Chapter 1: Introduction to Rapid Application Development (RAD)

1. Introductions

RAD refers to a development life cycle designed to give much faster development and higher quality systems than the traditional life cycle. It is designed to take advantage of powerful development software like CASE tools, prototyping tools and code generators. The key objectives of RAD are: High Speed, High Quality and Low Cost.

RAD is a people-centered and incremental development approach. Active user involvement, as well as collaboration and co-operation between all stakeholders are imperative. Testing is integrated throughout the development life cycle so that the system is tested and reviewed by both developers and users incrementally.

Rapid Application Development as "an approach to building computer systems which combines Computer-Assisted Software Engineering (CASE) tools and techniques, user-driven prototyping, and stringent project delivery time limits into a potent, tested, reliable formula for top-notch quality and productivity. RAD drastically raises the quality of finished systems while reducing the time it takes to build them."

Online Knowledge defines Rapid Application Development as "a methodology that enables organizations to develop strategically important systems faster while reducing development costs and maintaining quality. This is achieved by using a series of proven application development techniques, within a well-defined methodology."

In short, Rapid Application Development is exactly that. It is a process through which the development cycle of an application is expedited. Rapid Application Development thus enables quality products to be developed faster, saving valuable resources.

2. Why do you need to be RAD?

RAD takes advantage of automated tools and techniques to restructure the process of building information systems. This new process, extrapolated to the entire IS organization, results in a profound transformation of information systems development. RAD replaces hand-design and coding processes, which are dependent upon the skills of isolated individuals, with automated design and coding, which is an inherently more stable process. RAD may thus give an IS organization its first real basis for continuous improvement. In addition to being more stable, Rapid Application Development is a more capable process, as it is much faster and less error prone than hand coding.

Most organizations are faced with a large backlog of new systems to be developed. Over 65% of the typical Information System's budget is spent on the maintenance of existing systems. These systems have little documentation and were developed with programming languages and database systems that are difficult and time consuming to change. These organizations are thus faced with upgrading their aging systems or building new applications. Traditional development lifecycles, however, are too slow and rigid to meet the business demands of today's economy. A new methodology must be implemented, one that allows organizations to build software applications faster, better, and cheaper. RAD enables such development.

The availability of powerful CASE software makes it possible for developers to create systems much faster than ever before. These new integrated CASE toolsets are breaking out of the bubble of traditional software development thought. They take application development beyond generation coding, just as generation, many years ago, surpassed textual coding. These tools enable a developer to drag-and-drop previously generated code, saving that developer the time and effort of individually hand-coding the text of the application. CASE tools also enable a developer to implement Rapid Application Development irrespective of their programming language or platform.

There are only two things of importance. One is the customer, and the other is the product. If you take care of customers, they come back. If you take care of the product, it doesn't come back. It's just that simple. And it's just that difficult. Rapid Application Development, in addition to providing a more quality product in less time, also ensures greater customer satisfaction. By reducing the elapsed time between User Design and Cutover, RAD increases the likelihood that the system will be satisfactory to the users, whose demands are met much quicker than ever before. The RAD process also directly integrates the end-users in the development of the application. Iterative prototyping mandates that the development teams concentrate on delivering a series of fully functional prototypes to designated user experts. Each prototype is tested by those users and returned to the development team for reworking, at which point the cycle repeats. The series of prototypes thus evolves into the final product, giving the users the opportunity to fine-tune the requirements and review the resulting software implementation.

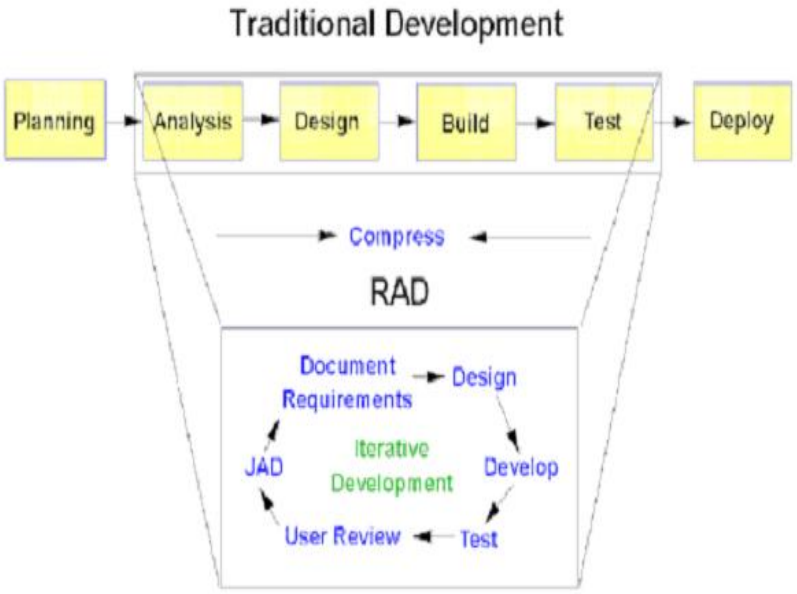
3. The history of RAD

Traditional lifecycles devised in the 1970s, and still widely used today, are based upon a structured step-by-step approach to developing systems. This rigid sequence of steps forces a user to "sign-off" after the completion of each specification before development can proceed to the next step. The requirements and design are then frozen and the system is coded, tested, and implemented. With such conventional methods, there is a long delay before the customer gets to see any results and the development process can take so long that the customer's business could fundamentally change before the system is even ready for use.

