Rapid Application Development
CSC2213 - Lecture 2

Binuri Raigamkorale
binuriyr@dcs.ruh.ac.lk

Today’s Outline

1. Introduction to RAD.
2. RAD Model phases.
3. History of RAD
4. Four dimensions of Rapid development
5. Why RAD
6. Companies using RAD
7. RAD Vs Agile.

Learning Objectives

- To introduce the RAD concepts
- To give an overview of the RAD methodology
- To introduce the essential RAD techniques
- Describe the systems development components essential to RAD
- Discuss the conceptual pillars that support the RAD approach.
- Explain the Rapid Application Development (RAD) approach and how it differs from traditional approaches.

Introduction to RAD

- RAD is another software development life cycle, designed .

- Compare to traditional life cycle RAD is **faster** development with **higher quality** systems, **saves valuable resources**.

- The key objectives of RAD are:
  High Speed
  High Quality
  Low Cost

- It is designed to take advantage of powerful development software like:
  CASE tools **(**computer-aided software engineering)
  Prototyping tools
  Code generators

- RAD is a people-centered and incremental development approach.
  - active user involvement
  - collaboration and co-operation between all stakeholders
  - system is tested & reviewed by both developers & users

Introduction to RAD

- In certain situations,

  a **usable 80% solution** can be produced in **20% of the time** that would have been required to produce a total solution.

  the **business requirements** for a system can be fully satisfied even if some of its **operational requirements** are not satisfied.

  the acceptability of a system can be assessed against the agreed minimum useful set of requirements rather than all requirements.

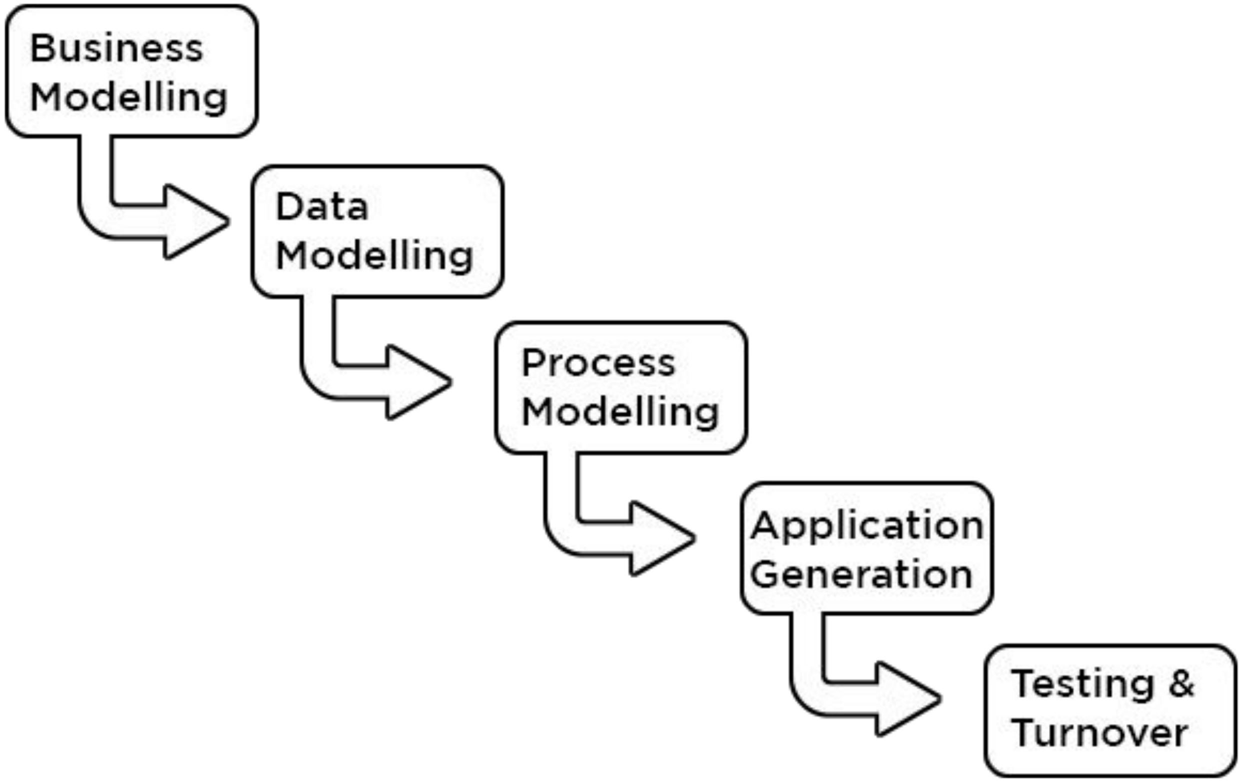
Introduction to RAD ..

