommendations on their use and content. Examples are given throughout the reading and some diagrams are presented as clarification tools.

### Introduction

#### The Case for RAD

Rapid Application Development is a software development methodology created to support the construction of business applications for the enterprise in a timely and efficient manner while guaranteeing the usefulness of the end product to the user. As business applications become prevalent, large corporations have come to depend on information systems to increase their productivity and the proficiency of their processes and work models. The holding-back factor for implementing such solutions is the high cost, unpredictable budgets and lengthy delivery times. Additionally, since the use of enterprise wide applications is relatively new, users have difficulty envisioning their needs and the multitude of uses which such systems bring forward. To address these various issues, the RAD methodology is based on a prototype and iterations approach. This means that the user drives the requirements up front and is involved in a hands-on manner throughout the project life cycle.

In general, the RAD process has three main objectives to meet:

- Timely and speedy delivery of a working application (in about 20 weeks)
- Aiding the customer to properly identify the requirements
- Deliver the customer's requirements with accuracy using Prototyping and Iterations

The RAD process works best in cases where the data is known, the requirements can be defined and kept unchanged during the development and the functional requirements can be met within a short time frame with a small team (3-4 months with 5-6 technical resources). For more complex applications with a larger set of requirements, divide the requirements among several RAD projects. Here, the approach is to break down a large and complex solution into a set of smaller and simpler solutions. This has the benefit of delivering solutions in shorter timeframes to demonstrate progress and gain approvals for budgets and resources in an incremental manner.

Not all projects are fit for a RAD implementation. In some cases, the nature of the project does not allow the use of RAD. Where the requirements cannot be locked-down in the first few weeks of the project or where requirements are not available until the later part of the project, RAD cannot be employed. This is also true for research projects where discovery and risk play a big role and are prevalent throughout the project. Furthermore, RAD is not recommended when the requirements push the limits of the technology. If however, all the risks can be mitigated by prototyping the unknown aspects and performing the required discovery up-front, then RAD can be adapted to work by modifying the process to have a longer prototyping and analysis phases – thus prolonging the project timeline but preserving the overall approach.

### Overview of the RAD Phases

The phases are similar to common software development processes but the RAD process emphasizes the prototyping and analysis phases. The listing of the phases in chronological order is as follows:

- **Prototype Phase** – The customer is heavily involved and takes iterations at finalizing the functionality. This phase also includes producing a business case to properly solidify the goals of the application.
- **Analysis and Planning Phase** – Statement Of Work (SOW) contract is created, all approvals and signatures collected. Also, the functionality matrix is constructed.
- **Design Phase** – Includes Technical and functional designs and a traceability matrix is completed.
- **Construction Phase** - Includes at least 3 iterations with a time-buffer for working on incorporating customer feedback.
- **Testing Phase** - Includes QA, iterative bug fixing, and customer feedback for minor improvements.
- **Deployment Phase** - Includes interface to the production data and post deployment testing and final bug fixes.
- **Turn-Over Phase** - Includes training the users of the various roles.

### Overview of The RAD Team

Members of the RAD team must be team players with an urge to learn and to create quality business solutions. They must have a positive attitude and an energetic zeal. Choosing the proper team members is key to the success of the project and if a member is identified to be unsuitable for this fast-paced environment, corrective action must be taken immediately to replace the person. The RAD team consists of the following main roles:

- The Project Sponsor
- The Customer
- The Project Manager (PM)
- The Business Analyst (BA)
- Technical Lead (TL)
- The Developers (2-3 in number)

All the roles are discussed in more detail in the sections to follow and Table 1 highlights the major responsibilities of the various team members throughout the project phases.