In response to these rigid, cascading, one-way steps of Stage wise or Waterfall Models of development, Barry Boehm, Chief SW Engineer at TRW, introduced his Spiral Model. The Spiral Model is a risk-driven, as opposed to code-driven, approach that uses process modeling rather than methodology phases. Through his model, Boehm first implemented software prototyping as a way of reducing risk. The development process of the Spiral Model separates the product into critical parts or levels while performing risk analyses, prototyping, and the same steps at each of these levels. Similarly, Tom Gilb's Evolutionary Life Cycle is based on an

evolutionary prototyping rationale where the prototype is grown and refined into the final product.

RAD compresses the step-by-step development of conventional methods into an iterative process. The RAD approach thus includes developing and refining the data models, process models, and prototype in parallel using an iterative process. User requirements are refined, a solution is designed, the solution is prototyped, the prototype is reviewed, user input is provided, and the process begins again.



4. Essential Aspects of RAD

Rapid Application Development has four essential aspects:

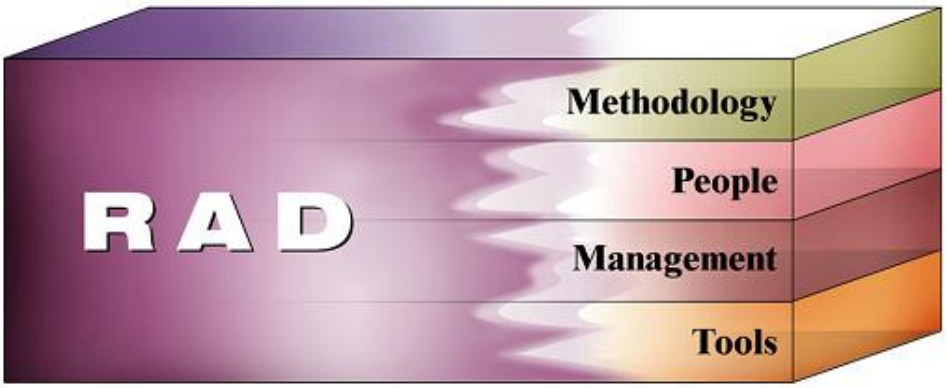
Methodology

People

Management

Tools

If any one of these ingredients is inadequate, development will not be high speed. Development lifecycles, which weave these ingredients together as effectively as possible, are of the utmost importance.



4.1 Methodology

The challenges facing software development organizations can be summarized as more, better, and faster. The RAD development path attacks these challenges head-on by providing a means for developing systems faster, while reducing cost and increasing quality. Fundamentals of the RAD methodology thus include:

- Combining the best available techniques and specifying the sequence of tasks that will make those techniques most effective.
- Using evolutionary prototypes that are eventually transformed into the final product.
- Using workshops, instead of interviews, to gather requirements and review design.
- Selecting a set of CASE tools to support modeling, prototyping and code reusability, as well as automating many of the combinations of techniques.
- Implementing time boxed development that allows development teams to quickly build the core of the system and implement refinements in subsequent releases.
- Providing guidelines for success and describing pitfalls to avoid.

Active user involvement throughout the RAD lifecycle ensures that business requirements and user expectations are clearly understood. RAD takes advantage of powerful application development tools to develop high quality applications rapidly. Prototyping is used to help users visualize and request changes to the system as it is being built, allowing applications to evolve iteratively. RAD techniques are also very successful when faced with unstable business requirements or when developing non-traditional systems.

The structure of the RAD lifecycle is thus designed to ensure that developers build the systems that the users really need. This lifecycle, through the following four stages, includes all of the activities and tasks required to scope and define business requirements and design, develop, and implement the application system that supports those requirements.

Requirements Planning

Also known as the Concept Definition Stage, this stage defines the business functions and data subject areas that the system will support and determines the system's scope.

User Design

Also known as the Functional Design Stage, this stage uses workshops to model the system's data and processes and to build a working prototype of critical system components.

Construction

Also known as the Development Stage, this stage completes the construction of the physical application system, builds the conversion system, and develops user aids and implementation work plans.

Implementation

Also known as the Deployment Stage, this stage includes final user testing and training, data conversion, and the implementation of the application system.