RAD is based on followings:

- Reusable code in an accessible repository
- Quick prototyping
- clients’ feedback
- Building usable software as quickly as possible
- The client and developer work together to create an end product that functions exactly as the client wants.

RAD Model phases

There are five phases.



RAD Model phases ..

1. **Business modeling (**Requirements gathering and analyzing **)**

In this phase, **business functions and product scope** are decided during various meetings between the requirements planning team and the client team.

All the relevant members (managers, IT staff, users, etc.) plan and agree on the project’s needs, scope, challenges, and requirements.

The information flow among business functions is defined by answering questions like
  what information drives the business process,
  what information is generated,
  who generates it,
  where generates it,
  where does the information go ,
  who process it and so on.

RAD Model phases ..

2. Data modelling

In the data modeling phase, all the information derived in the business modeling phase is analyzed and separated into different data elements (**objects**, **entities**, **and their relationships**) important for the business.

A relationship between these objects and their usefulness as defined in the Business Modeling step is also established during this phase of the RAD model.

3. **Process modelling**

In this phase, all the data objects gathered in the process modeling phase are transformed into required useful information.

During this stage, changes and optimizations can be done, and the sets of data can be further defined. Any descriptions for adding, removing, or changing the data objects are also created during this phase.

RAD Model phases …

4. Application modelling

The Application Generation step is when all the information gathered is coded, and the system that is going to be used to create the prototype is built using different automated CASE tools.

The data models created are turned into actual prototypes that can be tested in the next step.

5. Testing and turnover

The Testing and Turnover stage allows for reduced time in the overall testing of the prototypes created. Every model is tested separately to identify and adapt the components quickly to create the most effective product. Since most of
the elements have already been examined previously, there should not be any major problems with your prototype.