**The Rapid Application Development Process**

**Table 1. Responsibilities Overview Throughout Project**

Phase	Sponsor	Customer	PM	BA	TL + Developers
Prototype	View Final Prototype	Define Prototype	NA	Gather Requirements	Translate Requirements to Build Prototype
Analysis and Planning	Signoff on SOW & Plan	Refine Requirements	Plan Project	Lead the Analysis and Refinements	Refine Prototype
Design	NA	Verify System Architecture and Structure	Track Progress	Point of Contact Customer and Developers	Design and Architect Solution
Construction	Iteration Reviews	Iteration Reviews and Feedback	Track Progress & Solve Issues	Test & Define Test Cases	Construct the Solution
Testing	Signoff	Test Application and Log Change Requests	Obtain Signoff	Test & Produce Test Results	Fix Bugs
Deployment	NA	Test Deployed	Mitigate Issues with IT Group and DBA	Test Deployed Application & Guide Customer Use	Deploy and Fix Bugs & Provide User Manual and Installation Documents
Hand-Over	NA	Attend Training	Sign-in all Documents	Provide Training	NA

**Details of The RAD Team**

**The Project Sponsor(s)**

The sponsor is the one or more persons who will define the goals of the project at a high level and communicate them to the customer team and the Project Manager. The Project Sponsor(s) sign(s) the approvals to commence the project such as the Statement Of Work, the budget allocation, customer resource allocation, etc. They communicate with the project manager in the planning and analysis phase for kicking off the project and they participate in the iterations testing and in the final testing of the product to sign-off on the project as done.

**The Customer**

The Customer team must represent all the types of users of the application. For example, if an application will be used by the sales group, and the marketing group and the manufacturing group, then the customer team must include a few users of each group. Additionally, most applications will need an administrator for configurations and maintenance tasks and it would be important to involve at least one user as the administrator early on. The Customer plays a key role in all the phases starting with the functional requirements definition, to iteration reviews throughout the construction, to QA and acceptance testing, to deployment/pilot reviews. Thus the Customer is heavily involved from start to finish and without this participation, the usability and functionality of the end-product will suffer.

## The Rapid Application Development Process

### The Project Manager

Works with the Project Sponsor to refine the goals of the project and its success criteria. The Project Manager (PM) plans the project and allocates the resources with help from the Tech Lead and calculates the budget. The PM communicates with the Project Sponsor for approvals and finalizing the Statement of Work. Thereafter, the PM must track the progress on a daily basis and steer the team in order to meet the timelines. To avert risk, the PM must plan ahead for each phase, track deliverables and create contingency plans to mitigate risks of slipping the schedule. Simultaneously, the manager must control the customer feedback to prevent scope creep but still produce a useful product.

### The Business Analyst

The Business Analyst (BA) helps the users analyze the business problem in order to identify the proper solution. Therefore, the BA helps refine the functional requirements and the screens and flow of the application. Therefore, the BA must properly and accurately learn and comprehend the customer's business processes and culture. The BA is the liaison between the users and the technical lead so all communications concerning the functionality of the end product must be controlled by the BA. Additionally, a BA plays the lead role in producing the Application Definition Document and the Test Cases Document and Test Plan Document.

### The Technical Lead

The Technical Lead (TL or Tech Lead) leads the technical team to produce the prototype and then the final product. The TL is responsible for leading the design and development and integration testing efforts. The TL sets the architecture and the development methodologies and coding standards. He also helps track the progress and performance of the developers and provides mentoring and help when needed. The TL also participates in coding some of the modules and conducting code builds and integration testing.

### The Technical Team

The technical team typically consists of the various expertise including HTML/JavaScript GUI Developer, Java/J2EE Developer(s), and a SQL/Data-Modeling expert. The GUI developer is mostly needed in the first two stages of the project unless the requirements change during the other stages.

The Java/J2EE developer(s) implement the business and presentation code and need to reuse existing similar code when possible. Typically two or more Java developers are needed on a project of small to medium size.

The SQL Expert creates database related designs and code starting from the Data Model to the ER (Entity Relationship) Diagrams and ending with the SQL scripts for bootstrapping the application to Views that provide clean and abstracted interfaces to the data. Additionally, some components can be written in PL/SQL code for batch processing and data manipulations or calculations.

## Phases of a RAD Project

The RAD process can be divided into seven phases as listed in Section 2.2. The phases vary in duration but, in general, the phase's can be measured in percentage to the overall project length. The following is a table of the typical durations of each phase.