## 4.2 People

The success of Rapid Application Development is contingent upon the involvement of people with the right skills and talents. Excellent tools are essential to fast application development, but they do not, by themselves, guarantee success. Fast development relies equally heavily on the people involved. These people must thus be carefully selected, highly trained, and highly motivated. They must be able to use the tools and work together in close-knit teams. Rapid development usually allows each person involved to play several different roles, so a RAD project mandates a great degree of cooperative effort among a relatively small group of people.

Each stage of a rapid development project includes activities that need to move fast. As a result, it is critical that management initiates the project quickly, cutting through any political delays or red tape. At the Requirements Planning and User Design stages, key end users must be available to participate in workshops. While the system is being constructed, the Construction Team, which uses the CASE toolset to accomplish detailed design and code generation, must be

poised to move quickly. At the end of the development cycle, the Cutover Team, which handles training and cutover, must also be ready to move quickly.

The key players in a Rapid Application Development project include:

**Sponsor**

A high-level user executive who funds the system and is dedicated to both the value of the new system and to achieving results quickly.

**User Coordinator**

A user appointed by the Sponsor to oversee the project from the user perspective.

**Requirements Planning Team**

A team of high-level users who participate in the Joint Requirements Planning workshop.

**User Design Team**

A team of users who participate in the design workshop. This team should be comprised of both high-level users from the Planning Team and lower-level users with a more detailed knowledge of the system.

**User Review Board**

A team of users who review the system after construction and decide whether modifications are necessary before cutover.

**Training Manager**

The person responsible for training users to work with the new system.

**Project Manager**

The person who oversees the development effort.

**Construction (SWAT) Team**

The SWAT (Skilled Workers with Advanced Tools) Team is a small team of two to six developers who are highly trained to work together at high speed. To achieve the fastest possible development, the team members must be highly skilled in the RAD methodology and in using the chosen CASE toolset.

**Workshop Leader**

The specialist who organizes and conducts the workshops for Joint Requirements Planning and Joint Application Design.

4.3 Management

Achieving high-speed development is a complex process. Systems will not be developed and deployed rapidly if bureaucracy and political obstacles stand in the way or if users are not appropriately involved. Management must be totally committed to RAD in order to manage the change in culture. They must be prepared to motivate both users and IT staff, selects and manage SWAT teams, and demonstrate through the use of performance measurements that RAD does mean speed, quality, and productivity. Good management and dedication to the ideals of Rapid Application Development are thus essential to faster system building.

To successfully introduce rapid development, management must pay careful attention to human motivation. Managers should target those professionals whom they deem as 'Early Adapters.' 'Early Adapters' are those people who see the value of a new methodology and lead the way in making it practical to use. These employees are enthusiastic about the new methodology and they want to make it work well in their environment. Similarly, managers must be aware of the type of motivation that is most effective for each individual employee, whether it be money, pride, prestige, excitement, or some combination thereof.

Because Rapid Application Development is such a sweeping change from the conventional development methods, the best way for a manager to introduce new rapid development techniques is to start small. Original Construction Teams of two to four people should be established and their members should be thoroughly trained in the use of the tools and techniques. As these teams gain experience, they will be able to fine-tune the development lifecycle to improve its effectiveness in their environment. Underlying all of this progress,

however, managers must remember the importance of comprehensive and quality training in the use of tools. Good training with tools that are exciting to use can have a profound impact on the attitude of IT professionals, as well as ensure the uninterrupted success of the rapid development project.

### 4.4 Tools

The RAD methodology uses both computerized tools and human techniques to achieve the goals of high-speed and high quality. The above has been primarily concerned with the goals of Rapid Application Development and the role of organizations and the people within those organizations in the achievement of those goals. The success of any Rapid Application Development project is primarily dependent, however, upon the tools used.

Examples of tools that can be used in RAD projects are CASE tools. These tools play a pivotal role in eradicating some problems that exist in other models of software development. For example, CASE tools can be used to develop models(using eg UML diagrams) and directly generate code based on those models instead of hard coding.

## 5. Conclusion

The conclusions emanating from the findings of this deliberation are suggestive of the fact that Rapid Application Development has brought about a new dimension in the software system development. The main points to be noted include the following:

- RAD has successfully achieved the objective of reducing costs on project whilst not compromising on quality by effective reducing the project time-frame and the number of people involved in such project.
- It has also been successful in encouraging the involvement of customers in the entire process of its development lifecycle. This proves advantages in many respects but most importantly this improves the development process by ensuring full acceptance from the customer whilst the system is still being created.
- RAD has also demonstrated strength in being able to speed up the development process by appropriately fusing its methodology, people, management and high tech computer aided tools.
- RAD has also proven to have challenges. Amongst these challenges are the fact that it tends to learn too much on emphasizing more on delivery deadline and then compromising on other features that could have been added if there was not deadline set.

Having said all that, it can be noted that the advantages of RAD outweigh the disadvantages and hence that means RAD has successfully met its objective to create quality systems, faster and at minimum cost.