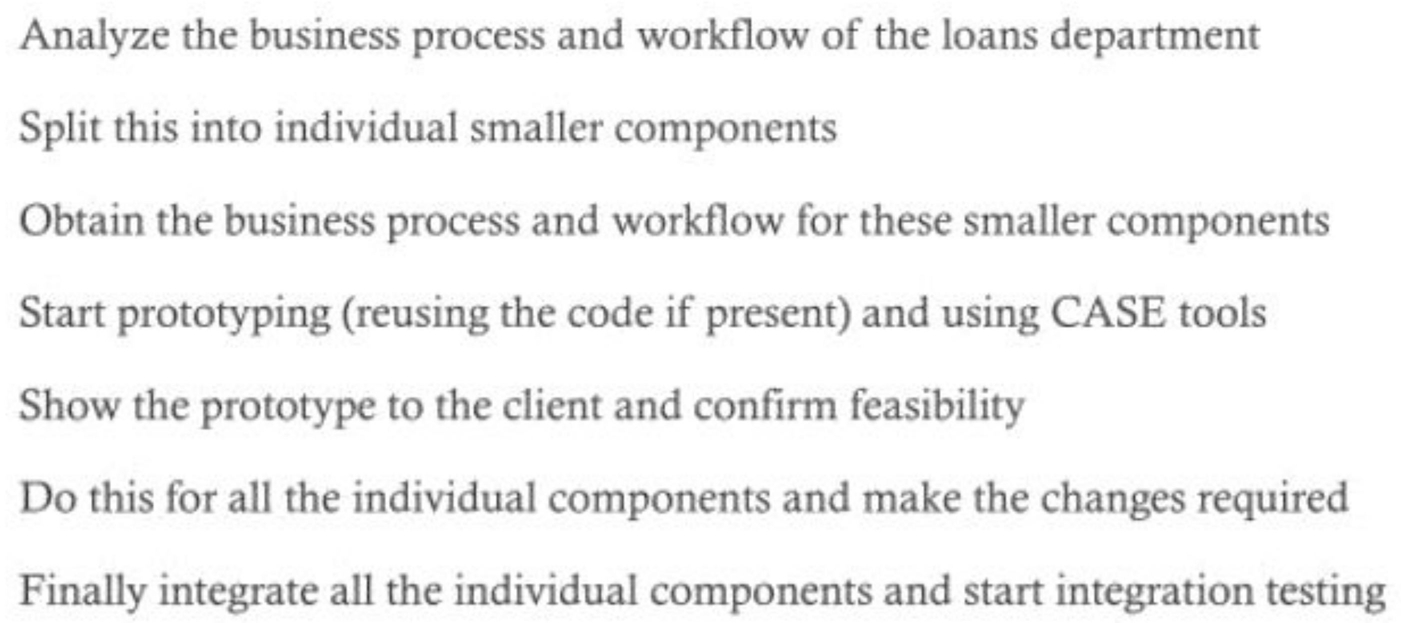
How to Apply the RAD Model

Where a RAD model is successful:

- When the system can be modularized and then distributed in a divided form
- When there are many designers available for the modeling
- When there is money in the budget for using automated code generating tools;
- When there is an expert available to decide which model should be used (RAD or SDLC);
- When a prototype is expected by the customer within two to three months;
- When changes to the product are planned throughout the development process.

How to Apply the RAD Model

Consider example:
Loan department in a bank.



Analyze the business process and workflow of the loans department

Split this into individual smaller components

Obtain the business process and workflow for these smaller components

Start prototyping (reusing the code if present) and using CASE tools

Show the prototype to the client and confirm feasibility

Do this for all the individual components and make the changes required

Finally integrate all the individual components and start integration testing

How to Apply the RAD Model - Ex. explained

Loan department in a bank

Business modelling:

- Identify the loan department's business goals, such as increasing loan approval efficiency and reducing manual paperwork.
- Determine the key activities involved; loan application submission, credit evaluation, and loan approval.
- Analyze the existing loan processes and identify where to apply enhancements, such as automating document verification or integrating credit scoring systems

Data Modelling

- Identify the key entities, such as customer information, loan applications/details, and loan approval status.
- Define the attributes for each entity, such as customer name, address, income details, loan amount, loan rate, and approval status.
- Determine the relationships between entities, such as a one-to-many relationship between customers and loan applications.

Process Modelling

model the loan origination and approval process flow. For example:

- Application Submission: Customer submits a loan application with required documents.
- Document Verification: Bank staff verifies the submitted documents for accuracy and completeness.
- Credit Evaluation: Bank system assesses the customer's creditworthiness based on credit score, income verification, and other factors.
- loan approval: loan offer reviews the credit evaluation results and approves or reject the loan application.
- funds are transferred into customer’s account.

How to Apply the RAD Model - Ex. explained

Loan department in a bank - continued…

Application modelling:

- Design the loan management system's user interface and functionality. For example:
    - User Interface: Develop a user-friendly web interface where customers can easily submit loan applications and track their status.
    - Credit Scoring Integration: Integrate a credit scoring system that automatically evaluates the creditworthiness of applicants based on predefined criteria.
    - Document Upload: Enable customers to securely upload and submit necessary loan documents directly through the system.
    - Approval Workflow: Implement an automated workflow that routes loan applications to the appropriate loan officers for review and approval.