Phase	Length in %	Out of 20 Weeks
Prototype	15 %	3 weeks
Analysis and Planning	10 %	2 weeks
Design	15 %	3 weeks
Construction	30 %	6 weeks
Testing	15 %	3 weeks
Deployment	10 %	2 weeks
Hand-Over	5 %	1 weeks

Table 2. Typical Durations of the RAD Project Phases

Typically, the RAD project last between 18 and 20 weeks depending on the complexity of the requirements. Each phase has its well defined goals and deliverables and the completion of the phase is marked by the completion of the assigned deliverables. The deliverables will mostly fall into two categories: documentation, or code; although we do not differentiate between the two in this article. Both deliverable types are equally important and should be tracked on the overall project plan. See Figure 1 for a listing of the deliverables of each stage and the sections that follow for the detailed descriptions. The deliverables can vary among organizations and can be adjusted to fit the variety of cultures and standards thereof; here we give a general structure that is generic in nature.

**The Rapid Application Development Process**

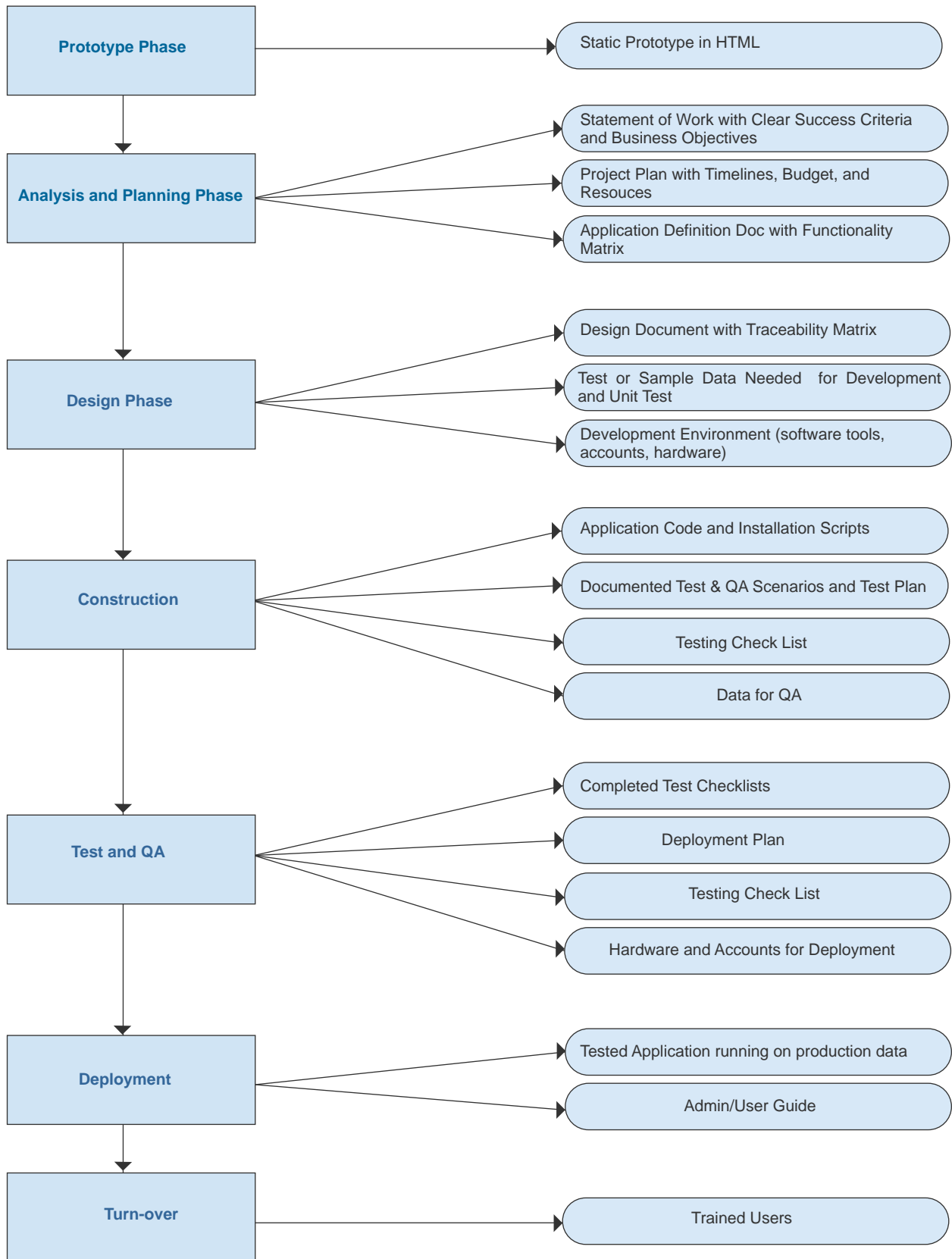


Figure 1. Phase Deliverables

## The Rapid Application Development Process

### Prototype Phase

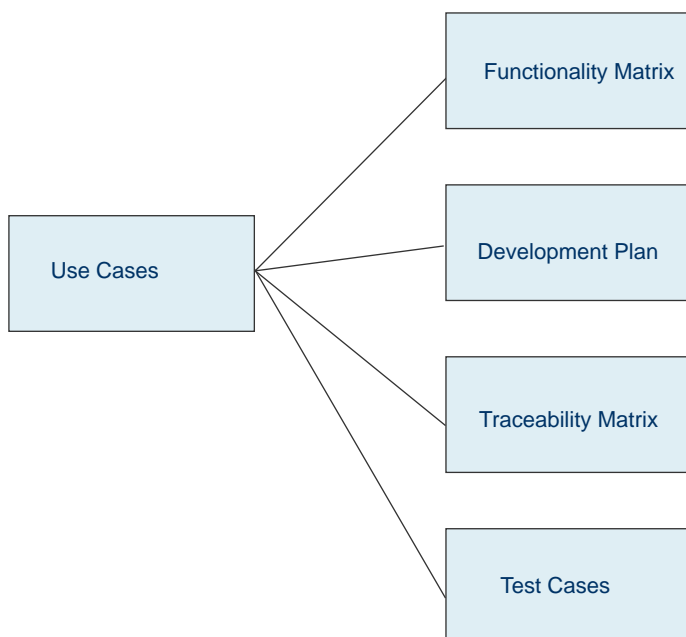
The prototype phase focuses on communicating with the customer in a setting where the players can work together to define the goals of the application and its success criteria. The brainstorming transforms a high level vision issued by upper management and materializes it into clear functional requirements. Typically, the Business Analyst, the Tech Lead, the GUI expert, and the project manager spend a week with the customer during which the initial draft of the requirements is created and results in a static prototype. The customer team at this stage is limited to 5 or 6 users and should be a complete representation of the usertypes.

The prototype is a static representation of the application's functions and features, i.e., it consists of the screens (menus, navigation bars, charts, buttons, tables, graphical structures, etc.) and links between the screens, i.e., the screen flow. The goal is to help and lead the customer to envision the final product and clarify its usability and business benefits. Elaborate technical issues are left for the Analysis phase, i.e., security, network infrastructure, hardware requirements, and other non-functionally related subjects. These should only be briefly discussed as they can impede the progress and present a risk to the timely completion of the prototype.

### Analysis and Planning Phase

The analysis phase includes the in-depth analysis of the prototype and the planning of the rest of the project timelines and scope. The analysis must address both functional and technical requirements and the details must be documented in the Application Definition Document (ADD). The ADD addresses all aspects of the application's functions and features, including technical issues like security levels, accessibility, network structure, connections to external systems. Functional issues include screens, data validation rules, business rules, work flows, etc. The ADD should include the functionality matrix which lists all the use-cases and their corresponding screens. The use-case drive the listing in various documents as shown in Figure 2 so it is important to enumerate the use-cases and use the numbers as keys to link the various documents. Typically, each organization will have a template for the ADD to make sure all the subjects are covered and to keep some consistency among projects and to build upon the lessons learned.

Figure 2. Use Case Listing Drives Various Documents



The customers/users are also heavily involved in this phase to help the BA and the technical team refine the requirements articulated in the prototype into accurate and measurable deliverables that need to be signed-off. Daily meetings with the users are recommended with 2-3 hrs of discussions. These meetings are usually lead by the Business Analyst and Project Manager. All issues should be addressed and the resolutions should be documented. The users must be heard but they must also understand that the scope is to be locked down and finalized during this phase and that meeting the 20 week timeline is essential for the success of the project.