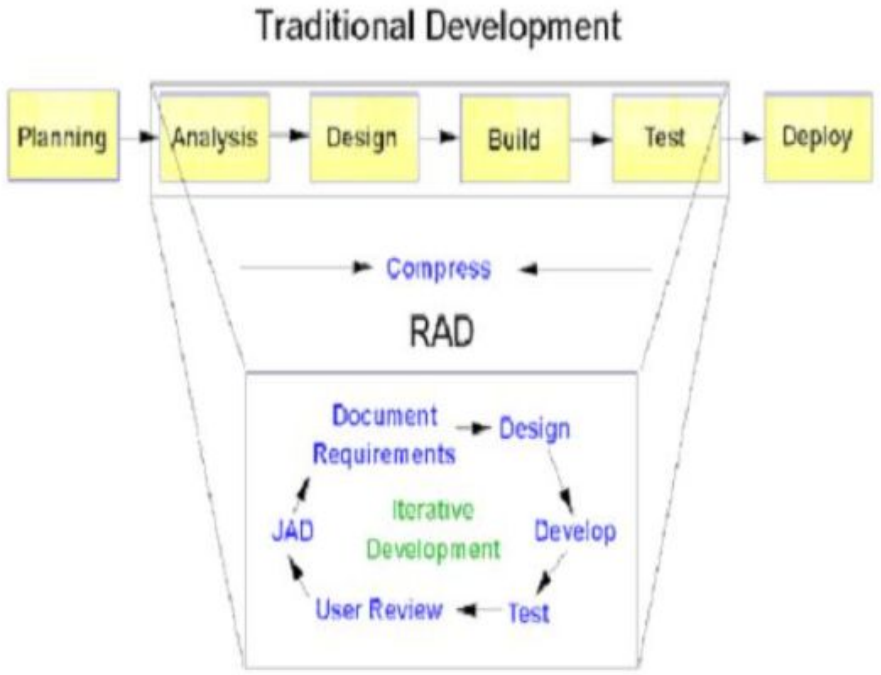
Testing and Turn Over:

- Conduct various testing stages, such as:
    - Unit Testing: Test individual components and functions of the loan management system for correctness and functionality.
    - Integration Testing: Verify the seamless integration of different system modules and ensure proper data flow between them.
    - User Acceptance Testing: Engage loan officers and customers to test the system's usability, functionality, and accuracy in a real-world environment.
- Solving identified issues or bugs during the testing phase and make necessary refinements.
- Once the system is stable and meets the requirements, prepare it for deployment and turnover to the loan department. Provide training and support to loan officers and staff for a smooth loan processes.

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.

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.



The History of RAD

- RAD was introduced by James Martin

  IBM Brought the process to IBM to combat the same problems Gielan saw.

  After successful releases, Martin wrote a book, Rapid Application Development giving the new process.

- RAD was inspired by the following process model:

  Barry Boehm – spiral model

  Tom Gilb – evolutionary life cycle

  Scott Shultz – rapid iterative productive prototyping

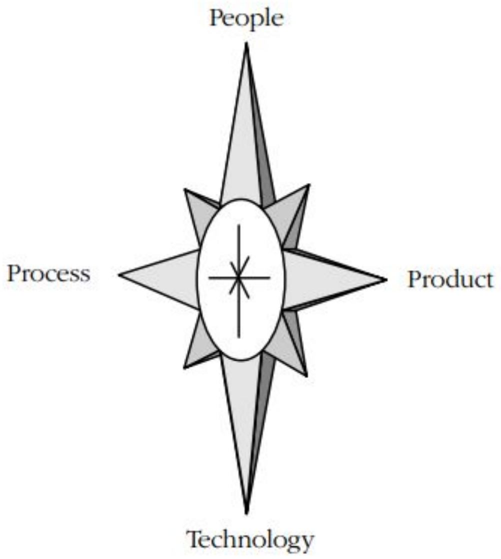
- One major pitfall in previously used methodologies was that the application took a long time to build

- The **requirements often changed** before the system was completed, causes for unusable/inefficient systems.