It is also important to notify the users that all requirements that are generated after the completion of the ADD will go through a Change Management Request process which is timely and expensive. It is recommended that a larger team of users be involved at this stage then was involved in the prototyping stage and that their attendance to the meetings be recorded and kept for future reference.

The other goal of this phase is to construct a Statement of Work (SOW) and the project plan. The SOW is the contract that binds the parties to the deliverables. It is a contract between the Project Manager and the Project Sponsor and must be signed off by both sides before any other work can be done. The SOW must state the commitments of the customer as well as that of the developing team. Customers play a major role in the RA D process and their accountability must be clarified up front.

The project plan contains all the milestones and the resource allocations. This will serve as the guide for tracking the remaining phases of the project. The Project Plan starts off high-level based on the phase durations shown in Table 2 – the key feature of the project plan is the time-boxing. Time-boxing means that the project must end within 20 weeks and this drives all subsequent decisions such as scope and number of resources needed. Toward the end of the phase, after the ADD has been finalized, the technical lead can create a detailed plan for the construction of the application and works with the project manager for producing the finalized Project Plan. The detailed construction plan must show all the use-cases of the functionality matrix and must cross-reference the assigned ids of the functionality matrix.

### Design Phase

In the design phase, the technical lead and senior developers delve into the details of the code design and application architecture. All technicalities must be considered from software tools to external interfaces to batch jobs, etc.. The data model and the entity relationship diagram are created along with the class diagrams and sequence diagrams. Also, the screen templates and menu structures for reuses among the various screens are designed. It is important to identify common structures and functions at this stage and design utility classes and manager classes for code reuse, coding efficiency, and code maintainability. Also, include interface definitions between the various screens and modules of the application especially those to be built by different programmers – clear interface definitions make coding easier by reducing ambiguities and confusion among the developers. The result is an Application Design Document which includes a traceability matrix to list all use-cases and functionality as listed in the functionality matrix with the additional information of classes and methods that implement the each functionality. This serves as a check list that all the requirements have been designed into the solution.

The art in creating a well balanced design document lies in specifying enough details to ensure the proper delivery of the requirements and the quality of the code without getting lost in the details and prolonging the phase. The idea is to let the developers make some of the decisions during the construction phase as they work on each module but provide them the framework and guidelines for proper coding techniques such as error handling, logging, database connectivity, and more. The Tech Lead and Senior Developer build these guidelines and verify each other's ideas and approaches and make suggestions – they work closely together to optimize and clarify.

## The Rapid Application Development Process

Additionally, in preparation for the construction phase, the base code tree must be created and loaded onto a source-control system such as PVCS or StarTeam or Microsoft Source Safe. The development environment should be created with scripts for building the code (e.g., ANT build.xml file, data source files, etc.). Decisions on what development tools are to be used must be finalized as well as any other issues that might hinder the construction process.

During this phase, the customer validates high level design and architecture details that affect or are affected by the customer's network and existing systems. Examples of such concerns are batch and background process (since these usually need to be configured and managed), network and server access (e.g. email servers to be used by the application, data-source access, firewalls, etc.), security and encryption (e.g. HTTPS certificates and database password encryption), external system interfaces, etc.. Concurrently, the project manager should drive the customer to deliver the data exports and any other items needed for development in preparation for the construction phase (e.g. database accounts, UNIX/Windows accounts, access to existing reusable components, etc.).

### Construction Phase

The construction starts with each developer reviewing their deliverables as dictated by the project plan and completes when all modules have been built and unit tested. Throughout the construction phase, the technical team members communicate with each other concerning interfaces between their modules since some might not be completely specified in the design document. The Technical Lead must review the code and verify that the coding standards and procedures are adhered to without being too detail oriented as to hinder the progress of the team.

Coding should start with the framework pieces such as the configuration manager, logging manager, etc. and functionally common pieces such as the login module (if applicable) so that other higher level modules can use them. Similarly for screens, such as when working with JSP, the custom tag libraries should first be constructed as well as template JSP, and reusable common JavaScript and style sheets. Also include Data Definition Language (DDL) code for the data model and Data Modification Language (DML) for bootstrapping the application.

In the second week of this phase, the main framework pieces should be in place and a daily build can be commenced. Once a few modules are completed, simple integration testing can be conducted by the Tech Lead to verify the interfaces among the various units. Toward the end of this phase, the Tech Lead can start deploying on a QA box in preparation for the Testing Phase. Other preparation for the next phase must be performed as well. This includes data exports for QA, user accounts of UNIX machines for QA deployment, database usernames for QA testing, an Application Test Document including complete test scenarios for all the use-cases, and a testing check list.

Construction must be accompanied by iteration. Three iterations are recommended where the users join the technical team for a demonstration of what was built to that point. Of course, some links and buttons may not be functional at that point, but regardless, the Tech Lead must be able to successfully demonstrate the features that were planned to be built (according to the project plan). This presents a last chance for the customers to suggest minor enhancements or changes. Small changes are planned for with a buffer in the project plan and should not impact the schedule. On the other hand, major requests such as new functionality must go through a *Change Management Request* process where the impact is assessed in terms of time/materials and communicated in writing to the Project Sponsor. A sign-off is required for such changes to be implemented. Iterations are a key feature of the RAD process and involve both the customer and the technical team.

### Testing and QA Phase

At this stage, it is assumed that all code has been unit tested and basic integration testing has been completed. The Testing Phase focuses on rigorous scripted testing of every functional requirement in the application as listed in the functionality matrix. In the first week of testing, only the technical team is involved and the users are brought in to test after the system has been stabilized and when the major bugs are fixed (i.e., bugs which hamper the testing itself, such as broken links to pages). The success of the testing stage lies in the quality of the test scripts and their comprehensiveness. It is imperative that the business analyst lead the test effort by conducting full tests of all the scenarios as listing in the Application Test Document and that she/he maintain a check list of tests to make sure all test are covered and date the tests.

A bug tracking system is needed so that the Business Analyst can log bugs and move on with the test without getting slowed-down by having to personally communicate the bugs to the developers. As the BA logs bugs, the developers check for bugs assigned to them and fix them accordingly as fast as possible. This process is iterative so that when ever a bug is fixed, it is marked as "fixed" by the developer, and becomes the responsibility of the tester (typically the BA) to verify the fix and either close it or re-open it.

Once the application is in a reasonable state, the customers/users can help with the testing. Involvement of the users must be properly managed because any of the following things could happen:

- If users find too many bugs they get a negative impression of the product
- If users are not guided and trained to use the product they will feel confused and be unproductive.
- If users start discussing the screens and functionality, they could delve into a new "requirements gathering session" and turn the testing phase into the planning phase

To avert these problems from occurring, the BA must lead the testing should kick off the first session with a statement of the objectives of testing and leading the testing by projecting the application onto an overhead screen and marching (clicking) through the test scenarios with the users.

User testing can be done over a period of two weeks where the users meet every other day to log bugs and verify that previously logged bugs have been fixed. They can log bugs on a "Bug Sheet" prepared by the BA that guides them in writing down the appropriate and needed information on how the bug occurred and how to reproduce it – the BA will, later on, enter them into the bug tracking system.

Testing the application takes place on the QA server, i.e., the users point their browsers to the QA URL as provided by the system manager. The Tech Lead deploys the application on a daily basis from Development to QA so that all the bug fixes are transferred for verifications. At this stage, it is essential to separate the development and QA systems to avoid confusion and duplication of efforts. The deployment process is shown in Figure 3 where the separation of the systems is emphasized. An example of the need for separate systems is to avoid having the programmers change the data while the testers change the same data; both parties will be confused and will not be able to come to any useful conclusions since they do not have control over the data.

All testers whether BA, programmers, or users, must track the test scenarios they complete and date their completion on the Testing Check List. The Project Manager reviews these lists to certify the completion of the test phase and validate that sufficient testing has been conducted and that it is safe to move into production and introduce new users.