- Assumed as they have collected all critical requirements in the Requirement analysis phase.

Four dimensions of RAD

People
process
product
Technology



Four dimensions of Development Speed ..

1. People

figure: orchestra - different level of skilled people worked together

Team organization:
The way that people are organized has a great effect on how efficiently they can work. Software shops can benefit from tailoring their teams to match project size, product attributes, and schedule goals. A specific software project can also benefit from appropriate specialization.

Motivation:
Motivation is how potential or strongest you are on a rapid-dev project.

Four dimensions of Development Speed ..

Five principles of Staff selection for team projects.

- *Top talent* - Use better(skillful) and fewer people.
- *Job matching* - Fit the tasks to the skills and motivation of the people available.
- *Career progression* - Help people to self-actualize rather than forcing them to work where they have the most experience or where they are most needed.
- *Team balance* - Select people who will complement and harmonize with each other.
- *Misfit elimination* - Eliminate and replace the problem by team members as quickly as possible.

Four dimensions of Development Speed ..

2. Process:

Quality assurance.
Quality assurance has two main purposes. The first purpose is to assure that the product you release has an **acceptable level of quality.** The second function of quality assurance is to detect errors at the stage when they are least time-consuming (and least costly) to correct. This nearly always means catching errors as close as possible to the time that they are introduced. The longer an error remains in the product, the more time consuming (and more costly) it will be to remove. Quality assurance is thus an indispensable part of any serious rapid-development program.

Risk management.
One of the specific practices that's focused on avoiding disaster is risk management. Developing rapidly isn't good enough if you get your feet knocked out from under you two weeks before you're scheduled to ship. Managing schedule-related risks is a necessary component of a rapid development program.

Four dimensions of Development Speed ..

2. Process:

Development fundamentals.

A lot of that work has focused on "**productivity**" rather than **speed** of development, some of it has been oriented toward getting the same work done with fewer people rather than getting a project done faster.

Resource targeting.

Resources can be focused effectively and contribute to overall productivity, or they can be misdirected and used ineffectively. Best practices such as productivity offices, timebox development, accurate scheduling, and voluntary overtime help to make sure that you get as much work done each day as possible.

Lifecycle planning.

One of the keys to targeting resources effectively is to apply them within a lifecycle framework that makes sense for your specific project. Without an overall lifecycle model, you can make decisions that are individually on target but collectively misdirected. A lifecycle model is useful because it describes a basic management plan. For example, if you have a risky project, a risk-oriented lifecycle model will suit you; and if you have vague requirements, an incremental lifecycle model may work best. Lifecycle models make it easy to identify and organize the many activities required by a software project so that you can do them with the utmost efficiency.

Four dimensions of Development Speed ..

2. Process:

Customer orientation.

software development styles has been the switch to a strong focus on customers' needs and desires. Developers have learned that developing software to specification is only half the job. The other half is helping the customer figure out what the product should be, and most of the time that requires an approach other than a traditional paper-specification approach.

Putting yourself on the same side as the customer is one of the best ways to avoid the massive rework caused by the customer deciding that the product you just spent 12 months on is not the right product after all. The best practices of staged releases, evolutionary delivery, evolutionary prototyping, throwaway prototyping, and principled negotiation can all give you leverage in this area.

Four dimensions of Development Speed ..

3 Product

Product size.
Product size is the largest single contributor to a development schedule. Large products take a long time. Smaller products take less time. Additional features require additional specification, design, construction, testing, and integration. They require additional coordination with other features, and they require that you coordinate other features with them. Because the effort required to build software increases disproportionately faster than the size of the software, a reduction in size will improve development speed disproportionately. **Cutting the size of a medium-size** program by one half will typically cut the effort required by almost *two-thirds*. You can reduce product size outright by striving to develop only the most essential features, or you can reduce it temporarily by developing a product in stages. You can also reduce it by developing in a higher-level language or tool set so that each feature requires less code.