## The Rapid Application Development Process

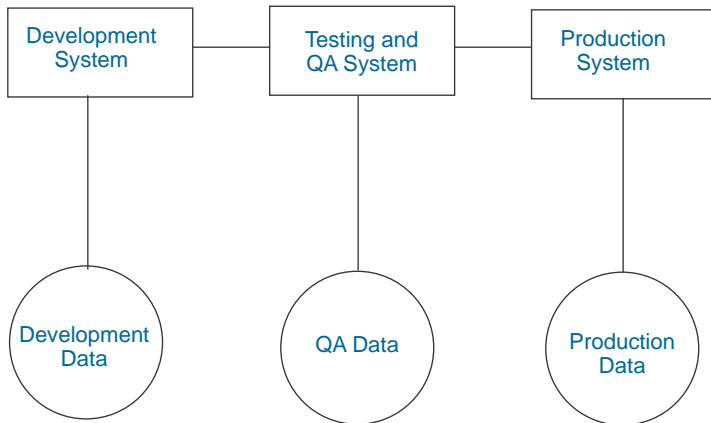


Figure 3. QA Deployment Steps

Deployment on production should not include the source code, it should be limited to copying the built code from the QA machine. Since the database is different on the production system, some configuration differences exist between the QA system and Production system. For example, port numbers, directory structures, Hostname/IP -address, and other server related items differ. These difference should be clearly documented in a deployment/installation guide which should clearly indicate the changes needed and the best way to achieve them.

Invariably, it has been found that production data differs from QA data and minor code changes will be needed after testing the production system to compensate for corresponding issues. In some cases, major bugs are found after deployment, either because the data on the QA system was not sufficient for complete testing, or due to differences in data formats and data integrity issues in production. These issues risk the project slipping so two weeks are allocated for deployment, testing, and fixes thereafter. Furthermore, it is recommended that a maintenance contract be construed at this stage so that enhancements and other change requests can be handled after project completion – some requests might be essential and could further encourage usage of the newly introduced application.

### Phased-Deployment with a Pilot

It is preferable to introduce the application to a subset of users (varying between 10% and 20% of the final user base). These pilot users may provide another level of feedback which might prove essential to the success of the application with the complete user base. This provides a final chance to identify and fix critical issues or implementing essential enhancements. Furthermore, a phased approach can start with 20% of the user base then move to 50% then 75% then 100%. This helps test the scalability and performance of the application in its final form and setup and verifies the underlying hardware and network infrastructure.

### Turn-Over Phase

Once the application is running on the production server and has been stabilized, the users can be trained to use it properly and efficiently and the administrator for the application can be trained on the various maintenance aspects. This can be done in a week or so of organized classes. Note that there should be a good number of users who are familiar with the application at this points since they partook in the various phases and had the chance of using the application extensively in the Testing Phase. This phase might require a training guide or can use the testing document for guiding the users or the user guide if it is made available in the previous stage.

### Bold moves to drive the effort

The commanders for the Navy Personnel Command and Naval Personnel Development Command designed a process to support the creation and deployment of the technology-based solution. In the spirit of Sea Enterprise, the commanders drew from the best practices in Corporate America.\*

\*Kotter, John P: Leading Change: Why Transformation Efforts Fail. Harvard Business Review, March-April 1995

- ➔ **Establish a sense of urgency.** The overarching force transformation, as well as the Navy transformation, is progressing well, which creates an external impetus to move quickly with Sea Warrior. More importantly, however, the commanders created internal urgency and immediate pressure by fervently advocating SeaWarrior across the Navy. They established a set of internal deadlines, such as a brief to the Chief of Naval Operations, to create a performance thrust.

In preparation for the next phase, the usernames and accounts for the database and production machines must be created and proper access be granted to the supporting structures and objects. The Project Manager ensures this by working with the customer's system team and database administrator. A deployment plan is completed as well to clarify all security, accessibility, firewall workarounds or limitations, external system interface connectivity, etc.

Note that during the QA phase, new requests for enhancements will come up because the customer are actually using the application now instead of being presented with screens and ideas. These suggestions should be assessed and implemented when possible.

### Deployment Phase

#### Overview

Deploying on the production environment will typically exposed new bugs due to the differences in the data and privileges between QA and Production. For example, even if the data is identical, sometimes the access rights/restrictions of the database username on the production database will produce problems. These should be minor problems and can be fixed in a quick and simple manner by the DBA. For this reason, we recommend a week of testing the application (the BA using the Test Document) on production before opening up the system to the users.

Create a User Guide and an Installation/Administration document before the completion of this stage. The User Guide should contain directions for the various use-cases and clarifications of business rules that are not apparent in the screens and menus. Depending of the complexity of the application, the User Guide can contain the Installation/Administration information or it can be separated into a separate document. This document should cover installation and configuration and administration of the application and on using the administration screens and command line tools (where applicable). The level of detail of such a document depends on the combination of application complexity, sophistication of the user-base, timeline for delivery and, resource availability. The Project Manager and Technical Lead must decide accordingly.

## The Rapid Application Development Process

- ➔ **Form a powerful guiding coalition.** The commanders have created a broad front to support the effort, including influencers in the enlisted ranks, policymakers, users, etc. A set of offsite workshops to engage these influencers in defining the policies, IT solutions, benefit analysis, and deployment roadmaps, etc, at a very concrete level was a key to success. The academic environment for these workshops has enabled free exchange of ideas and give-and-take atmosphere that are atypical for the Navy but very beneficial for quickly converging on a shared solution.
- ➔ **Create and communicate a vision.** The commanders have used multiple tools to illustrate their goal for Sea Warrior in very practical terms. They have "packaged" the vision in a compelling and comprehensive manner by including:
  - ➔ Role-plays to highlight the issues with the current approach (e.g., nuclear submarine captain playing an agitated wife of an enlisted sailor who has been "slammed" to Japan for two years).
  - ➔ Rapidly prototyped screens for IT solutions to concretize how the assignment marketplace could look like to a sailor or a detailer, for example.
  - ➔ Comprehensive business case analyses to illustrate - in a quantitative fashion - the positive impact of each element of the Sea Warrior effort.
  - ➔ Detailed implementation roadmaps with key milestones and cost estimates to simplify decision-making and speed up task assignments.
- ➔ **Empower others to act on the vision, and plan for and create short-term wins.** The commanders broke the grand Sea Warrior vision into four distinct streams of work. Each of these streams of work is contained (but coordinated in taxonomy, IT architecture, etc.) and is independently beneficial. At the end, the streams of work will converge to complete the Sea Warrior effort.
  - ➔ The first stream of work codifies, structures, and links the tasks, skills, and training into a consistent taxonomy and data structures.
  - ➔ The second stream of work creates the transparent assignment marketplace.
  - ➔ The third stream of work incorporates training and feedback variables in the assignment marketplace.
  - ➔ The fourth stream of work enables the linkage between missions and the assignment marketplace.
- ➔ **Consolidate improvements and produce still more change - institutionalize new approaches.** The separate Sea Warrior streams of work are solidly under way, and the feedback has been very positive. The converged Sea Warrior will be operational by the end of 2004. However, changing the manners of a very large organization will require several years. To ensure that the organization won't lapse back, the commanders have instituted the new Sea Warrior approaches into policies and structures. For example, they have secured the approval of the Congress for the incentive pay for job auctions.

### Conclusion

In summary, the RAD process has proven to be successful when properly implemented on the appropriate type of projects. This paper discussed the benefits and limitations of the process as well as giving a synopsis of the team and phases that are needed to complete a RAD project. Various references are available on the Internet and many books have been published that give more details on each phase and its deliverables and can be referenced accordingly.

The latter three streams of work were started with an offsite workshop to detail its contribution to the Sea Warrior vision and to rapidly (in days) create a prototype and key requirements for the IT solution. The workshops were followed by a 15-week pilot deployment for a selected homogenous user group. Although the pilot solutions lacked features, the short time frames were essential in making the approach very tangible and in creating concrete short-term wins to build momentum. The pilots enabled broadening the visibility and support for the Sea Warrior effort within the ranks. Additionally, the pilots provided valuable input in shaping the rollout of the solution to a broader user group.