Product characteristics. Although not as influential as product size, other product characteristics do have an effect on software schedules. A product with ambitious goals for **performance, memory use, robustness**, and **reliability** will take longer to develop than a product without any goals for those characteristics. Choose your battles. If rapid development is truly top priority, don’t shackle your developers by insisting on too many priorities at once.

Four dimensions of Development Speed ..

4. Technology

Changing from less effective tools to more effective tools can also be a fast way to improve your development speed.
Choosing tools effectively and managing the risks involved are key aspects of a rapid-development initiative.

Why RAD?

- Rapid application development came as a response to problems when applying the waterfall model to software development.
  - 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. Meanwhile customer's business need could change.
  - Software is uniquely different than other types of engineering because changes can be made almost immediately and even very late in the development process
  - RAD compresses the step-by-step development of conventional methods into an iterative process.

With conventional methods, there is nothing until 100% of the process is finished, then 100% of the software is delivered.

Companies using RAD

SAP

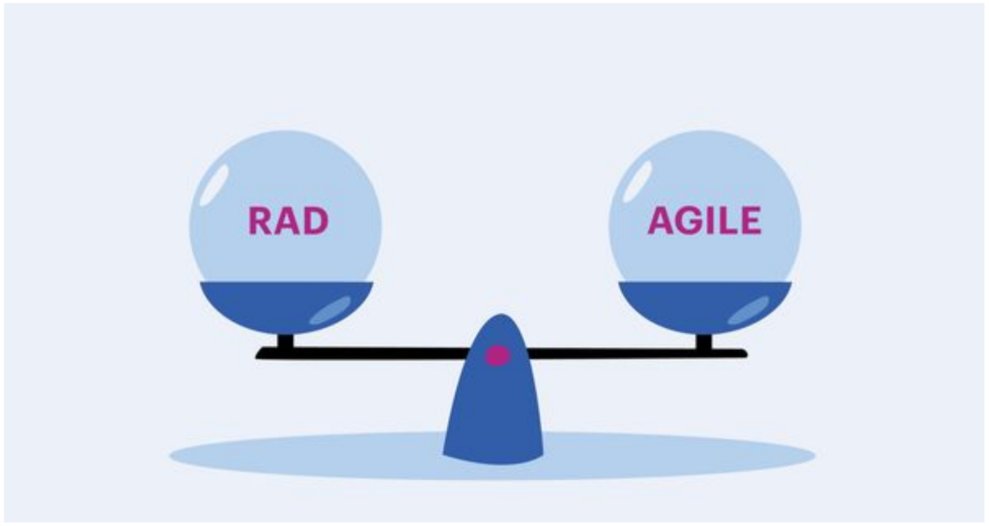
ORACLE

IBM

SYBASE®

Microsoft

Agile vs. rad are they similar?



Both are very similar

In Agile,

Rad is more focused on prototypes, work for to get something usable.

Working on the client feedbacks as much as possible.

How does Rad differ from Agile

RAD	Agile
Developers focus on creating the feature (no matter how bad it is at first) and then improve it	The team breaks down the solution into features
RAD teams are managed by the project manager (PM)*	Team members are self-managing*
The Agile standards described aren't use into RAD	Developers find and fix the bugs in the code as quickly as possible, and the team has the confidence to change the code without breaking the product.
Developers work as individuals (often results in unmaintainable and poorly designed code)	Agile teams focus on communication and developing the product as a team

Conclusion

- RAD has successfully achieved the objective of reducing costs on project not compromising on quality.
- Rapid Application Development model is a software development process based on prototyping without any specific planning.
- Encouraging the involvement of customers in the entire process of its development lifecycle.
- RAD has also demonstrated strength in being able to speed up the development process by appropriately combine its methodology, people, management and high tech computer aided tools.
- RAD has also proven to have challenges.
- RAD is good for: Small – Medium sized projects with changing technology Projects that need high flexibility RAD is bad for: Projects that need highly formalized standards Large projects.

References

- “Rapid Application Development” by Steve McConnell